

# ИНФОРМАТИКА

Учебник для 10 класса естественно-математического направления  
общеобразовательной школы

# 10

Рекомендовано Министерством образования и науки  
Республики Казахстан



Алматы «Атамұра» 2019

УДК 373.167.1  
ББК 32.973 я 72  
И 74

*Учебник подготовлен в соответствии с Типовой учебной программой по предмету «Информатика» для 10–11 классов естественно-математического направления уровня общего среднего образования по обновленному содержанию, утвержденной Министерством образования и науки РК.*

Авторы: **Д. Н. Исабаева, Л. Б. Рахимжанова, Е. А. Киселёва, Н. А. Курмангалиева, М. А. Аубекова.**

### УСЛОВНЫЕ ОБОЗНАЧЕНИЯ:



Выполним вместе



Знание и понимание



Применение, анализ



Синтез



Оценка



Подумай



Индивидуальная работа



Парная работа



Групповая работа

**И 74 Информатика:** Учебник для 10 кл. естественно-математического направления общеобразоват. шк./Д. Н. Исабаева, Л. Б. Рахимжанова, Е. А. Киселёва, Н. А. Курмангалиева, М. А. Аубекова. – Алматы: Атамұра, 2019. – 304 с.

ISBN 978-601-331-524-9

УДК 373.167.1  
ББК 32.973 я 72

ISBN 978-601-331-524-9

© Исабаева Д. Н., Рахимжанова Л. Б., Киселёва Е. А., Курмангалиева Н. А., Аубекова М. А., 2019  
© «Атамұра», 2019

### Дорогие ученики!

В текущем учебном году вы начнете изучать адаптивный курс информатики. Вы подробно изучите такие разделы, как «Компьютерные сети и информационная безопасность», «Представление данных», «Алгоритмизация и программирование», «Web-проектирование», «Информационные системы».

В этих разделах вы познакомитесь с:

- сетевыми устройствами;
- IP-адресацией;
- методами защиты информации;
- переходом от одной системы счисления к другой, логическими операциями (дизъюнкция, конъюнкция, инверсия), построением таблиц истинности, принципами кодирования текстовой информации;
- пользовательскими функциями и процедурами, работой со строками на языке программирования Python;
- работой с файлами, методами сортировки, алгоритмами на графах;
- основными понятиями базы данных, созданием базы данных, структурированными запросами;
- HTML, CSS, использованием скриптов, внедрением мультимедиа.

Такое распределение позволит вам усваивать материал последовательно, опираясь на знания и умения, полученные ранее, а также на опыт, приобретенный в ходе использования в повседневной жизни различных цифровых устройств.

Изучая каждую тему, вы не только узнаете теорию вопроса, подкрепленную большим количеством примеров, но и выполните практические задания. Работа индивидуально, в парах или по группам позволит практически применить полученные знания и умения. Проверить себя вы можете, выполнив задания, направленные на знание фактов, терминологии, понимание главной идеи, применение знаний в практической ситуации и решение продуктивных задач. Некоторые задания предполагают умение классифицировать элементы и связи, анализировать изученный материал и синтезировать идеи, формулировать собственные суждения относительно методов или решений.

В конце каждой главы вам предлагается выполнить тестовые задания, которые соответствуют уровням целей обучения по таксономии Б. Блума.

Глоссарий поможет вам вспомнить понятия и определения. Если вам нужна дополнительная информация, ее можно получить, изучив учебные материалы, указанные в списке использованной литературы.

Теоретические сведения и разноуровневые задания, представленные в учебнике, позволят вам использовать полученные знания в вашей повседневной жизни и дадут возможность в будущем способствовать формированию цифрового Казахстана.

**ЖЕЛАЕМ УСПЕХА!**

# 1.1. Сетевые устройства: узлы, коммутаторы

Можно ли использовать на нескольких компьютерах одно периферийное устройство?

Назначение компонентов сети (узлы, маршрутизаторы, коммутаторы).

**Узел** – Торан – Node  
**Маршрутизатор** – Маршрутизатор – Router  
**Адаптер** – Адаптер – Adapter  
**Коммутатор** – Коммутатор – Switch  
**Концентратор** – Концентратор – Хаб

Если бы Facebook был государством, то это было бы третье по численности населения государство в мире.

вы узнаете

ключевые слова

это интересно

## КЛАССИФИКАЦИЯ КОМПЬЮТЕРНЫХ СЕТЕЙ

### ЗАПОМНИ

#### АРХИТЕКТУРА СЕТИ

– это реализованная структура сети передачи данных, определяющая ее топологию, состав устройств и правила их взаимодействия в сети.

На сегодня существует огромное количество компьютерных сетей, которые классифицируются по различным признакам. **Проводные** и **беспроводные** сети различают по используемой среде передачи данных, **низкоскоростные, среднескоростные** и **высокоскоростные** – по скорости передачи данных, **глобальные, региональные** и **локальные** сети – по размеру охваченной территории.

Рассмотрим характеристики и описание сетей по территориальному делению.

Если компьютеры находятся географически на больших расстояниях друг от друга, например, в различных городах, странах и континентах, то их объединяют в **глобальные сети** (*Wide Area Networks, WAN*). В глобальные сети также могут объединяться отдельные компьютеры, локальные и региональные сети. Интернет является первой, самой большой и популярной глобальной сетью.

Если сети компьютеров расположены на ограниченной размером (в радиусе не более 1–2 км) небольшой территории, то их объединяют **локаль-**



**ными сетями** (*Local Area Networks, LAN*). Локальная сеть является коммуникационной системой, относящейся к одной организации, например, школьная локальная сеть. Первыми исторически появились глобальные сети, а затем – локальные.

Если сети предназначены для обслуживания территории крупного города или региона, то они называются **региональными сетями**, или **сетями мегаполисов** (*Metropolitan Area Networks, MAN*), соответственно.

Если с помощью глобальных сетей, например, Интернета, предприятие создает сеть путем объединения нескольких территориально разделенных локальных сетей своих филиалов (рис. 1.1.1), то она называется **виртуальной частной сетью** (англ. *Virtual Private Network, VPN*).

Для создания локальной сети требуются следующие физические устройства:

- кабели;
- сетевые адаптеры;
- сетевые устройства.



Рис. 1.1.1. Виртуальная частная сеть

Чем отличаются коаксиальные, оптоволоконные, кабели на витых парах, с помощью которых работает проводная сеть? Что такое коннекторы и для чего они используются?

Будем считать, что кабели проложены в нужных местах и установлены панели для подключения сетевых устройств. Теперь требуется выбрать сетевые устройства, которые позволяют объединить компьютеры в единую сеть.

Рассмотрим наиболее часто используемые сетевые устройства.

**Узел** сети (англ. *node*) – устройство, соединенное с другими устройствами как часть компьютерной сети. Узлами могут быть такие **пользовательские сетевые устройства**, как компьютеры, средства связи, сканеры, принтеры, медиаустройства и другие, а также **специальные сетевые устройства**, такие как маршрутизатор, коммутатор или концентратор, Wi-Fi и Bluetooth-адаптеры и другие.

Для работы в сети каждому узлу необходима плата сетевого интерфейса (*Network Interface Card – NIC*), которая называется **сетевым адаптером** (рис. 1.1.2). Сетевой кабель подключается к разъемам сетевого адаптера.



Рис 1.1.2.  
Сетевой адаптер

Сетевой адаптер бывает **проводным** или **беспроводным**. Многие современные компьютеры имеют встроенный сетевой адаптер Ethernet и Wi-Fi, которые находятся на материнской плате. Если в компьютере нет сетевого адаптера, то его легко можно установить в слот расширения компьютера или в порт USB.

Следует также установить специальное программное обеспечение – **драйвер** сетевого адаптера, который позволяет операционной системе (ОС) работать с этим устройством. Современные ОС (например, Windows 10) сами распознают устройство и устанавливают для него необходимый драйвер. Если с автоматически установленным драйвером сеть не работает, то драйвер необходимо установить вручную с официального сайта производителя адаптера.

Каждый сетевой адаптер имеет уникальный код, который называется **MAC-адресом**. Этот

## ЗАПОМНИ

### MAC-АДРЕС (ИЛИ АППАРАТНЫЙ АДРЕС)

– это цифровой код длиной 6 байт, устанавливаемый производителем сетевого адаптера и однозначно идентифицирующий данный адаптер.

адрес применяется при организации работы данных устройств в сети. По стандартам сети Ethernet нельзя использовать сетевые адаптеры с одинаковыми MAC-адресами. MAC-адрес записывается следующим образом: **00:E0:18:C3:11:89**.

Для объединения устройств сети в сегменты предназначено сетевое устройство, которое называется **концентратором**. Основной его функцией является трансляция информационных пакетов, поступающих на один из его портов, на все другие порты. Другими словами, пакет, поступивший в сеть, будет отправлен всем остальным устройствам сети.

Необходимо отметить, что большое количество современных компьютерных сетей строится на коммутаторах.

**Коммутатор** обеспечивает передачу данных между компьютерами одного сегмента сети (рис. 1.1.3). Внешний вид коммутатора может не отличаться от концентратора, однако принцип работы коммутатора совершенно другой. Вместо того, чтобы передавать данные всем компьютерам в сети, коммутатор передает пакеты данных только тому компьютеру, для которого эти пакеты предназначены. Другими словами, благодаря коммутатору организовывается прямое соединение между компьютером, запросившим данные, и компьютером, передающим запрошенную информацию. Благодаря прямому соединению между системами для каждого порта коммутатора доступна вся свободная пропускная способность сети.

Другим устройством, которое более эффективно используется для соединения локальных сетей с общим сетевым протоколом, является **маршрутизатор** (рис. 1.1.4). Он дает возможность, например, разделять сообщения большего объема на более мелкие части, тем самым организуя взаимодействие локальных сетей с различными размерами пакета.

В обычной локальной сети маршрутизатор обеспечивает соединение локальной сети (LAN) с глобальной (WAN). Он анализирует сетевые пакеты



Рис. 1.1.3. Коммутаторы

## ЗАПОМНИ

### МАРШРУТИЗАЦИЯ

– это процесс выбора маршрута следования пакета. Устройство, совершающее этот выбор, называется **маршрутизатором**.



Рис. 1.1.4.

Маршрутизатор



**ЗАПОМНИ****ПАКЕТ (КАДР)**

в телекоммуникации и компьютерных сетях – это фрагмент (блок данных) определенного формата для передачи данных по каналу связи.

**ЗАПОМНИ****СЕТЕВОЙ ПРОТОКОЛ**

– набор правил и действий (очередности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включенными в сеть устройствами.

данных и определяет не только компьютер, для которого эти пакеты предназначены, но и самый оптимальный способ их доставки. Маршрутизаторы применяются для соединения между собой различных сетей, а также для обеспечения доступа к Интернету из локальной сети. Современные маршрутизаторы часто оснащаются модемами ADSL или кабельными модемами, могут быть беспроводными или предназначенными для работы в качестве коммутатора.

Для обеспечения безопасности многие маршрутизаторы поддерживают шифрование данных и содержат брандмауэр (программно-аппаратный комплекс), используемый для защиты от внешних атак. Так же, как и к коммутаторам, к маршрутизаторам можно подключать другие маршрутизаторы.



Проверим наличие сетевых адаптеров.

- 1) Выбираем **Пуск – Панель управления**, а затем выберем пункт **Диспетчер устройств** из списка результатов.
- 2) В **Диспетчере устройств** нужно выбрать **Сетевые адаптеры**, а затем – имя сетевого адаптера.
- 3) После этого необходимо нажать и удерживать (или щелкнуть правой кнопкой мыши) имя сетевого адаптера, а затем выбрать **Обновить драйвер – Автоматический поиск** обновленных драйверов. Следуем указаниям, а затем выбираем **Заккрыть**. Если будет обнаружен новый драйвер, он скачается и установится автоматически.
- 4) Если после установки обновленного драйвера потребуются перезапуск, нужно нажать **Пуск – Выключение – Перезапуск**, а затем проверить, помогло ли это решить проблему подключения.





1. Назовите распространенные архитектуры сетей.
2. Приведите примеры узлов сети.
3. Что такое MAC-адрес?
4. Для чего нужен сетевой адаптер? Коммутатор? Маршрутизатор? Концентратор?
5. Назовите отличие концентратора от коммутатора.



1. Определите MAC-адрес компьютера в ОС Windows 10.  
– Нужно выполнить следующие действия: **Пуск – Выполнить – cmd** и нажать **ОК**. В командной строке набрать **ipconfig /all** и нажать **Enter** (рис. 1.1.5);



```

Ethernet адаптер:
DNS-суффикс этого подключения . . . :
Описание . . . . . : Marvell Yukon 88E
Ethernet Controller
Физический адрес . . . . . : 00-17-31-A7-CD-21
Dhcp включен . . . . . : да
Автонастройка включена . . . . . : да
IP-адрес . . . . . : 192.168.13.81
Маска подсети . . . . . : 255.255.255.0
Основной шлюз . . . . . : 192.168.13.150
DHCP-сервер . . . . . : 192.168.13.101
DNS-серверы . . . . . : 192.168.13.101
  
```

Рис. 1.1.5. Показан аппаратный адрес ПК

- найти пункт **Физический адрес** – это и есть MAC-адрес. Если на компьютере установлено несколько сетевых карт, то пунктов **Физический адрес** может быть несколько.
2. Найдите другие методы определения MAC-адреса компьютера.



- Составьте презентацию на тему «Сетевые устройства» с использованием web-сайта Canva.com.

# 1.2. IP-адресация. Система доменного именования

Что такое адрес? Для чего он используется?

– Что такое IP-адрес и зачем он нужен;  
– что такое домен и DNS.

**IP-адрес** – IP-мекенжай – IP address

**Маршрутизатор** – Маршрутизатор – Router

**Домен** – Домен – Domain

**Доменное имя** – Домендік атау – Domain name

**Доменная зона** – Домендік аймақ – Domain zone

Для удобства распределения и назначения доменных имен для каждой из стран были выделены собственные (в основном двухсимвольные) домены верхнего уровня. Но это не означает обязательную привязку серверов в данных домена к их географическому расположению.

Примеры доменов первого уровня для стран:

**.kz** – Kazakhstan (Казахстан);

**.ru** – Russia (Россия);

**.uk** – United Kingdom (Великобритания).

вы узнаете

ключевые слова

это интересно

## ЗАПОМНИ

### IP-АДРЕС

– это уникальный адрес в сети, необходимый для нахождения, передачи и получения информации от одного компьютера (узла) к другому.

Для работы в Интернете требуется глобальная уникальность IP-адреса. Для частной сети достаточно, чтобы были исключены совпадения в локальном пространстве. IP-адрес определяет одно сетевое соединение, а не отдельный компьютер или маршрутизатор.

## ЗАПОМНИ

### ОКТЕТ

– восемь двоичных разрядов.

## СТРУКТУРА IP-АДРЕСА

**IP-адрес** – это уникальная 32-разрядная последовательность двоичных цифр, разделенная на группы по 8 бит, называ-

емых **октетами**. С его помощью компьютер однозначно определяется в 8-разрядном виде в IP-сети (рис. 1.2.1).

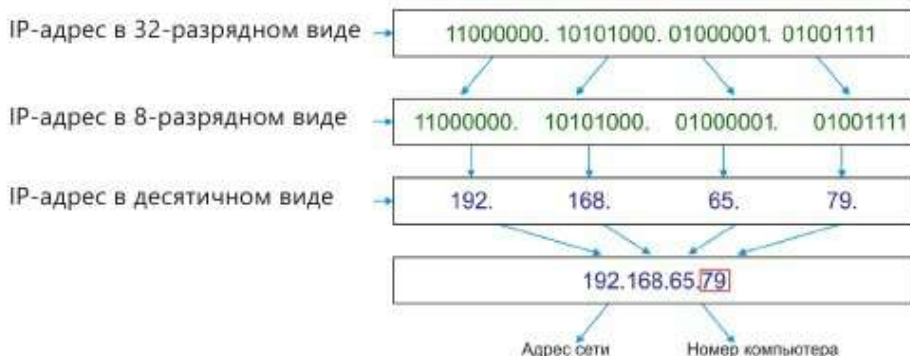


Рис. 1.2.1. Структура IP-адреса

### КЛАССЫ IP-АДРЕСОВ

Существует пять классов IP-адресов: **A**, **B**, **C**, **D** и **E**. Первые биты IP-адресов обозначают принадлежность к классу (табл. 1.2.1).

Классы IP-адресов						ТАБЛИЦА 1.2.1
Класс	A	B	C	D	E	
Первые биты	0	10	110	1110	11110	

### ПРАВИЛА НАЗНАЧЕНИЯ IP-АДРЕСОВ

IP-адрес назначается в соответствии с физическим подключением узла к тому или иному сегменту. Если узел перемещается в другой сегмент, то IP-адрес должен изменяться. Маршрутизаторы определяют границы сегментов.

Компьютерная сеть образует один сегмент, когда она собрана на коммутаторах. У всех компьютеров данного сегмента в IP-адресе должна быть общая часть – номер сегмента. Вторая часть будет номером устройства в сегменте. Поэтому IP-адрес принято разделять на две части. В связи с этим вместе с IP-адресом задается маска подсети. Обычно во всех сегментах используют одну и ту же маску.

Чаще всего применяют три маски:

- **255.0.0.0** – маска класса А;
- **255.255.0.0** – маска класса В;
- **255.255.255.0** – маска класса С.

**Маска подсети** (*subnet mask*) – это число, которое применяется в паре с IP-адресом; двоичная запись маски содержит единицы в тех разрядах, которые должны в IP-адресе интерпретироваться как номер сети.

Маску подсети описывают аналогично записи IP-адреса, например 255.255.255.0, либо вместе с IP-адресом, указав число единичных разрядов в записи маски, например 192.168.1.1/24. Это значит, что в маске содержится 24 единицы.

## ВИДЫ IP-АДРЕСОВ

Для систематизации интернет-пространства адреса разделяют на **внутренние** и **внешние**. Внутренние применяются в локальных сетях, а внешние – для выхода в Интернет. Внешний IP-адрес бывает статическим или динамическим.

**Статические IP-адреса** являются неизменными (постоянными). Они задаются устройству автоматически в момент его подключения к компьютерной сети или прописываются пользователем вручную. Статические адреса доступны для применения неограниченное время. Они предназначены для идентификации только одного сетевого узла.

**Динамические IP-адреса** присваиваются устройству на время. Они выдаются автоматически в момент подключения к сети и имеют ограниченный срок действия.



### КАК УЗНАТЬ IP-АДРЕС СВОЕГО КОМПЬЮТЕРА?

Существует много способов определения IP-адреса компьютера. Для этого можно воспользоваться средствами Windows. Чтобы узнать свой локационный номер, необходимо:

1. Открыть меню **Пуск – Служебные** и выбрать в нем опцию **Выполнить**.
2. В окне, которое появилось, напечатать **cmd** и нажать **ОК**.
3. Написать **ipconfig**.





4. После нажатия клавиши **Enter** внизу страницы можно увидеть основную информацию об IP-адресе подключения вашего компьютера (рис. 1.2.2).

```

Администратор: Командная строка
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Users\XE>ipconfig

Настройка протокола IP для Windows

Ethernet adapter Подключение по локальной сети 2:

DNS-суффикс подключения . . . . . : None
IPv6-адрес . . . . . : Fde8:c732:41f5:0:7d9b:7564:9296:9cf1
Временный IPv6-адрес . . . . . : Fde8:c732:41f5:0:84aa:14a3:3804:827d
Локальный IPv6-адрес канала . . . . : Fe80::7d9b:7564:9296:9cf1%14
IPv4-адрес . . . . . : 192.168.1.3
Маска подсети . . . . . : 255.255.255.0
Основной шлюз . . . . . : 192.168.1.1

C:\Users\XE>
  
```

Рис. 1.2.2. Окно **Определение IP-адреса**

## ДОМЕННАЯ СИСТЕМА ИМЕН

IP-адрес сложно запомнить. В связи с этим создана доменная система имен, которая позволяет пользователям применять вместо IP-адреса более понятное название.

**Домен** (англ. *domain*) – это определенная область пространства уникальных имен в сети Интернет. Обслуживание каждой из этих областей осуществляется группой серверов DNS (*Domain Name System* – система доменных имен), которая применяется для изменения IP-адреса в доменное имя. Все домены определяются специфическим уникальным именем.

**Доменное имя** – символическое имя домена. Оно должно быть уникальным в рамках одного домена. Полное имя домена состоит из имен всех доменов, в которые он входит, разделенных точками. Например, полное имя *www.smk.edu.kz* обозначает домен третьего уровня, который входит в домен второго уровня *edu.kz*, который входит в домен *.kz*, который входит в корневой домен (рис. 1.2.3).



Рис. 1.2.3. Пример структуры доменных имен

### ДОМЕНЫ ВЕРХНЕГО УРОВНЯ ОБЩЕГО НАЗНАЧЕНИЯ

- .com** – коммерческие организации (без ограничений);
- .edu** – высшие учебные заведения, признаваемые в качестве таковых Министерством образования и науки РК;
- .info** – информационные ресурсы (без ограничений);
- .name** – физические лица;
- .net** – организации, имеющие отношение к функционированию Интернета (без ограничений);
- .org** – некоммерческие организации (без ограничений).

### НАБЛЮДЕНИЕ ЗА DNS-ПРЕОБРАЗОВАНИЕМ URL-АДРЕСА В IP-АДРЕС



1. Нажмите кнопку **Пуск – Службные – Выполнить**, введите в поле поиска **cmd** и нажмите клавишу **Enter**. Откроется окно командной строки.
2. В командной строке введите команду **ping**, нажмите пробел и введите адрес сайта. Например, *www.icann.net*. (ICANN – корпорация по присвоению имен и номеров в Интернете). Компьютер должен преобразовать **www.icann.net** в **IP-адрес**, чтобы определить, куда отправлять пакеты протокола **ICMP (протокола управления сообщениями в сети Интернет)**.
3. В первой строке выходных данных отображается доменное имя **www.icann.net**, преобразованное DNS в IP-адрес (рис. 1.2.4).
4. Запишите IP-адрес сайта **www.icann.net**.



```

Выбрать Командная строка

C:\Users\User>www.icann.net
"www.icann.net" не является внутренней или внешней
командой, исполняемой программой или пакетным файлом.

C:\Users\User>ping www.icann.net

Обмен пакетами с www.icann.net [192.0.43.22] с 32 байтами данных:
Ответ от 192.0.43.22: число байт=32 время=185мс TTL=242
Ответ от 192.0.43.22: число байт=32 время=185мс TTL=242
Ответ от 192.0.43.22: число байт=32 время=185мс TTL=242
Ответ от 192.0.43.22: число байт=32 время=196мс TTL=242

Статистика Ping для 192.0.43.22:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 185мсек, Максимальное = 196 мсек, Среднее = 187 мсек

C:\Users\User>
  
```

Рис. 1.2.4. DNS-преобразование

5. Вместо URL-адреса введите в адресную строку браузера IP-адрес, полученный при выполнении шага 4. Откроется домашняя страница web-сайта **ICANN**.

Большинству людей проще пользоваться словами, чем цифрами, поэтому адрес web-сайта **www.icann.net** запомнится пользователями с гораздо большей вероятностью, чем IP-адрес **192.0.43.22**.



1. Что такое IP-адрес? Какова его структура?
2. Назовите возможные способы описания IP-адресов.
3. Что такое классы IP-адресов?
4. Что такое маска подсети? Для чего она нужна?
5. Для чего разделяют IP-адреса на идентификаторы сети и узла?
6. Что такое доменные имена? Каково их назначение?
7. В чем заключается основное преимущество DNS?
8. Объясните структуру доменных имен.
9. Откройте понравившийся сайт и запишите его IP-адрес. Что произойдет, если скопировать IP-адрес и вставить его в адресную строку браузера?



1. Скачайте программу **LanCalculator** и с ее помощью выполните: **Пуск – Программы – Стандартные – Калькулятор**, потом **Вид – Программист (DEC – десятичная система, BIN – двоичная система)**. С помощью калькулятора преобразуйте IP-адреса в двоичную форму.

- 1) 192.168.1.3.
- 2) 192.0.68.15.
- 3) 23.85.0.15.
- 4) 45.10.23.87.

2. Определите класс сети по заданному IP-адресу.

- 1) 100.110.120.130.
- 2) 140.160.180.200.
- 3) 160.180.200.220.
- 4) 180.200.220.240.
- 5) 200.210.220.230.

3. Зайдите на сайт **www.nic.kz** или **ps.kz** и проверьте информацию о доменных именах вашей школы, известных университетов и других популярных сайтов (рис. 1.2.5).

Рис. 1.2.5. Сайт **www.nic.kz**



Создайте презентацию на тему «IP-адрес и доменные имена».



## ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ К §§1–2.



Программа **NetEmul** (можно скачать по адресу <https://sourceforge.net/projects/netemul/files/release-1.0/netemul-1.0.exe/download>) используется для визуализации работы компьютерных сетей и облегчения понимания происходящих в сети процессов.



**ПРИМЕР.** Построить простейшую локальную сеть.

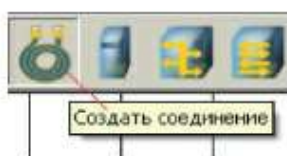
1. Установим программу и настроим ее на русский язык, выбирая команды **Сервис – Настройки**.

2. Выполним команду **Файл – Новый** и нарисуем схему сети, как на *рис. 1.2.6*.

После размещения трех ПК и концентратора можно создать их соединение (*рис. 1.2.7*).

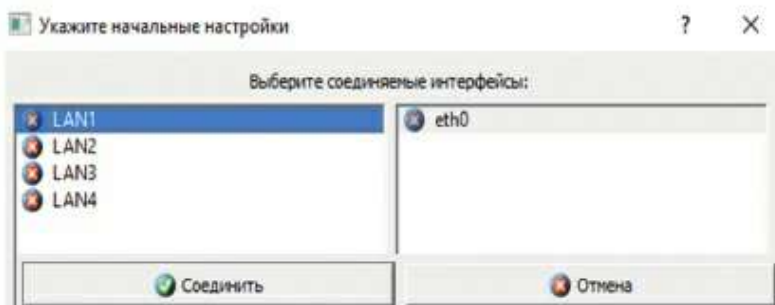


**Рис. 1.2.6.** Схема с тремя компьютерами и концентратором



**Рис. 1.2.7.** Инструмент создания соединений сетевых устройств

Создавая связи между устройствами, выберем соединяемые интерфейсы и нажмем на кнопку **Соединить** (*рис. 1.2.8 и 1.2.9*).



**Рис. 1.2.8.** Выбор начальных настроек соединения



Рис. 1.2.9. Соединение устройств произведено

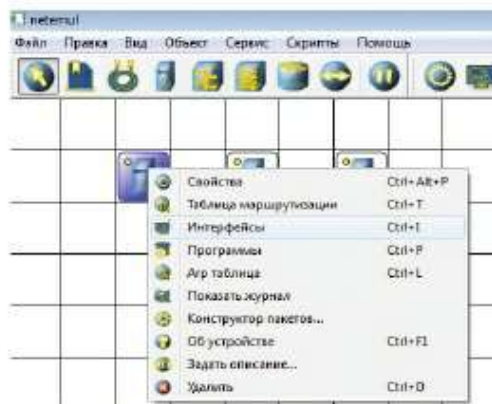


Рис. 1.2.10. Настройка интерфейса

Теперь на компьютере настроим интерфейс (сетевую карту) (рис. 1.2.10).

**ПРИМЕЧАНИЕ.** После написания **192.168.0.1** маска появляется автоматически. При нажатии на кнопку **Применить** и **ОК** появляется анимация движущихся по сети информационных пакетов (рис. 1.2.11).



Рис. 1.2.11. Кнопка **Отправить данные**



Рис. 1.2.12. Выбор протокола

В результате сеть создана и настроена. Нужно отправить данные по **протоколу TCP** (рис. 1.2.12 и рис. 1.2.13).

Если где-то была ошибка, то появится сообщение, а если ошибок нет, то произойдет анимация движущихся по сети пакетов (рис. 1.2.14).

Необходимо знать, что каждый компьютер имеет одну или несколько сетевых карт. Для того чтобы добавить для компьютера адаптер, нужно щелкнуть на нем правой кнопкой мыши и выбрать пункт меню **Интерфейсы**.

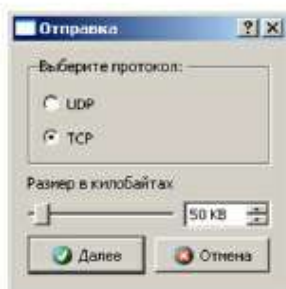


Рис. 1.2.13. Движение пакетов по сети

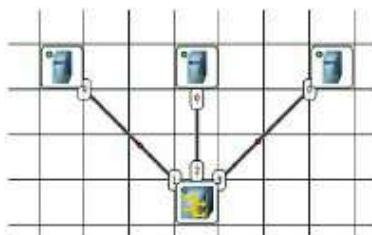


Рис. 1.2.14. Диалоговое окно работы с сетевым интерфейсом ПК

Чтобы создать еще один интерфейс, необходимо нажать на кнопку **Добавить**, выбрать тип нового адаптера, нажать **ОК**.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Постройте сеть из восьми компьютеров, концентратора, коммутатора и маршрутизатора, как на рисунке 1.2.15. Настройте ее правильную работу.

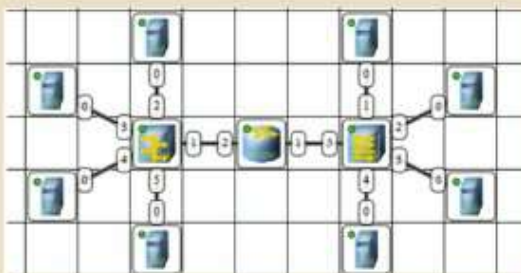


Рис. 1.2.15. Две подсети по топологии Звезда

# 1.3. Информационная безопасность

Как вы понимаете значения терминов «информационная безопасность», «конфиденциальность» и «целостность» данных?

Значения терминов «информационная безопасность», «конфиденциальность» и «целостность» данных.

**Информационная безопасность** – Ақпараттық қауіпсіздік – Information security  
**Конфиденциальность** – Құпиялылық – Confidentiality  
**Целостность** – Тұтастық – Integrity  
**Доступность** – Қолжетімділік – Availability

Действующий Уголовный кодекс Республики Казахстан предусматривает наказания за преступления, связанные с нарушением конфиденциальности информации. Глава 7 «Уголовные правонарушения в сфере информатизации и связи» содержит статьи 205–213, посвященные преступлениям, связанным с неправомерным доступом к компьютерной информации, созданием, использованием или распространением вредоносных компьютерных программ и программных продуктов.

вы узнаете

ключевые слова

это интересно

## ЗАПОМНИ

**«Информационная безопасность в сфере информатизации** (далее информационная безопасность) – состояние защищенности электронных информационных ресурсов, информационных систем и информационной инфраструктуры от внешних и внутренних угроз».

*Закон «Об информатизации» (11 апреля 2019 г.)*



Под информационной безопасностью также понимают процесс обеспечения конфиденциальности, целостности и доступности информации.

Задача информационной безопасности – предотвращение несанкционированного и непреднамеренного воздействия на защищаемую информацию и утечки информации по возможным каналам.

20



Какое место занимают правовые меры в системе защиты информации?

Защита информации обеспечивается правовыми, организационными и инженерно-техническими мерами.

**Правовые меры** законодательно регламентируют юридические вопросы в информационной сфере, устанавливают правила использования данных ограниченного доступа и устанавливают меру ответственности за нарушение этих правил. Республикой Казахстан принята Концепция кибербезопасности «Киберщит Казахстана» (от 30 июня 2017 года), основанная на оценке текущей ситуации и международного опыта в сфере информатизации общества и перспектив развития «цифровой» экономики и процессов расширения сферы оказания информационно-коммуникационных услуг. Концепция определяет основные направления реализации государственной политики в сфере защиты электронных информационных ресурсов, информационных систем и сетей телекоммуникаций, обеспечения безопасного использования ИКТ.

Правовое обеспечение содержит как международные, так и национальные правовые нормы (рис. 1.3.1).

## ЗАПОМНИ

Правовую основу информационной безопасности обеспечивает государство. Защита информации регулируется международными конвенциями, Конституцией Республики Казахстан, государственными законами.

## ЗАПОМНИ

Под **защитой информации** понимают порядок и правила применения принципов и средств защиты информации.

### ПРАВОВОЕ ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

#### Международные и национальные правовые нормы:

- |                   |                          |
|-------------------|--------------------------|
| – соглашения;     | – конституция;           |
| – договоры;       | – указы;                 |
| – лицензии;       | – кодексы;               |
| – патенты;        | – руководящие документы; |
| – авторские права | – нормативные акты       |

Рис. 1.3.1. Правовые меры обеспечения информационной безопасности

## 10 класс | информатика

К **международным правовым мерам** обеспечения безопасности информации относится ряд таких документов, как Глобальная программа кибербезопасности, Резолюция Генеральной Ассамблеи ООН, в которых содержатся подходы к пониманию кибербезопасности, охватывающие сферу безопасного использования информационно-коммуникационных технологий.

В последние годы национальная правовая система расширилась новыми документами и дополнениями в действующие нормативные акты по обеспечению информационной безопасности в области информатизации (Уголовный кодекс, Кодекс «Об административных правонарушениях», законы «О государственных секретах», «О персональных данных и их защите», «Об электронном документе и электронной цифровой подписи», «О связи» и целом ряде подзаконных актов, разработанных для реализации новой редакции Закона Республики Казахстан «Об информатизации», вступившего в силу с 1 января 2016 года).

Какие организационные и технические меры необходимо применять для обеспечения информационной безопасности?

**Организационные меры** регулируют поведение персонала при работе с информацией, в том числе ограниченного доступа (рис. 1.3.2).

<b>Организационное обеспечение информационной безопасности:</b>	
<ul style="list-style-type: none"> <li>– порядок хранения документов;</li> <li>– режим допуска;</li> <li>– соблюдение защитных мер при проектировании и разработке информационных систем и информационно-коммуникационной инфраструктуры</li> </ul>	<ul style="list-style-type: none"> <li>– контроль смены паролей;</li> <li>– выявление уязвимостей в системе защиты;</li> <li>– проведение тестирования средств защиты информации</li> </ul>

**Рис. 1.3.2.** Основные организационные меры обеспечения информационной безопасности

Технические мероприятия направлены на защиту информации от несанкционированного доступа к компьютерной системе, резервирование важных компьютерных систем, разработку и реализацию программных и аппаратных комплексов безопасности и т.д. (рис. 1.3.3).

**Техническое обеспечение информационной безопасности:**

– охрана и недопущение нарушений функционирования объектов информационно-коммуникационной инфраструктуры

– использование аппаратных, программных и криптографических средств защиты электронных информационных ресурсов и информационно-коммуникационной инфраструктуры

**Рис. 1.3.3.** Основные технические меры обеспечения информационной безопасности

Отметим, что рассмотренные меры защиты информации носят обобщенный характер. Для оценки состояния уровня защищенности информации с точки зрения уязвимости необходима система мер и использование специальных средств защиты.

Информация считается защищенной, если соблюдаются свойства конфиденциальности, доступности и целостности данных.

**Конфиденциальность** – это свойство информации быть доступной для просмотра и редактирования только тем пользователям, которые зарегистрированы в системе защиты. Нарушение конфиденциальности, или несанкционированный доступ (НСД), происходит, если доступ к информации получает неавторизованное лицо.

Для некоторой информации, защищаемой законом или владельцем, конфиденциальность – одно из важных требований. Например, служебная информация, личные данные ограниченного доступа, данные о клиентах банка или налоговых служб, сведения о здоровье пациентов и др. – конфиденциальная информация.

**Целостность** – это свойство информации (данных) сохранять свою структуру и содержание в процессе хранения, использования и передачи.

Не предусмотренные владельцем изменения информации, например ошибка оператора или преднамеренное действие постороннего лица, приводят к нарушению целостности. Существующие средства поддержки целостности данных предотвращают переход данных в несогласованное состояние и получение ошибочных (некорректных) результатов.



## 10 класс | информатика

Особенно важна целостность данных, связанных с функционированием различных объектов информационно-коммуникационной инфраструктуры, таких как автоматизированные системы управления воздушным движением, электро- и энергоснабжением и т.д.

**Доступность** – это свойство информации, которое подразумевает, что все зарегистрированные пользователи должны иметь доступ к конфиденциальной информации.

Достаточно нарушить одно из свойств, чтобы система перестала быть защищенной.

### ЗАПОМНИ

#### КОМПЬЮТЕРНЫЕ ВИРУСЫ

– разновидность вредоносных программ, которые могут создавать свои копии и внедрять их в файлы, системные области компьютера и компьютерных сетей и тем самым самостоятельно распространяться.

Применение и использование компьютеров практически во всех сферах деятельности человека имеют и негативные последствия, которые проявились в создании вредоносных программ.

Вредоносная программа предназначена для получения несанкционированного доступа к информации, а также для вмешательства в работу компьютера или программного обеспечения. Разработка

и использование вредоносных программ носят криминальный характер. В отчетах «Лаборатории Касперского» (<http://securelist.com/ru>) отмечается рост появления вредоносных программ и проведенных кибератак.

### ЗАПОМНИ

Вирусы можно классифицировать:

- **по поражаемому объектам:** файловые вирусы, загрузочные вирусы, скриптовые вирусы, макровирусы, вирусы, поражающие исходный код;
- **по поражаемым операционным системам и платформам:** Microsoft Windows, Linux;
- **по технологиям, используемым вирусом:** полиморфные вирусы, стелс-вирусы, руткиты;
- **по языку, на котором написан вирус:** ассемблер, высокоуровневый язык программирования, скриптовый язык и др.;
- **по дополнительной вредоносной функциональности:** бэкдоры, кейлоггеры, шпионы, ботнеты и др.



Кроме вредоносных программ, существуют и другие виды информационных угроз. Например, фишинг (*phishing*) представляет собой разновидность компьютерного мошенничества с целью кражи личных конфиденциальных данных.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Чтобы проверить уровень безопасности своего компьютера, выполните команду: **Пуск – Панель управления – Система и безопасность**.

2. В окне **Система и безопасность** перейдите по ссылке **Проверка состояния компьютера и решение проблем** (рис. 1.3.4).

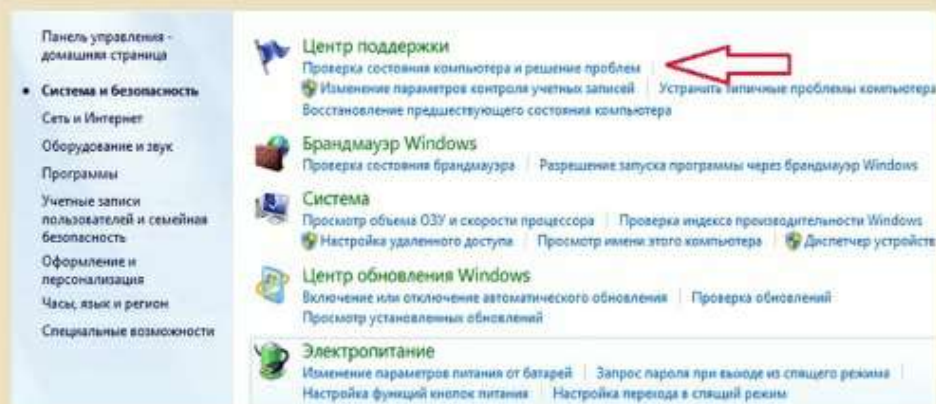


Рис. 1.3.4. Вид окна **Система и безопасность**

На экране появится окно **Центр поддержки**. Ознакомьтесь с информацией, имеющей отношение к антивирусным и антишпионским программам, установленным на компьютере.

Если Windows не обнаружила антивирусных и антишпионских программ на вашем компьютере, перейдите по ссылке **Найти программу в сети** и просмотрите рекомендуемые программы.



Обязательно установите антивирусные и антишпионские программы и регулярно их обновляйте. Благодаря им вы сможете предотвратить загрузку на свой компьютер вредоносного ПО.



1. Что такое информационная безопасность?
2. Что понимается под защитой информации?
3. Какими мерами обеспечивается защита информации?
4. К каким последствиям могут привести действия вредоносных программ?
5. Как проверить безопасность компьютера?
6. Какие средства защиты информации от несанкционированных воздействий имеются в ОС Windows?
7. Какие правила позволяют защитить компьютер от вредоносных программ?



1. Каковы отличительные особенности персонального компьютера как объекта защиты?
2. Какие свойства информации являются наиболее важными с точки зрения обеспечения ее безопасности?
3. Назовите основные цели Республики Казахстан в области обеспечения информационной безопасности.
4. Проблемы информационной безопасности в Казахстане регламентируются Концепцией «Киберщит Казахстана» (*egov.kz*). На основании этого документа поясните, какое место отводится обеспечению целостности и конфиденциальности данных.



1. Соотнесите правила информационной безопасности с примерами. Правила защиты: **1)** компьютера от вредоносных программ, спама, хакеров и мошенничества; **2)** авторских прав; **3)** личных данных.
  - А)** Используйте свой пароль для каждого аккаунта.
  - В)** Не предоставляйте личную информацию.
  - С)** Никогда не давайте номера документов.
  - Д)** Работайте с ограниченными правами на своем компьютере.
  - Е)** Внимательно работайте с электронной почтой, особенно с вложениями.
  - Ф)** Обновляйте операционную систему и самые распространенные программы.
  - Г)** Будьте осторожны при работе с внешними носителями информации.
  - Н)** Установка капчи.



I) Не доверяйте спаму.

J) Не переходите по ссылкам на сторонние сайты.



K) Периодически сканируйте свой компьютер на наличие вирусов с помощью дополнительных антивирусных утилит.

Самостоятельно дополните список примеров. К какой группе мер относятся придуманные вами примеры?

2. Создайте комикс на тему «Сетевой этикет». Расскажите свою историю (storytelling) и выполните ее доступным способом, используя программные средства.

## 1.4.

### Методы защиты информации

Как вы обычно защищаете свою конфиденциальную информацию?

Оценивать необходимость шифрования данных.

**Шифрование** –  
Шифрлеу – Encryption

31 марта – Международный день резервного копирования. Самый распространенный способ хранения данных – на жестких дисках компьютеров. Однако он не гарантирует, что однажды информация не будет утеряна.

вы узнаете

ключевые слова

это интересно

На сегодняшний день шифрование данных является одним из эффективных методов защиты информации.

Алгоритмы шифрования данных принято делить на следующие типы:

- асимметричные (с открытым ключом);
- симметричные (с закрытым ключом).

Симметричные алгоритмы используют один и тот же ключ и для шифрования, и для дешифрования. Основным недостатком симметричного шифрования является

#### ЗАПОМНИ

##### ШИФРОВАНИЕ ДАННЫХ (КРИПТОГРАФИЯ)

– это преобразование информации в другую форму таким образом, что только пользователь, имеющий секретный ключ, называемый ключом дешифрования, может получить к нему доступ.



## 10 класс | информатика

невозможность установить подлинность ключа. Секретная информация может быть легко расшифрована, поэтому чаще используется **комбинированный метод**.

Примеры симметричных алгоритмов:

- **DES (Data Encryption Standard)**, разработанный фирмой IBM и утвержденный в 1977 году как официальный стандарт;
- **AES (Advanced Encryption Standard)** – алгоритм блочного шифрования, принят в качестве стандарта шифрования.

Асимметричные алгоритмы используют два разных ключа: один открытый (для шифрования) и один закрытый ключ (для дешифрования). Они также отличаются уровнем устойчивости.

Пример асимметричного алгоритма:

- **RSA** (слово образовано от фамилий авторов *Rivest, Shamir и Adleman*) стала первой системой, пригодной и для шифрования, и для цифровой подписи.

Первым появилось симметричное шифрование, асимметричное было изобретено позже, чтобы обеспечить большую надежность. ЭЦП (электронная цифровая подпись) основана на асимметричном шифровании с открытым ключом и позволяет подтвердить авторство электронного документа. Подпись связана как с автором, так и с документом.

В соответствии с законодательством РК **электронный документ** – документ, в котором информация представлена в электронно-цифровой форме и удостоверена посредством электронной цифровой подписи (пп. 12 статьи 1 Закона Республики Казахстан «Об электронном документе и электронной цифровой подписи»).

## ЗАПОМНИ

### ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ (ЭЦП)

– это личная подпись, представляющая собой набор электронных цифровых символов, подтверждающая достоверность электронного документа, его принадлежность и неизменность содержания (рис. 1.4.1).



**Рис. 1.4.1.** Электронная цифровая подпись



## ЗАПОМНИ

### Основные преимущества шифрования данных:

- основная защита регулярно используемых конфиденциальных цифровых данных;
- защита от несанкционированного доступа баз данных, корпоративной почты и другой информации;
- предоставление доступа к конфиденциальным данным только доверенным сотрудникам;
- экстренная блокировка доступа к данным;
- обеспечение уверенности в надежной защите конфиденциальной информации.



Автоматизируем метод шифрования **Виженера** с использованием ключевого слова, например, слова **sport**. Допустим, что длина слова для шифрования не превышает 10 символов и состоит из строчных латинских букв. Зашифруем слова **bank**, **computer** и **musician**.

1. Для решения задачи используем текстовые функции Excel **СИМВОЛ** и **КОДСИМВ**.

2. Каждая буква текста должна храниться в отдельной ячейке.

3. Буквы латинского алфавита имеют подряд идущие номера (коды), поэтому порядковый номер буквы в алфавите равен коду данной буквы минус код буквы «а» плюс единица. Так вычисляется сдвиг, соответствующий букве ключевого слова. Величина сдвига соответствует буквам ключа и определяется формулой. Например, в ячейке **B3** находится формула: **=КОДСИМВ(B2)-КОДСИМВ("а")+1**

4. В строке 4 располагается шифруемое слово. В ячейках строки 5 помещаются формулы шифрования. Формула в ячейке **B5** такая: **=СИМВОЛ(КОДСИМВ("а")+ОСТАТ(КОДСИМВ(B4)-КОДСИМВ("а")+B3;26))**

5. Функция **СИМВОЛ(код символа)** возвращает символ по значению его кода. Функция **ОСТАТ(делимое; делитель)** возвращает остаток от целочисленного деления. Латинский алфавит содержит 26 букв. Остатки деления на 26 – числа в диапазоне от 0 до 25. Это позволяет оставаться в пределах кодов латинского алфавита (строчных букв): от кода буквы «а» до кода буквы «z» (рис. 1.4.2).

6. Дешифруйте слова (рис. 1.4.3). Как следует изменить формулу?



	A	B	C	D	E	F	G	H	I	J	K
1	<b>Шифр Виженера</b>										
2	ключ	s	p	o	r	t	s	p	o	r	t
3	сдвиг	19	16	15	18	20	19	16	15	18	20
4	исходный текст	b	a	n	k						
5	зашифрованный текст	u	q	c	c						
6	исходный текст	c	o	m	p	u	t	e	r		
7	зашифрованный текст	v	e	b	h	o	m	u	g		
8	исходный текст	m	u	s	i	c	i	a	n		
9	зашифрованный текст	f	k	h	a	w	b	q	c		

Рис. 1.4.2. Шифр Виженера

исходный текст									
зашифрованный текст	t	y	o	m	r				
исходный текст									
зашифрованный текст	h	v	g	w	x	p	d	s	u

Рис. 1.4.3. Дешифрование с помощью шифра Виженера



1. Что такое шифрование данных?
2. Какие типы шифрования существуют?
3. В чем заключается цель шифрования?
4. Назовите основные преимущества шифрования данных.



1. Проанализируйте преимущества резервного копирования и шифрования данных.
2. Заполните таблицу.

Достоинства и недостатки симметричного и асимметричного методов шифрования		
Симметричный метод		
Асимметричный метод		



Какие типовые проблемы решают системы шифрования данных?



Для разблокировки планшетных компьютеров и сотовых телефонов может использоваться графический ключ, который представляет собой одну ломаную линию, соединяющую какие-то узлы. Сколько различных вариантов графических ключей можно создать, если использовать в качестве узлов все вершины квадрата, причем каждую ровно один раз?

## 1.5. Методы идентификации личности

Приведите примеры из повседневной жизни.

Как аргументировать использование различных методов идентификации личности.

**Идентификация** –  
Сэйкестендipу –  
Identification  
**Аутентификация** –  
Аутентификациялау –  
Authentification

В последнее десятилетие интенсивно развивается направление электронной цифровой идентификации, в которой сбор информации происходит с минимальным участием человека. Это осуществляется благодаря технологиям, которые максимально соответствуют требованиям компьютерных систем распознавания объектов в реальном времени.

вы узнаете

ключевые слова

это интересно

Основой любых систем защиты являются **идентификация** и **аутентификация**.

Общая процедура идентификации и аутентификации пользователя при его доступе в систему представлена на *рис. 1.5.1*. Если в процессе аутентификации подлинность субъекта установлена, то система защиты информации должна определить его полномочия (права). Это необходимо для

**ЗАПОМНИ****ИДЕНТИФИКАЦИЕЙ**

называют процесс предъявления идентификатора.

**ИДЕНТИФИКАТОР** (*ID, data name, iden-tifier – опознаватель*)

– это уникальный признак объекта, позволяющий отличать его от других объектов.

**ЗАПОМНИ****АУТЕНТИФИКАЦИЕЙ**

называют процесс проверки подлинности или принадлежности пользователю предъявленного им идентификатора. Другими словами, аутентификация заключается в проверке, является ли подключающийся пользователь тем, за кого он себя выдает.



**Рис. 1.5.1.** Структура идентификации и аутентификации пользователя

последующего контроля и разграничения доступа к ресурсам. По направленности аутентификация может быть односторонней (пользователь доказывает свою подлинность системе, например при входе в систему) и двусторонней (взаимной).



Методы аутентификации:

- **все виды электронной подписи**;
- **одноразовые** (SMS-сообщения с кодом) и **многоразовые пароли** (PIN-коды, кодовые слова, цифры, графические ключи);
- **биометрические: физиологические и поведенческие** (отпечаток пальца, сетчатка глаза, тембр голоса, почерк и т.д.);
- **географическое местоположение** (геоданные GPS, точка доступа к Интернету).

Методы аутентификации, основанные на измерении биометрических параметров человека, обеспечивают почти 100%-ную идентификацию.

Основными элементами системы аутентификации, основанными на измерении биометрических параметров человека, являются:

- сканирующее устройство;
- программное обеспечение идентификации, формирующее идентификатор пользователя;
- программное обеспечение аутентификации, производящее сравнение отсканированного образца с имеющимися в базе данных.

Рассмотрим наиболее используемые системы биометрической аутентификации:

- при сканировании **отпечатков пальцев** с помощью оптического элемента формируется изображение, которое в дальнейшем сравнивается с эталоном. Биологическая повторяемость отпечатка пальца составляет  $10^{-5}\%$ , что дает самую высокую степень защиты;



- при сканировании **формы ладони** создается объемное изображение на основе измерений длины пальцев, толщины и площади поверхности, которое преобразуется в образец для дальнейших сравнений. Идентификация по геометрии руки применяется в законодательных органах, международных аэропортах, больницах, иммиграционных службах и т.д.;



- сканирование **узора радужной оболочки глаза** обладает наибольшей точностью. Сканеры могут отличать 2D-изображения от реального человеческого глаза, но эти устройства легко обойти;



## 10 класс | информатика

– работа **системы сканирования лица** заключается в построении трехмерной модели лица человека, которая строится после вычислений основных расстояний между частями лица. Наличие у человека бороды, усов, очков или шляпы не препятствует идентификации личности человека;





– при записи образца и в процессе последующего **распознавания по голосу** опираются на индивидуальные особенности голоса человека, такие как высота, модуляция и частота звука. Эти показатели уникальны и определяются физическими характеристиками голосового тракта.



Учетные записи пользователей в Windows бывают трех типов: **администратор, стандартная, гость**.

Рассмотрим процесс создания новой учетной записи пользователя компьютера на примере Windows 10, в других версиях все действия аналогичны.

Чтобы создать учетную запись пользователя или администратора, нажмите кнопку **Пуск** в Windows , выберите **Параметры**  – **Учетные записи**.

Далее следуйте указаниям службы поддержки **Microsoft Windows** (<https://www.microsoft.com/ru-kz/>).

## ЗАПОМНИ

Использование пароля повышает уровень безопасности компьютера. Когда на компьютере работают несколько пользователей, личные настройки, программы и системные ресурсы будут лучше защищены, если для учетной записи пользователя назначен пароль.



1. Что такое идентификация личности?

2. Что мы называем аутентификацией?



3. Назовите примеры методов биометрии. Какие наиболее используемые биометрические системы идентификации вы знаете?

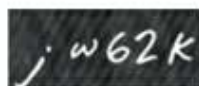


1. Используя прием анаграммы, герои сказок придумывали логины для системы идентификации пользователя в компьютерную систему: *doyssey, vergilul, omen, solas, sarlcon, oinkpkio*. Приведите их в соответствие с именами героев.

2. Дополните рассказ, используя слова: *идентификация, авторизация, аутентификация*.

Новый сотрудник впервые приходит на службу. Он представляется при входе и говорит, что теперь будет работать здесь менеджером. Таким образом, он ... себя – сообщает, кто он такой. Охранник требует представить доказательство того, что он действительно новый менеджер и имеет право входа в служебное помещение. Предъявление пропуска с фотографией и сличение его с имеющимся у охранника списком сотрудников, подтверждает ... . Допуск получен – состоялась ... .

3. Как называется компьютерный тест, используемый для определения пользователя системы: человека или компьютера?



4. Правило шифрования определяется следующим образом: двигаясь от начала и конца пароля к его середине, меняем местами каждую вторую пару символов. Например, пароль *password* – *prswsoad*.

Зашифруйте с помощью этого правила термины данного параграфа.



1. Каковы отличительные особенности аутентификации по физическим и поведенческим параметрам?

2. Какие преимущества имеют биометрические системы идентификации?



Решите кроссворд.

*По горизонтали:*

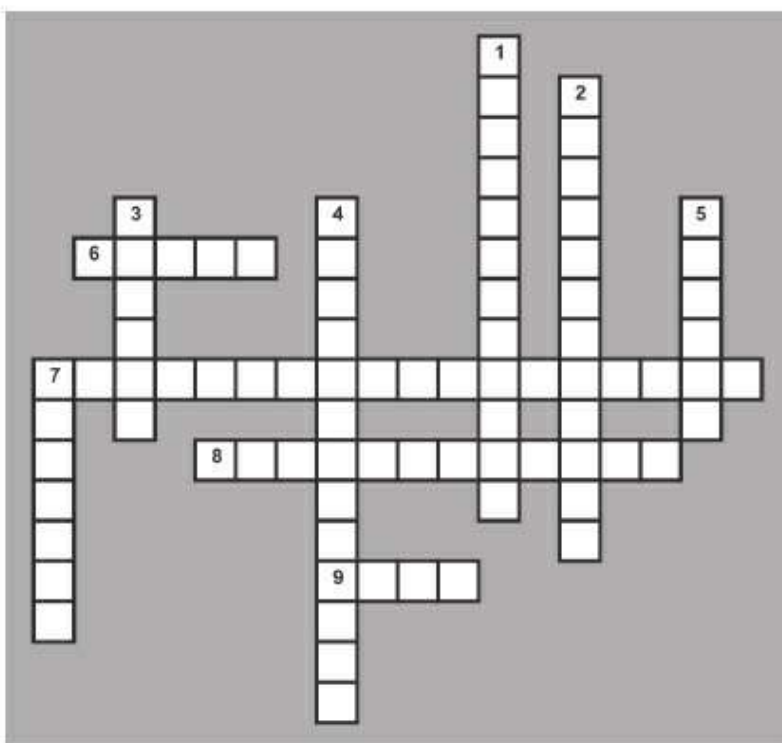


6. Вредоносная программа. 7. Необходимость предотвращения разглашения, утечки какой-либо информации. 8. Способность системы противостоять несанкционированному доступу к конфиденциальной информации, ее искажению или разрушению. 9. Вредоносная рассылка по электронной почте.



*По вертикали:*

1. Преследование с помощью сообщений, содержащих оскорбления, агрессию и запугивание.
2. Состояние защищенности интересов личности, общества, государства от внутренних и внешних угроз.
3. Вид интернет-мошенничества, цель которого – получение доступа к конфиденциальной информации пользователей – логинам и паролям.
4. Злоупотребление доверием пользователей Сети с преступными целями.
5. Гарантия безопасности или ряд действий и мероприятий по осуществлению этой гарантии.
7. Распространение заведомо ложных сведений, порочащих честь и достоинство другого лица или подрывающих его репутацию.





## ТЕСТОВЫЕ ЗАДАНИЯ

## «КОМПЬЮТЕРНЫЕ СЕТИ И ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ»



Знание



Понимание



Применение



Анализ



Синтез

**1. Установите соответствие между названием устройства и его определением.**

1)	Концентратор	<b>А)</b> Это устройство организывает прямое соединение между компьютером, запросившим данные, и компьютером, передающим запрошенную информацию
2)	Маршрутизатор	<b>В)</b> Это простые устройства, оборудованные необходимыми электронными компонентами для передачи сообщений между узлами в сети
3)	Коммутатор	<b>С)</b> Это устройство, которое предназначено для обеспечения доступа из локальной сети в глобальную

**2. Домен – это...**

- А)** часть адреса, определяющая адрес компьютера пользователя в сети;
- В)** название программы для осуществления связи между компьютерами;
- С)** название устройства, осуществляющего связь между компьютерами;
- Д)** единица скорости информационного обмена.

**3. Установите соответствие между названием устройства и его определением.**

1)	Сетевой протокол	<b>А)</b> Это фрагмент (блок данных) определенного формата для передачи по каналу связи
2)	Пакет (кадр) в телекоммуникации и компьютерных сетях	<b>В)</b> Набор правил и действий (очередности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включенными в сеть устройствами



4. Выберите корректный IP-адрес компьютера в сети.

- A) 108.214.198.112.
- B) 18.274.198.0.
- C) 1278.214.198.
- D) 10,0,0,1225.



5. Определите номер компьютера в сети по IP 215.128.255.106.

- A) 215.128.255.106.    B) 128.255.106.    C) 255.106.    D) 106.



6. Компьютер, подключенный к сети Интернет, обязательно имеет...

- A) URL-адрес;    B) IP-адрес;    C) web-страницу;    D) доменное имя.



7. Определите последовательность действий, чтобы проверить наличие сетевых адаптеров.

- 1) В диспетчере устройств нужно выбрать **Сетевые адаптеры**, а затем – имя сетевого адаптера.
- 2) Выбрать **Пуск – Панель управления**, а затем нужно выбрать пункт **Диспетчер устройств** из списка результатов. Если будет обнаружен новый драйвер, он загрузится и установится автоматически.
- 3) Если после установки обновленного драйвера потребуется перезапуск, нужно нажать кнопку **Пуск – Выключение – Перезапуск**, а затем проверить, помогло ли это решить проблему подключения.
- 4) После этого необходимо нажать и удерживать (или щелкнуть правой кнопкой мыши) имя сетевого адаптера, а затем выбрать **Обновить драйвер – Автоматический поиск обновленных драйверов**. Следуем указаниям, а затем выбираем **Заккрыть**.



8. Сформулируйте определение понятия «информационная безопасность», используя набор карточек.

случайных	– это защищенность	информации	информации
нанесение	от любых	может	которых
воздействий	быть	самой	результатом
ущерба	или злонамеренных	или ее владельцам	информационная безопасность



**9. Выберите косвенные признаки, которые могут свидетельствовать о заражении компьютерным вирусом.**

- |   |   |
|---|---|
| <p><b>A)</b> Экран мерцает.</p> <p><b>B)</b> Отсутствует доступ к Интернету.</p> <p><b>C)</b> Не работают мышь и клавиатура.</p> <p><b>D)</b> Залипают клавиши на клавиатуре.</p> <p><b>E)</b> Отсутствует движение любых объектов на экране компьютера.</p> <p><b>F)</b> На компьютере появляются неожиданные сообщения, изображения или звуковые сигналы.</p> <p><b>G)</b> Программы без вашего участия могут запускаться или подключаться к Интернету.</p> <p><b>H)</b> Другим на почту или через мессенджер приходят сообщения, которые вы не отправляли.</p> | <p><b>I)</b> В вашем почтовом ящике много сообщений без обратного адреса и заголовка.</p> <p><b>J)</b> Компьютер работает медленно или часто зависает.</p> <p><b>K)</b> При включении компьютера операционная система не загружается.</p> <p><b>L)</b> Файлы и папки могут исчезнуть или изменить содержимое.</p> <p><b>M)</b> Всплывает множество системных сообщений об ошибке.</p> <p><b>N)</b> Браузер зависает или ведет себя неожиданным образом. Например, вы не можете закрыть вкладку.</p> |
|---|---|



**10. Расположите категории классификации тайн в порядке возрастания конфиденциальности.**

- A)** Банковская тайна.
- B)** Служебная тайна.
- C)** Государственная тайна.
- D)** Персональные данные.
- E)** Профессиональная тайна.



## 2.1. Системы счисления

Какие системы счисления вы знаете?

Чем позиционная система счисления отличается от непозиционной?

– Что такое система счисления;  
– виды систем счисления;  
– как переводить целые числа из десятичной системы счисления в двоичную, восьмеричную и шестнадцатеричную системы счисления и обратно.

**Восьмеричная система** –  
Сегіздік жүйе – Octal system  
**Шестнадцатеричная система** –  
Он алтылық жүйе –  
Hexadecimal System  
**Десятичная система** –  
Ондық жүйе – Decimal system  
**Система счисления** –  
Санау жүйесі – Scale of notation  
**Двоичная система** –  
Екілік жүйе – Binary system

Древнее изображение десятичных цифр неслучайно: значение каждой цифры соответствует количеству углов в ее изображении. Например, 0 – углов нет, 1 – один угол, 2 – два угла и т.д.



вы узнаете

ключевые слова

это интересно

### ЗАПОМНИ

#### АЛФАВИТ СИСТЕМЫ СЧИСЛЕНИЯ

– это используемый набор цифр.

Помимо знакомой всем с детства **десятичной** (где используются десять цифр от **0** до **9**), в компьютерах широкое распространение нашли такие системы счисления, как двоичная (используются цифры **0** и **1**), восьмеричная и шестнадцатеричная системы счисления.

**Двоичная система счисления** – позиционная система счисления с основанием **2**. Ее алфавит содержит две цифры: **0** и **1**. Использование двоичной системы связано с тем, что запоминаящие элементы компьютера могут находиться только в двух устойчивых состояниях. Одно из этих состояний обозначается «**0**», а другое – «**1**». Следовательно, вся информация в компьютере должна быть представлена в виде комбинаций **0** и **1**.

### ЗАПОМНИ

Для обозначения различных систем счисления вводят нижний индекс. Он называется **ОСНОВАНИЕМ СИСТЕМЫ** счисления, в которой представлены числа. Пример обозначения двоичных, восьмеричных и шестнадцатеричных чисел:  $101000_2$ ;  $156_8$ ;  $AB_{16}$ .

**Восьмеричная система счисления** – позиционная система счисления

с основанием **8**. Для записи чисел в ней применяются цифры от **0** до **7**. Восьмеричная система часто используется в областях, связанных с цифровыми устройствами. Ранее она широко использовалась в программировании и компьютерной документации, однако в настоящее время почти полностью вытеснена шестнадцатеричной.

**Шестнадцатеричная система счисления** – позиционная система счисления с основанием **16**. Для представления чисел в ней используются десятичные цифры от **0** до **9** и буквы латинского алфавита – **A, B, C, D, E, F**.

**ПРИМЕЧАНИЕ:** A – 10; B – 11; C – 12; D – 13; E – 14; F – 15.

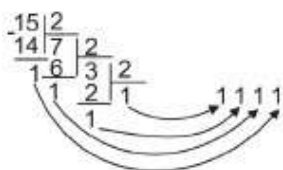


### ПРАВИЛА ПЕРЕВОДА ЦЕЛЫХ ЧИСЕЛ ИЗ ДЕСЯТИЧНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДВОИЧНУЮ, ВОСЬМЕРИЧНУЮ И ШЕСТНАДЦАТЕРИЧНУЮ

#### Перевод чисел из десятичной системы в двоичную

При преобразовании десятичного числа в двоичное его последовательно делят на **2** до тех пор, пока не останется остаток, меньший или равный **1**. В двоичной системе число состоит из последнего результата деления и остатков от деления, записанных в обратном порядке.

**ПРИМЕР 1.** Перевести число  $15_{10}$  в двоичную систему счисления.



Записав в обратном порядке остатки от деления, получим число  $15_{10} = 1111_2$ .

#### Перевод чисел из десятичной системы в восьмеричную и шестнадцатеричную

Процесс преобразования в восьмеричную и шестнадцатеричную системы счисления аналогичен переводу в двоичную систему. При изменении основания системы счисления меняется основание системы счисления, т.е. число, на которое проводится деление.

**ПРИМЕР 2.** Преобразовать число  $137_{10}$  в восьмеричную систему счисления. Последовательно делим число  $137_{10}$  на основание систе-



мы счисления **8** до тех пор, пока не останется остаток, меньший или равный 7.

$$\begin{array}{r|l} 137 & 8 \\ -136 & \\ \hline 1 & \\ \hline & 16 \\ - & \\ \hline & 2 \\ \hline & 1 \end{array}$$

Arrows indicate the sequence of remainders: 1, 2, 1, 1.

Записав в обратном порядке остатки от деления, получим число  $137_{10} = 211_8$ .

**ПРИМЕР 3.** Преобразовать число 558 в шестнадцатеричную систему счисления.

Последовательно делим число **558** на основание системы счисления **16** до тех пор, пока не останется остаток, меньший или равный 15.

$$\begin{array}{r|l} 558 & 16 \\ -544 & \\ \hline 14 & \\ \hline & 34 \\ - & \\ \hline & 2 \\ \hline & 32 \\ - & \\ \hline & 2 \end{array}$$

Arrows indicate the sequence of remainders: 2, 2, E.

Запишем в обратном порядке остатки от деления и получим число  $558_{10} = 22E_{16}$ .

### ПЕРЕВОД ИЗ ДВОИЧНОЙ, ВОСЬМЕРИЧНОЙ И ШЕСТНАДЦАТЕРИЧНОЙ СИСТЕМ СЧИСЛЕНИЯ В ДЕСЯТИЧНУЮ

Чтобы преобразовать число из произвольной системы счисления в десятичную, необходимо записать его в виде многочлена, составленного из произведений цифр числа и соответствующей степени основания. Затем нужно вычислить многочлен по правилам десятичной арифметики.

**ПРИМЕР 4.** Преобразовать число двоичной системы счисления  $1001_2$  в десятичную систему.

Для удобства пронумеруем разряды у исходного числа справа налево, начиная с 0. Номер разряда равен степени основания числовой системы.

$$\begin{array}{cccc} 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}_2$$

$$1001_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 0 + 0 + 1 = 9.$$

### ЗАПОМНИ

#### РАЗРЯД

– позиция цифры в числе. Разряды в записи целых чисел нумеруются от нуля справа налево.



## ПОЭКСПЕРИМЕНТИРУЙТЕ

Процесс преобразования из восьмеричной и шестнадцатеричной систем счисления в десятичную аналогичен преобразованию из двоичной системы в десятичную. Основываясь на вышеприведенном примере, попробуйте перевести из восьмеричной и шестнадцатеричной систем счисления в десятичную. Основанием степени теперь будет не **2**, а, соответственно, **8** и **16**.

$$211_8 = 137_{10}$$

$$22E_{16} = 558_{10}$$



1. Что такое система счисления?

2. Какие виды систем счисления существуют?



3. Какие способы перевода целых десятичных чисел в двоичную, восьмеричную и шестнадцатеричную системы счисления и обратно вы знаете?



1. Переведите данные числа из десятичной системы счисления.

1)  $263_{10} \rightarrow X_8$

6)  $1323_{10} \rightarrow X_{16}$

2)  $92_{10} \rightarrow X_2$

7)  $2945_{10} \rightarrow X_{16}$

3)  $614_{10} \rightarrow X_8$

8)  $781_{10} \rightarrow X_8$

4)  $1201_{10} \rightarrow X_{16}$

9)  $111_{10} \rightarrow X_8$

5)  $901_{10} \rightarrow X_8$

10)  $913_{10} \rightarrow X_{16}$

2. Переведите число в десятичную систему счисления (табл. 2.1.1).

ТАБЛИЦА 2.1.1

Вариант	Задание	Вариант	Задание	Вариант	Задание
1	$10001110_2$	5	$11101110_2$	9	$1111100_2$
2	$10011010_2$	6	$1000001_2$	10	$101001110_2$
3	$1001110_2$	7	$10111011_2$	11	$10011000_2$
4	$11111110_2$	8	$10010101_2$	12	$10111110_2$



**3.** Выполните задания.

**1)** Выберите число, которое является минимальным среди следующих чисел:

$1000000_2$ ,  $10011_2$ ,  $39_{10}$ ,  $52_{10}$ .

**2)** Расположите числа в порядке возрастания:

$110010_2$ ,  $73_{10}$ ,  $40_{10}$ ,  $101010_2$ .

**3)** Выберите число, которое является максимальным среди следующих чисел:

$100001_2$ ,  $1001110_2$ ,  $42_{10}$ ,  $63_{10}$ .

**4)** Расположите числа в порядке убывания:

$101001_2$ ,  $43_{10}$ ,  $100010_2$ ,  $52_{10}$ .

**5)** Выберите число, которое является минимальным среди следующих чисел:

$100110_2$ ,  $11110_2$ ,  $56_{10}$ ,  $23_{10}$ ,  $521_8$ .

**6)** Расположите числа в порядке убывания:

$110111_2$ ,  $76_{10}$ ,  $101_2$ ,  $54_{10}$ .

**7)** Выберите число, которое является максимальным среди следующих чисел:

$11001_2$ ,  $1111_2$ ,  $24_{10}$ ,  $240_{10}$ ,  $7B_{16}$ .

**8)** Выберите число, которое является минимальным среди следующих чисел:

$11001_2$ ,  $100_2$ ,  $38_{10}$ ,  $23_{10}$ ,  $1F_{16}$ .

**9)** Расположите числа в порядке убывания:

$110010_2$ ,  $10010110_2$ ,  $200_{10}$ ,  $74_{10}$ ,  $2473_8$ .

**10)** Расположите числа в порядке возрастания:

$100010_2$ ,  $100111_2$ ,  $61_{10}$ ,  $32_{10}$ ,  $37_8$ .



Напишите эссе на тему «Система счисления и компьютер».

## 2.2. Перевод чисел из одной системы счисления в другую

Какие системы счисления существуют?

Как переводить числа из одной системы счисления в другую.

**Восьмеричная система** –  
Сегіздік жүйе – Octal system

**Шестнадцатеричная система** –

Он алтылық жүйе –  
Hexadecimal system

**Десятичная система** –

Ондық жүйе –  
Decimal system

В те времена, когда наши предки только начинали считать, они применяли систему из 20 символов, используя для этого пальцы на руках и ногах. Позже эта тенденция изменилась. Десятичная система счисления в настоящее время наиболее известная и используемая.

вы узнаете

ключевые слова

это интересно

### ПЕРЕВОД ЧИСЕЛ ИЗ ДВОИЧНОЙ В ВОСЬМЕРИЧНУЮ СИСТЕМУ СЧИСЛЕНИЯ

1. Сгруппируем по три цифры (триады) исходного двоичного числа справа налево, начиная с разряда единиц. Если количество цифр исходного двоичного числа не кратно 3, оно дополняется слева нулями до 3.

2. Заменяем каждую триаду восьмеричной цифрой в соответствии с таблицей 2.2.1.

В восьмеричной системе каждой цифре соответствует набор из трех цифр в двоичной системе счисления (табл. 2.2.1):

ТАБЛИЦА 2.2.1

$N_8$	0	1	2	3	4	5	6	7
$N_2$	000	001	010	011	100	101	110	111

**ПРИМЕР 1.** Перевести число  $1111011100_2$  в восьмеричную систему счисления ( $2 \Rightarrow 8$ ).



1 <sub>8</sub>	7 <sub>8</sub>	3 <sub>8</sub>	4 <sub>8</sub>
001	111	011	100

**ОТВЕТ:**  $1111011100_2 = 1734_8$ .

\*Зеленые цифры – добавленные нули

### ПЕРЕВОД ИЗ ВОСЬМЕРИЧНОЙ В ДВОИЧНУЮ СИСТЕМУ СЧИСЛЕНИЯ

Для преобразования восьмеричного числа в двоичную систему счисления каждая цифра восьмеричного числа заменяется триадой двоичных цифр в соответствии с таблицей. Если в таблице двоичное число имеет менее 3 цифр, оно дополняется слева нулями до 3 для целых чисел.

#### ПРИМЕР 2.

Перевести число  $1734_8$  в двоичную систему счисления (**8** => **2**).

001	111	011	100
1 <sub>8</sub>	7 <sub>8</sub>	3 <sub>8</sub>	4 <sub>8</sub>

**ОТВЕТ:**  $1734_8 = 1111011100_2$ .

### ПЕРЕВОД ЧИСЕЛ ИЗ ДВОИЧНОЙ В ШЕСТНАДЦАТЕРИЧНУЮ СИСТЕМУ СЧИСЛЕНИЯ

Разобьем двоичное число на группы по четыре цифры (тетрады), начиная с разряда единиц (самой правой цифры). Последняя (самая левая) тетрада может оказаться неполной. В этом случае она дополняется нулями (одним, двумя или тремя). Затем тетрады заменяются на соответствующие (по таблице тетрад) цифры шестнадцатеричной системы счисления.

Обратите внимание, что каждая тетрада – это запись соответствующей цифры в двоичной системе счисления.

В шестнадцатеричной системе каждой цифре соответствует набор из четырех цифр в двоичной системе счисления (табл. 2.2.2).

ТАБЛИЦА 2.2.2

$N_{16}$	0	1	2	3	4	5	6	7
$N_2$	0000	0001	0010	0011	0100	0101	0110	0111
$N_{16}$	8	9	A	B	C	D	E	F
$N_2$	1000	1001	1010	1011	1100	1101	1110	1111

**ПРИМЕР 3.**

Перевести число  $10111101101_2$  в шестнадцатеричную систему счисления ( $2 \Rightarrow 16$ ).

**ОТВЕТ:**  $10111101101_2 = 0101\ 1110\ 1101_2 = 5ED_{16}$ .

$5_{16}$	$E_{16}$	$D_{16}$
0101	1110	1101

### ПЕРЕВОД ИЗ ШЕСТНАДЦАТЕРИЧНОЙ В ДВОИЧНУЮ СИСТЕМУ СЧИСЛЕНИЯ

Для преобразования шестнадцатеричного числа в двоичную систему достаточно каждую цифру шестнадцатеричного числа заменить соответствующей ей двоичной тетрадой из таблицы 2.2.2.

**ПРИМЕР 4.** Перевести  $A78_{16}$  в двоичную систему счисления ( $16 \Rightarrow 2$ ).

**ОТВЕТ:**  $A78_{16} = 1010\ 0111\ 1000_2$ .

1010	0111	1000
$A_{16}$	$7_{16}$	$8_{16}$



1. Дайте характеристику двоичной, восьмеричной и шестнадцатеричной системам счисления: алфавит, основание, запись чисел. Приведите примеры записи чисел.

2. Как переводить целые числа из двоичного представления в восьмеричное и шестнадцатеричное представления и обратно?



1. Переведите данные числа.



1)  $714_8 = ?_2$

3)  $7_8 = ?_{10}$

2)  $2011_8 = ?_2$

4)  $117_8 = ?_{10}$



5)  $55_8 = ?_{10}$

6)  $001_2 = ?_8$

7)  $10010100_2 = ?_8$

8)  $110100_2 = ?_8$

9)  $537_{10} = ?_8$

10)  $1056_{10} = ?_8$

11)  $739_{10} = ?_8$

12)  $A63D_{16} = ?_2$

13)  $5FFF_{16} = ?_2$

14)  $7A5D_{16} = ?_2$

15)  $110110_2 = ?_{16}$

16)  $1111_2 = ?_{16}$

17)  $355_{10} = ?_{16}$

18)  $912_{10} = ?_{16}$

19)  $FFAD_{16} = ?_{10}$

20)  $76AC_{16} = ?_{10}$

2. Сравните.

1)  $123_8$  и  $1011100_2$

2)  $52_{16}$  и  $51_8$

3)  $367_8$  и  $1152_{10}$

4)  $1245_{10}$  и  $101011_2$

5)  $218_{10}$  и  $179_{16}$

6)  $100101110_2$  и  $7FF_{16}$

3. Заполните таблицу 2.2.3, в каждой строке которой одно и то же целое число должно быть записано в различных системах счисления.

ТАБЛИЦА 2.2.3

Двоичная	Восьмеричная	Десятичная	Шестнадцатеричная
1010110			
	427		
		275	
			F1B

4. В  $10^A$  и  $10^B$  классах девочек  $1111_2$ . Сколько мальчиков в классах, если всего  $20_{16}$  человек?

**Примечание:** проверьте результаты выполненных заданий с помощью онлайн-калькулятора (<https://calculatori.ru/perevod-chisel.html>).



Подготовьте реферат на тему «Системы счисления».



## 2.3. Логические операции. Построение таблиц истинности

Что такое высказывание?

Какое высказывание считается истинным,  
а какое – ложным?

– Как использовать логические операции (дизъюнкция, конъюнкция, инверсия);  
– как построить таблицу истинности для заданного логического выражения.

**Дизъюнкция** – Дизъюнкция – Disjunction

**Конъюнкция** – Конъюнкция – Conjunction

**Инверсия** – Инверсия – Inversion

**Таблица истинности** – Ақиқат кестесі – Truth table

Известный древнегреческий ученый Аристотель (384–322 г. до н.э.) заложил основы формальной логики. Он отделил форму мышления от содержания, применил математические методы к логике.

вы узнаете

ключевые слова

это интересно

**Высказывание (суждение)** – некоторое предложение, которое может быть истинно (верно) или ложно.

**Логическое выражение** – устное утверждение или запись, в которое наряду с постоянными величинами обязательно входят **переменные величины (объекты)**. Логическое выражение может принимать одно из двух возможных значений: **истина** (логическая 1) или **ложь** (логический 0).

**Сложное логическое выражение** – логическое выражение, состоящее из одного или нескольких простых логических выражений (или сложных логических выражений), соединенных с помощью логических операций.

Существуют три основные логические операции, с помощью которых можно записать любое логическое выражение:

1. Инверсия.

### ЗАПОМНИ

#### ЛОГИЧЕСКИЕ ОПЕРАЦИИ

– мыслительные действия, результатом которых является изменение содержания или объема понятий, а также образование новых понятий.

### ЗАПОМНИ

Логическая операция, ставящая в соответствие двум высказываниям новое, являющееся истинным тогда и только тогда, когда оба исходных высказывания истинны, называется **КОНЪЮНКЦИЕЙ**.

**ЗАПОМНИ**

Логическая операция, ставящая в соответствие двум высказываниям новое, являющееся ложным тогда и только тогда, когда оба исходных высказывания ложны, называется **ДИЗЬЮНКЦИЕЙ**.

2. Конъюнкция.

3. Дизъюнкция.

**Конъюнкция**, или «**Логическое И**», или «**Логическое умножение**». Допустим, у нас есть два простых выражения – **A** и **B**. Они могут быть равными или **1 (истина)**, или **0 (ложь)**. Применяя к ним операцию «Логическое И», мы получим сложное выражение, которое примет значение **1 (истина)** исключительно в том случае, если и **A**, и **B** имеют значение **1 (истина)**, во всех других случаях

результат будет **0 (ложь)**.

Операция «Логическое И» имеет обозначения **И, &&, AND, &, ∧, \***.

Таблица истинности для конъюнкции (табл. 2.3.1):

**ТАБЛИЦА 2.3.1**

<b>A</b>	<b>B</b>	<b>F = A ∧ B</b>
1	1	1
1	0	0
0	1	0
0	0	0

**Дизъюнкция**, или «**Логическое ИЛИ**», или «**Логическое сложение**». В «Логическом И» результат равен **1**, если и **A**, и **B** равны **1**, а в «Логическом ИЛИ» результат будет равен **1**, если **или A, или B** равны единице. Операция «Логическое ИЛИ» имеет обозначения: **ИЛИ, ||, OR, ∨, +**.

Таблица истинности для дизъюнкции (табл. 2.3.2):

**ТАБЛИЦА 2.3.2**

<b>A</b>	<b>B</b>	<b>F = A ∨ B</b>
1	1	1
1	0	1
0	1	1
0	0	0

**Инверсия**, или «**Логическое НЕ**», или «**Отрицание**». Операция «Логическое НЕ» имеет обозначения **НЕ**, **!**, **NOT**, **¬**, **–**. Если **A = 1** (истина), то после выполнения операции «Отрицание» **A** примет значение **0**, то есть становится ложным.

Таблица истинности для инверсии (табл. 2.3.3):

ТАБЛИЦА 2.3.3

A	A
1	0
0	1



#### Алгоритм построения таблицы истинности:

- 1) определить количество переменных **n** в логическом выражении;
- 2) вычислить количество строк в таблице по формуле  $m = 2^n$ , где **n** – количество переменных;
- 3) подсчитать количество логических операций в формуле;
- 4) определить последовательность выполнения логических операций с учетом приоритетов операций и скобок (табл. 2.3.4):

ТАБЛИЦА 2.3.4

Приоритет	Операция		Обозначение
I (Высший)	НЕ	NOT	¬, –
II (Высокий)	И	AND	∧, *
III (Средний)	ИЛИ, Искл. ИЛИ	OR, XOR	∨, + ⊕

- 5) определить количество столбцов: число переменных + число операций;

### ЗАПОМНИ

Логическая операция, которая каждому высказыванию ставит в соответствие новое высказывание, значение которого противоположно исходному, называется **ОТРИЦАНИЕМ**.

### ЗАПОМНИ

#### ТАБЛИЦА ИСТИННОСТИ

– таблица, отображающая значения составного логического выражения при всех возможных значениях, входящих в него простых высказываний.



- 6) выписать наборы входных переменных;  
7) спроектировать таблицу истинности.

### Построение таблицы истинности

1. Разбить столбец со значениями первой переменной на две части и заполнить верхнюю часть нулями, а нижнюю – единицами.
2. Разбить столбец со значениями второй переменной на четыре части и заполнить каждую часть чередующимися группами нулей и единиц, начиная с группы нулей.
3. Выполнять разделение столбцов значений последующих переменных на **8**, **16** и т.д. частей и заполнять их группами нулей или единиц до тех пор, пока группы не будут состоять из одного элемента.
4. Провести заполнение таблицы истинности по столбцам, выполняя логические операции с учетом их приоритетов.



**ПРИМЕР 1.** Для формулы  $A \wedge (B \vee \neg B \wedge \neg A)$  постройте таблицу истинности.

### Алгоритм построения таблицы истинности для формулы $A \wedge (B \vee \neg B \wedge \neg A)$

1. Количество логических переменных  $n = 2$ , так как в нашем примере  $A, B$ .
2. Количество строк:  $2^2 = 4$ .
3. Количество логических операций в формуле  $5 (\wedge, \vee, \neg, \wedge, \neg)$ .
4.  $A \wedge (B \vee \neg B \wedge \neg A)$
5. Количество столбцов равно  $2 + 5 = 7$ .
6. Наборы входных переменных.

A	B
0	0
0	1
1	0
1	1



**7. Заполнение таблицы истинности.**

A	B	$\neg B$	$\neg A$	$(\neg B) \wedge (\neg A)$	$B \vee ((\neg B) \wedge (\neg A))$	$A \wedge (B \vee ((\neg B) \wedge (\neg A)))$
0	0	1	1	1	1	0
0	1	0	1	0	1	0
1	0	1	0	0	0	0
1	1	0	0	0	1	1

**ПРИМЕР 2.** Используя онлайн-калькулятор (<https://math.semestr.ru/inf/table.php>), определите истинность логического выражения  $F(A, B, C) = \neg(\neg A \wedge C) \vee (\neg B \vee A)$  (рис. 2.3.1).

Логическое выражение:

x+y&z

- Вывод промежуточных таблиц для таблицы истинности
- Построение СКНФ
- Построение СДНФ
- Построение полинома Жегалкина
- Построение карты Вейча-Карно
- Минимизация булевой функции

Решить

Например, логическое выражение  $abc+ab^{\neg}c+a^{\neg}bc$  необходимо ввести так: `a*b*c+a*b~c+a~b*c`

Для ввода данных в виде логической схемы используйте этот [сервис](#).

**Рис. 2.3.1.** Вид онлайн-калькулятора



1. Назовите основные логические операции.
2. Что такое логическое высказывание? Отрицание? Конъюнкция? Дизъюнкция?
3. Почему операция И называется логическим умножением, а ИЛИ – логическим сложением?
4. Назовите основные логические операции и приведите их таблицы истинности.



5. Запишите алгоритм построения таблицы истинности.
6. Запишите следующее высказывание в виде логических выражений: «Я поеду к бабушке и, если встречу там друзей, то интересно проведу время».
7. Запишите приоритет выполнения логических операций.



1. Заполните таблицу 2.3.5 (Истина – 1. Ложь – 0.)

ТАБЛИЦА 2.3.5

Высказывание	Равносторонний треугольник	Прямоугольный треугольник	Равнобедренный треугольник
Все его стороны равны			
Две его стороны равны			
Один из его углов прямой			
Площадь равна половине произведения катетов и вычисляется по формуле: $S = \frac{1}{2}ab$			
Квадрат гипотенузы равен сумме квадратов катетов			
Углы при основании равны			
Медиана, проведенная к основанию, является одновременно и биссектрисой, и высотой этого треугольника			
Площадь со стороной $a$ равна $S = \frac{a^2\sqrt{3}}{4}$			

2. Запишите высказывание «Если яблоко зеленое или мелкое, то оно твердое», используя знаки логических операций.

3. Определите результат логического выражения при заданных параметрах:

$$\neg(\neg B \& \neg C) \& \neg(\neg A \& \neg C), \text{ при } A = 1, B = 1, C = 0.$$



4. Постройте таблицу истинности логической функции.

A)  $F = (A \vee B) \wedge (\neg A \vee \neg B)$ .

B)  $F = \neg A \wedge \neg B \wedge \neg C$ .



1. Символом F обозначено одно из указанных ниже логических выражений от трех аргументов: X, Y, Z. Дан фрагмент таблицы истинности выражения F.

Какое выражение соответствует F (табл. 2.4.5)?

A)  $\bar{X} \& Y \& \bar{Z}$ .

B)  $X \vee \bar{Y} \vee Z$ .

C)  $X \& \bar{Y} \& \bar{Z}$ .

D)  $\bar{X} \vee Y \vee \bar{Z}$ .

ТАБЛИЦА 2.4.5

X	Y	Z	F
0	1	0	1
1	0	1	0
1	1	0	0

2. Какое выражение соответствует F (табл. 2.4.6)?

A)  $X \vee Y \vee Z$ .

B)  $X \& Y \& \bar{Z}$ .

C)  $\bar{X} \& Y \& \bar{Z}$ .

D)  $\bar{X} \vee Y \vee \bar{Z}$ .

ТАБЛИЦА 2.4.6

X	Y	Z	F
0	0	0	0
1	1	0	1
1	1	1	0



Придумайте высказывания для следующих логических выражений.

1.  $\bar{A} \vee \bar{B}$ .    2.  $\bar{A} \& \bar{B} \vee \bar{C}$ .    3.  $A \& B$ .    4.  $\bar{A} \& \bar{B}$ .

## 2.4. Логические элементы компьютера

Какие логические операции вы знаете?

- Назначение основных логических элементов: конъюнктор, дизъюнктор, инвертор;
- как преобразовать логические выражения в логические схемы и наоборот.

**Логические элементы** – Логикалық элементтер – Logical elements

**Логическая схема** – Логикалық схема – Logic circuit

вы узнаете

ключевые слова

### ЗАПОМНИ

#### ЛОГИЧЕСКИЙ ЭЛЕМЕНТ (ВЕНТИЛЬ)

– простейшая структурная единица компьютера, выполняющая определенную логическую операцию над двоичными переменными.

На базе логических элементов основаны абсолютно все вычислительные устройства современной цифровой техники (компьютеры, мобильные устройства).

Базовые логические элементы компьютера реализуют три основные логические операции:

- конъюнктор (логический элемент «И») выполняет логическое умножение;
- дизъюнктор (логический элемент «ИЛИ») выполняет логическое сложение;
- инвертор (логический элемент «НЕ») выполняет операцию отрицания.

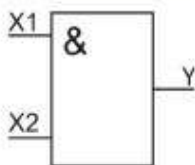


Рис. 2.4.1. Логический элемент «И»

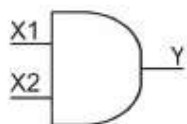


Рис. 2.4.2. Элемент AND

Схема **конъюнктора** имеет не менее двух входов и один выход. Сигнал «1» появляется на выходе в том случае, когда на все входы одновременно подаются входные сигналы «1».

На цифровых схемах логический элемент «И» обозначают, как показано на *рис. 2.4.1*.

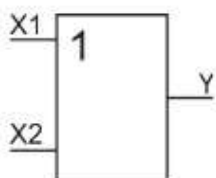
На иностранных схемах элемент «И» обозначается иначе (см. *рис. 2.4.2*) и называется **AND**.



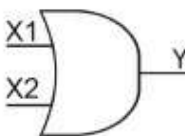
Схема **дизъюнктора** также имеет не менее двух входов и один выход. Сигнал «1» появляется на выходе, если хотя бы на один вход подается сигнал «1».

На цифровых схемах логический элемент «ИЛИ» изображается, как показано на *рис. 2.4.3*.

На иностранных схемах его изображают по-другому и называют элементом **OR** (*рис. 2.4.4*).



**Рис. 2.4.3.** Логический элемент «ИЛИ»

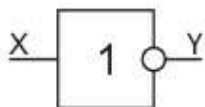


**Рис. 2.4.4.** Элемент **OR**

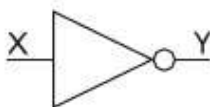
Схема **инвертора** имеет один вход и один выход. На выходе образуется сигнал «1», если на входе будет сигнал «0».

На цифровых схемах логический элемент «НЕ» изображается, как показано на *рис. 2.4.5*.

На иностранных схемах элемент «НЕ» обозначается, как показано на *рис. 2.4.6*. Сокращенно его называют **NOT**.



**Рис. 2.4.5.** Логический элемент «НЕ»



**Рис. 2.4.6.** Элемент **NOT**

### ДЛЯ ЧЕГО СТРОЯТСЯ ЛОГИЧЕСКИЕ СХЕМЫ?

Из логических элементов создают более сложные цифровые схемы, предназначенные для вычисления арифметических операций и хранения информации. Схему, способную выполнять заданные функции, можно составить из различных по сочетанию и количеству логических элементов.

### Правила построения логических схем

- 1) Определить количество логических переменных.
- 2) Определить число базовых логических операций и последовательность их выполнения.
- 3) Изобразить для каждой логической операции соответствующий ей логический элемент.
- 4) Соединить логические элементы между собой в порядке выполнения логических операций.

### ЗАПОМНИ

#### ЛОГИЧЕСКАЯ СХЕМА

– это электронное устройство, которое реализует любую логическую функцию, описывающую работу устройств компьютера.



Построим логическую схему, соответствующую логическому выражению  $F = A \wedge B \vee \neg(B \vee A)$ .

Вычислим значения выражения для  $A = 1, B = 0$ .

- 1) Переменных две:  $A$  и  $B$ .
- 2) Логических операций четыре: конъюнкция, две дизъюнкции и отрицание. Определяем порядок выполнения операций:

$$A \wedge B \vee \neg(B \vee A)$$

- 3) Схему строим слева направо в соответствии с порядком выполнения логических операций (рис. 2.4.7):

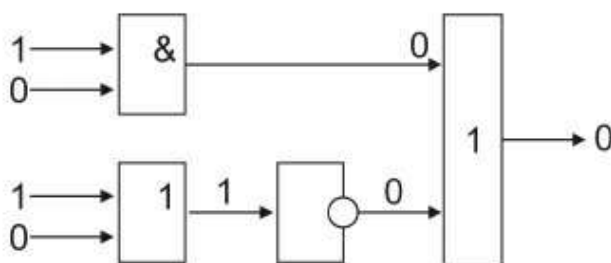


Рис. 2.4.7. Построение схемы

- 4) Вычислим значение выражения:  $F = 1 \wedge 0 \vee \neg(0 \vee 1) = 0$ .
- 5) Проверим правильность построения логических схем по ссылке <https://www.semestr.online>. Для начала посмотрим видеоинструкцию.



1. Что такое логический элемент (вентиль)?
2. Назовите базовые логические элементы и приведите их обозначения на схемах.
3. Какую функцию выполняет элемент «конъюнктор»?
4. Какую функцию выполняет элемент «дизъюнктор»?
5. Какую функцию выполняет элемент «инвертор»?
6. Для чего строятся логические схемы?
7. Составьте алгоритм построения логических схем.



Приведите примеры построения логических выражений по заданным логическим схемам.



1. Постройте соответствующую выражению логическую схему.

- 1)  $F = A \vee B \wedge (B \vee A)$ , если  $A = 1$ ,  $B = 0$ .
- 2)  $F = \neg (A \wedge B) \wedge (B \vee A)$ , если  $A = 0$ ,  $B = 1$ .
- 3)  $F = \neg A \vee B \wedge B$ , если  $A = 1$ ,  $B = 0$ .
- 4)  $F = A \wedge B \vee (C \vee B)$ .  $A = 1$ ,  $B = 0$ ,  $C = 1$ .
- 5)  $F = \neg A \vee C \wedge (A \vee B)$ .  $A = 1$ ,  $B = 1$ ,  $C = 0$ .



2. Определите структурную формулу по заданной схеме (рис. 2.4.8).

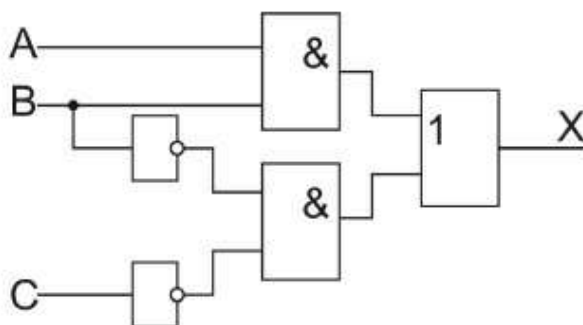


Рис. 2.4.8. Схема 1



3. Найдите значение выходного сигнала в приведенной схеме (рис. 2.4.9), если:

- 1)  $A = 0$  и  $B = 0$ ;
- 2)  $A = 0$  и  $B = 1$ ;
- 3)  $A = 1$  и  $B = 0$ ;
- 4)  $A = 1$  и  $B = 1$ .

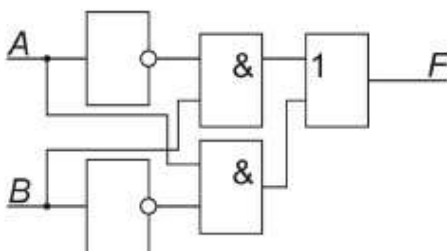


Рис. 2.4.9. Схема 2



Найдите ошибку.

Постройте логическую схему, соответствующую логическому выражению  $F = B \wedge A \vee \neg(B \vee A)$ . Вычислите значение выражения для  $A = 1$ ,  $B = 0$  (рис. 2.4.10).

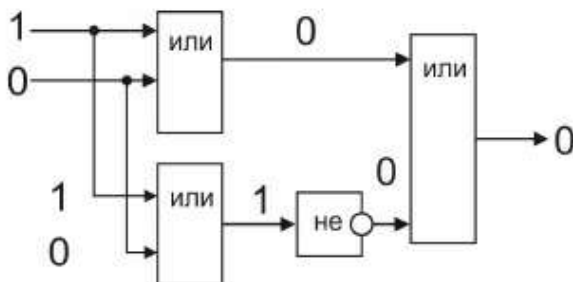


Рис. 2.4.10. Схема 3



## 2.5. Логические основы компьютера

Какие логические элементы вы знаете?

Как описывать функции устройства управления, арифметико-логического устройства и регистров памяти как отдельных частей процессора.

**Устройства управления** – Басқару құрылғысы – Control Unit (CU)

**Арифметико-логическое устройство** – Арифметика-логикалық құрылғы – Arithmetic and Logic Unit (ALU)

**Регистр памяти** – Жады регистры – Register-memory

Разработчик компьютера ENIAC Джон фон Нейман был первым создателем АЛУ. В 1945 году он опубликовал первые научные работы по новому компьютеру, названному EDVAC (Electronic Discrete Variable Automatic Computer). Архитектура этого компьютера позже стала прототипом архитектур большинства последующих компьютеров. В своих работах фон Нейман указывал устройства, которые, как он считал, должны присутствовать в компьютерах. Среди этих устройств присутствовало и АЛУ.

вы узнаете

ключевые слова

это интересно

Компьютер содержит три главных компонента: процессор, оперативное запоминающее устройство (ОЗУ), устройства ввода и вывода информации.



Рис. 2.5.1. Общая схема компьютера

Информация, с которой работает компьютер, делится на два принципиально разных типа:

- набор команд по обработке данных (программа);
- данные, которые обрабатывает программа.

Руководит обработкой процессор, который, в свою очередь, состоит из двух блоков: **устройства управления** и **арифметико-логического устройства** (рис. 2.5.1).

### КАКУЮ ФУНКЦИЮ ВЫПОЛНЯЕТ АРИФМЕТИКО-ЛОГИЧЕСКОЕ УСТРОЙСТВО?

#### ЗАПОМНИ

##### АРИФМЕТИКО-ЛОГИЧЕСКОЕ УСТРОЙСТВО (АЛУ)

– основная часть процессора, предназначенная для выполнения арифметических и логических операций.

Арифметико-логическое устройство (АЛУ) объединяет различные математические и логические операции в единое устройство. Например, типичное АЛУ может выполнять сложение, вычитание, сравнение величин, операции И и ИЛИ. АЛУ формирует «сердце» большинства компьютерных систем.

На рисунке 2.5.2 показана схема N-битного АЛУ с N-битными входами и выходами. АЛУ получает управляющий сигнал F, который указывает, какую функцию выполнять. Сигналы управления обычно отображаются синим цветом, чтобы отличить их от данных. В таблице 2.5.1 перечислены типичные функции, которые АЛУ может выполнять. Функция SLT (*Set Less Than* – установи меньше, чем) используется для сравнения величин.

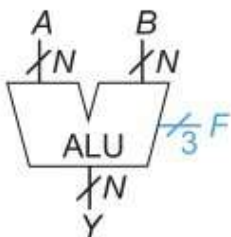


Рис. 2.5.2. АЛУ

ТАБЛИЦА 2.5.1

#### Операции АЛУ

$F_{2:0}$	Функция
000	A AND B
001	A OR B
010	A + B
011	не используется
100	A AND $\bar{B}$
101	A OR $\bar{B}$
110	A – B
111	SLT

## КАКУЮ ФУНКЦИЮ ВЫПОЛНЯЕТ УСТРОЙСТВО УПРАВЛЕНИЯ?

Устройство управления (УУ) отвечает за управление различными частями компьютера. Управляющее устройство организует работу других устройств и анализирует информацию об их состоянии.

Устройство управления имеет специальный регистр (ячейку), который называют **счетчиком команд**. После загрузки программы и данных в память в счетчике команд сохраняется адрес первой программной команды. УУ считывает из памяти содержимое ячейки, адрес которой находится в счетчике команд, и записывает его в специальное устройство – регистр команд. УУ определяет очередную операцию, которую надо выполнить, находит в памяти данные, адреса которых указаны в команде, и отслеживает процесс ее выполнения. Операцию вычисляет АЛУ, или аппаратные средства компьютера. После выполнения текущей операции счетчик команд наращивается на единицу и указывает на следующую команду программы. Если есть необходимость выполнения команд не последовательно, а в заданном порядке, то специальная команда перехода содержит адрес ячейки, куда требуется передать управление.

### ЗАПОМНИ

#### УСТРОЙСТВО УПРАВЛЕНИЯ (УУ)

– блок процессора, который выполняет управление компонентами процессора: блоком памяти, АЛУ, устройствами ввода и вывода.

## КАКУЮ ФУНКЦИЮ ВЫПОЛНЯЕТ РЕГИСТР ПАМЯТИ?

Регистры – это устройства, предназначенные для записи, хранения, выдачи и преобразования информации, представленной в виде двоичных кодов.

Области применения: устройства памяти, элементы задержки, преобразователи последовательных кодов в параллельные и наоборот, кольцевые распределители сигналов и т.д. В зависимости от функциональных свойств и схемной реализации они подразделяются на:

- регистры памяти;

### ЗАПОМНИ

#### РЕГИСТР ПАМЯТИ

– устройство, предназначенное для получения и хранения информации, представленной двоичными кодами.



## 10 класс | информатика

- регистры сдвига;
- универсальные регистры.

Назначение регистров памяти – сохранять двоичный код на протяжении промежутка времени.



1. Назовите основные компоненты компьютера. Каково их назначение?



2. Какую функцию выполняет АЛУ? УУ? Регистры?

3. Найдите в Интернете информацию о регистрах процессора Intel. Какие регистры являются наиболее важными?

4. Что такое сумматор?



Составьте кроссворд на тему «Логические основы компьютера».



Составьте мультимедийную презентацию на тему «Логические основы компьютера».

## 2.6. Принципы кодирования текстовой информации

Какие виды информации существуют?

Как сравнивать таблицы кодировки символов Unicode и ASCII.

**Юникод** – Юникод – Unicode

**Кодирование** – Кодтау – Coding

**Декодирование** – Декодтау – Decoding

Если к кириллическому тексту в кодировке Windows-1251 20 раз подряд применить перекодирование KOI8-R → Windows-1251, то в итоге будет получен исходный текст.

вы узнаете

ключевые слова

это интересно



Компьютер – универсальное устройство для обработки информации, представленной в числовой форме. Любая информация (звуковая, визуальная, текстовая и т.д.) для обработки на компьютере должна быть преобразована в числовую форму. Мы знаем, что все числа в компьютере представляются с помощью нулей и единиц.

Существует таблица кодов **ASCII** (*American Standard Code for Information Interchange*) – американский стандартный код для обмена информацией, содержащий 128 символов, в частности буквы латинского алфавита в виде чисел. Для кодирования букв других алфавитов эту таблицу дополняют до 256 символов. Таблица ASCII разделена на **стандартную** (коды от 0 до 127) и **расширенную** части (символы с кодом от 128 до 255). В набор были включены непечатаемые символы, используемые как команды для управления устройствами ввода/вывода (символы с кодами 0–31), символы латинского алфавита, знаки препинания, цифры, арифметические действия и некоторые дополнительные символы (символы с кодами 32–127).

Следующим этапом в развитии кодировок явилось появление **ANSI-кодировок**. Это **расширенные версии ASCII**, но из них были удалены различные дополнительные элементы и добавлены типографские символы. Всем известная Windows-1251 является примером ANSI-кодировки. Кроме типографских символов, в эту кодировку также были включены буквы алфавитов языков, использующих кириллицу. В 2002 году был принят стандарт официальной кодировки казахского языка – **KZ-1048**, совместимой с Windows-1251, фрагмент которой представлен на рисунке 2.6.7.

## ЗАПОМНИ

### КОД

– это набор условных обозначений (или сигналов) для записи (или передачи) некоторых заранее определенных понятий.

### КОДИРОВАНИЕ ИНФОРМАЦИИ

– это процесс формирования определенного представления информации. Другими словами, под «кодированием» часто понимают переход от одной формы представления информации к другой, более удобной для хранения, передачи или обработки.

## ЗАПОМНИ

### ДЕКОДИРОВАНИЕ

– преобразование данных из двоичного кода в форму, понятную человеку.

Множество символов, используемых при записи текста, называется **алфавитом**. Количество символов в алфавите называется его **мощностью**.

# 10 класс | информатика

	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.A	.B	.C	.D	.E	.F
в.	Ъ	Ѓ	Ү	Ғ	Ӣ	…	†	‡	€	%	Љ	‹	Њ	Ќ	Ў	Ц
	402	403	201A	453	201E	2026	2020	2021	20AC	2090	409	2099	40A	45A	40B	40F
ғ.	Һ	Ұ	Ү	Ӣ	Ӣ	•	—	—		™	Љ	›	Њ	Ќ	Ў	Ц
	452	201B	2019	201C	201D	2022	2013	2014		2122	459	203A	45A	45C	45B	45F
а.		Ү	у	Ј	□	Ѓ	і	§	Ё	©	Є	«	¬		®	Ү
	40	40E	45C	455	44	490	4E	47	401	4B	404	4D	4C	4D	4E	407
б.	°	±	І	і	г	µ	¶	-	ё	№	є	»	ј	ѕ	ѕ	ї
	50	51	408	450	491	55	58	57	451	211A	454	5D	455	405	455	457
с.	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
	410	411	412	413	414	415	416	417	418	419	41A	41B	41C	41D	41E	41F
д.	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
	420	421	422	423	424	425	426	427	428	429	42A	42B	42C	42D	42E	42F
к.	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
	430	431	432	433	434	435	436	437	438	439	43A	43B	43C	43D	43E	43F
р.	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
	440	441	442	443	444	445	446	447	448	449	44A	44B	44C	44D	44E	44F

Рис. 2.6.1. Кодировка WINDOWS-1251

## ЗАПОМНИ

### Кодовая таблица (или таблица кодировки)

– это таблица, содержащая упорядоченный некоторым образом перечень кодируемых символов, в соответствии с которым происходит преобразование символа в его двоичный код и обратно.

Нижняя часть таблицы кодировки (латиница) полностью соответствует кодировке ASCII. Числа под буквами обозначают шестнадцатеричный код подходящего символа в Юникоде.

На рисунке 2.6.2 показаны только отличающиеся строки, поскольку все остальное совпадает с кодировкой Windows-1251.

	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.A	.B	.C	.D	.E	.F
в.	Ъ	Ѓ	Ү	Ғ	Ӣ	…	†	‡	€	%	Љ	‹	Њ	Ќ	Ў	Ц
	402	403	201A	453	201E	2026	2020	2021	20AC	2090	409	2099	40A	45A	40B	40F
ғ.	Һ	Ұ	Ү	Ӣ	Ӣ	•	—	—		™	Љ	›	Њ	Ќ	Ў	Ц
	452	201B	2019	201C	201D	2022	2013	2014		2122	459	203A	45A	45B	45B	45F
а.		Ү	у	Ө	□	Ө	і	§	Ё	©	Ғ	«	¬		®	Ү
	40	4B0	4B1	4DB	44	4EB	4E	47	401	4B	492	4B	4C	4D	4E	4AE
б.	°	±	І	і	ө	µ	¶	-	ё	№	ғ	»	ө	Ң	ң	ү
	50	51	406	456	4E9	55	56	57	451	211A	493	5B	4D9	4A2	4A3	4A7

Рис. 2.6.2. Официальная кодировка KZ-1048 (казахский стандарт)

Первые 128 символов стандарта Юникод совпадают с соответствующими символами US-ASCII.

В современных компьютерных текстовых документах распространен стандарт кодирования символов 16-битным кодом **Юникод** (Unicode – унифицированное кодирование), который содержит алфавиты практически всех письменных языков мира. В 1991 году организация «Консорциум Юникода» предложила стандарт Юникод. Стандарт разделен на две основные части: универсальный набор символов (*UCS – Universal Character Set*) и семейство кодировок (*UTF – Unicode transformation Format*).

Юникод имеет несколько форм представления: UTF-8, UTF-16, UTF-32. UTF-8 и UTF-16 являются кодировками переменной длины. Однако в UTF-8 символ может занимать минимум 8 бит, тогда как в UTF-16 длина символа начинается с 16 бит. UTF-32 будет охватывать все возможные символы в 32 бит. UTF-8 широко применяется в операционных системах и web-пространстве.

Стандарт кодирования Unicode отводит на каждый символ два байта, что позволяет закодировать многие алфавиты в одной таблице.

$$N = 2^1 = 2^{16} = 65\,536$$

### ПОЭКСПЕРИМЕНТИРУЙТЕ

В текстовом процессоре Microsoft Word нажмите на вкладку **Вставка** – группу **Символы** – строку **Другие символы**. Диалоговом окне **Символ** с помощью раскрывающегося списка определите числовой код знака кириллицы «Қ».

### ЕДИНИЦЫ ИЗМЕРЕНИЯ ДЛИНЫ ДВОИЧНОГО КОДА

При вводе текстовой информации с клавиатуры каждый вводимый символ кодируется или другими словами, преобразуется в числовой код.

Объем информации, представленной в форме текста, зависит от количества букв, знаков препинания и пробелов. Знаки препинания и пробелы также являются символами, имеющими свои коды. При стандартном 8-битном кодировании текста каждый символ кодируется одним байтом.

Например, вам приходит сообщение «Вы старшеклассник» длиной в 17 символов. Если закодировать, как принято на компьютерах, каж-



## 10 класс | информатика

дую букву и пробел одним байтом, данная информация займет 17 байт. В восьмибитной кодировке ASCII объем сообщения равен 136 бит ( $17 * 8 = 136$  бит). Если информацию «Вы старшеклассник» сократить до «Старшеклассник», эта величина соответственно будет равна 14 байт (или 112 бит). При 16-битной кодировке Unicode объем сообщения равен 272 бит ( $16 * 17 = 272$  бит).



1. Какой принцип кодирования текстовой информации используется в компьютере?
2. Как называется международная таблица кодировки символов?
3. В какой системе счисления представлены коды в таблицах кодировок?
4. Что такое ASCII? Как применять ASCII?
5. Как называется официальная кодировка казахского языка?



Выбирая нужную кодировку, выполните задания.

1. Сколько бит памяти займет слово «кодирование»?
2. Текст занимает 0,25 Кбайт памяти компьютера. Сколько символов содержит этот текст?
3. Считая, что каждый символ кодируется 16 бит, оцените информационный объем следующего предложения в кодировке Unicode: «Каждый символ кодируется восемью битами».
4. Закодируйте слова из стихотворения Абая: «Бес нәрседен қашық бол» с помощью фрагмента таблицы кодов KZ-1048.
5. Используя программу Excel, постройте нижеуказанную таблицу со следующими данными: название учебника, количество строк, количество символов в одной строке, количество страниц. Определите примерный объем текстовой информации каждого учебника и просуммируйте. Допустим, у вас есть флеш-накопитель объемом 4 Гбайт. Подумайте, поместится ли электронный вариант учебника на ваш флеш-накопитель?





Название учебника	Количество строк на одной странице	Количество символов в одной строке	Количество страниц	Объем информации		
				байт	Кбайт	Мбайт
Информатика	33	63	164	$= 33 * 63 * 164$ $= 340\ 956$	332	0,324

6. Сравните размеры сохраненных файлов.

- 1) В текстовом процессоре Microsoft Word откройте **Новый документ**.
- 2) Наберите фразу «Если долго трудиться, что-то получится».
- 3) Выберите команду **Сохранить как** и в диалоговом окне **Сохранить как** выберите из списка тип файла **Документ Word**. Задав документу соответствующее имя, сохраните в своей папке.
- 4) Повторите пункт 3. В этот раз для сохранения документа выберите из списка тип файла **Текст в формате RTF**. Сохраните документ под другим именем в своей папке.
- 5) Закройте окно текстового редактора и откройте папку с сохраненными файлами.
- 6) Сравните размеры сохраненных файлов. Постарайтесь узнать причину различия размеров этих файлов.



С какой целью была введена кодировка Unicode? Назовите последнюю версию Unicode (Unicode.org). Подготовьте сообщение.

## ТЕСТОВЫЕ ЗАДАНИЯ «ПРЕДСТАВЛЕНИЕ ДАННЫХ»



Знание



Понимание



Применение



Анализ



Синтез



1. Сумма чисел  $a = 2D_{16}$  и  $b = 57_8$  равна:

- A)  $100001000_2$ ;
- B)  $272_8$ ;
- C)  $143_{10}$ ;
- D)  $84_{16}$ ;
- E)  $1011100_2$ .



2. Информационный объем предложения «А ларчик просто открывался.», закодированного Unicode (16 бит на каждый символ), равен:

- A) 27 байт;
- B) 46 байт;
- C) 127 бит;
- D) 54 байта;
- E) 432 бита.



3. Определите таблицы истинности логических операций.

1	Конъюнкция	A)	X	$\bar{X}$	
			0	1	
			1	0	
2	Дизъюнкция	B)	X	Y	$X \wedge Y$
			0	0	0
			0	1	0
			1	0	0
			1	1	1
3	Инверсия	C)	X	Y	$X \vee Y$
			0	0	0
			0	1	1
			1	0	1
			1	1	1



4. Дан фрагмент таблицы истинности выражения F.  
Какое выражение соответствует следующей таблице?

X	Y	Z	$\neg Y$	$(\neg Y) \& Z$	F
0	0	0	1	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	1	0	0	

A)  $\bar{X} + Y + Z$ . B)  $X \cdot Y \cdot Z$ . C)  $\bar{X} + \bar{Y} + Z$ . D)  $X + \bar{Y} \cdot Z$ .



5. Найдите значение x в неравенстве  $634_8 < x < 1F6_{16}$ .

- A) 412.
- B) 502.
- C) 90.
- D) 89.
- E) 88.
- F) 496.



6. Напишите двоичный код, соответствующий следующим латинским буквам.

A	B	C	D	E



7. Трехзначное число x в шестнадцатеричной системе счисления заканчивается на цифру  $7_{16}$ , а в восьмиричной системе счисления начинается на цифру  $26_8$ . Укажите это число в десятичной системе счисления.

ОТВЕТ: \_\_\_\_\_



8. Установите соответствие «Алгоритма построения таблицы истинности».

Действие	Порядок выполнения
<b>A)</b> Выписать наборы входных переменных	1
<b>B)</b> Провести заполнение таблицы истинности по столбцам, выполняя логические операции в соответствии с последовательностью	2
<b>C)</b> Определить количество столбцов: число переменных + число операций	3
<b>D)</b> Подсчитать количество переменных $n$ в логическом выражении	4
<b>E)</b> Подсчитать количество логических операций в формуле	5
<b>F)</b> Определить число строк в таблице по формуле $m = 2^n$ , где $n$ – количество переменных	6
<b>G)</b> Установить последовательность выполнения логических операций с учетом скобок и приоритетов	7



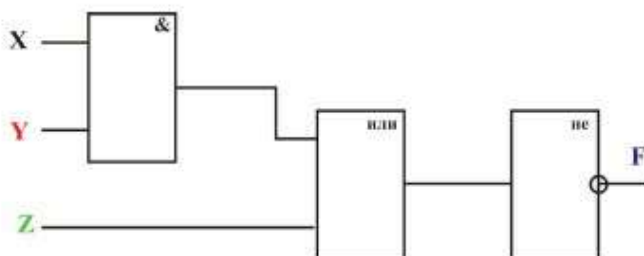
9. Установите соответствие «Приоритет выполнения логических операций» в примере.



$(A \vee B) \wedge (\neg A \vee \neg B)$



10. Напишите логическое выражение по следующей схеме.





**ПРОЕКТНАЯ ДЕЯТЕЛЬНОСТЬ ЗА I ЧЕТВЕРТЬ****1. Исследуйте локальную сеть своего компьютерного класса (школы). Подготовьте отчет, в котором укажите:**

- Быстродействие сети.
- Топологию сети.
- Типы используемых каналов связи (коаксиальный кабель, витая пара, оптоволокно).
- Укажите используемые коммуникационные устройства (концентраторы, коммутаторы, маршрутизаторы) и их паспортные характеристики (производитель, количество портов, дополнительные свойства).
- Опишите функции сервера вашей локальной сети (предоставляемые пользователям устройства и услуги, в том числе связь с Интернетом).
- Нарисуйте полную схему сети с использованием программы NetEmul.
- Кто обслуживает локальную сеть?
- Какое программное обеспечение используется для обеспечения работоспособности сети?
- Определите IP-адрес всех компьютеров в компьютерном классе.
- Определите DNS-адрес вашего региона.

**2. Подготовьте мини-проект. Проведите исследование по плану.**

**Цель проекта:** поиск наилучшей антивирусной программы для пользователя ПК в домашних условиях.

**1) Анализ и отбор значимых факторов.**

- Цена – стоимость покупки лицензионной версии на один год.
- Популярность – место данного антивируса в рейтинге.
- Лечение активного заражения – способность антивируса устранять активные заражения.
- Объем памяти – необходимый объем свободной оперативной памяти для установления программы.
- Скорость работы – скорость проверки файлов на наличие вирусной инфекции.
- Простота использования – комфортность и простота работы с ПК при использовании данного антивируса.
- Представление демо-версии – возможность использования временной бесплатной версии данного продукта.

## ПРОЕКТНАЯ ДЕЯТЕЛЬНОСТЬ ЗА I ЧЕТВЕРТЬ

2) Формализованное представление проведенного анализа информации.

Варианты	Факторы						
	Цена	Популярность	Лечение активного заражения	Объем памяти	Скорость работы	Простота использования	Представление демо-версии

3) Анализ и обоснование результатов.

Укажите лучшие два объекта исследования (1-е и 2-е место) и объект, стоящий на последнем месте в таблице.

3. Исследуйте указанные темы и объясните связь между ними.



## 3.1. Пользовательские процедуры

Какая структура у программы Python?

Как описать переменные?

Какие типы данных вы знаете?

– Что такое подпрограмма;  
– виды подпрограмм;  
– как они описываются  
и вызываются в основной  
программе.

**Подпрограмма** – Ішкі программа – Subprogram

**Процедура** – Процедура – Procedure

**Функция** – Функция – Function

**Системные функции** – Жүйелік функциялар –  
System functions

**Пользовательские функции** – Қолданушы функ-  
циялар – Custom functions

вы узнаете

ключевые слова

Программы большого размера обычно создают и отлаживают по частям. Отдельные части программы называют **подпрограммами**. Они оформляются таким образом, чтобы было возможно их применение при решении аналогичных подзадач в той же программе или даже при решении других задач.

Любая исполняемая программа содержит одну единственную **основную (или главную)** программу, с которой начинается работа, и произвольного количества **подчиненных ей подпрограмм**.

Основная программа может обратиться к любой подпрограмме непосредственно или косвенно. Подпрограмма точно таким же образом может вызвать любую другую подпрограмму, включая и себя. Ни одна из подпрограмм не может обратиться к основной программе.

Подпрограмма может как просто выполнять какие-то действия (например, вывести массив на экран), так и возвращать конкретный результат в виде некоторого значения (например, вычислять среднее арифметическое массива и возвращать получившееся число). Подпрограммы первого типа названы **процедурами**, а второго – **функциями** (это похоже на

### ЗАПОМНИ

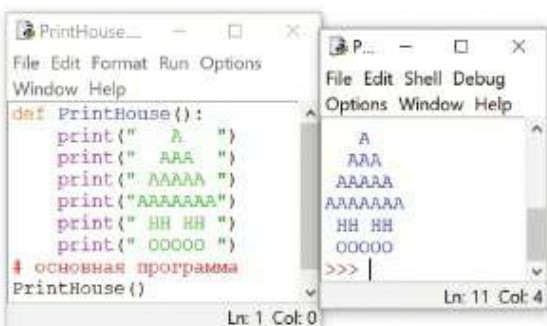
#### ПОДПРОГРАММОЙ

называют именованную группу операторов, выполняющих некоторую задачу. Подпрограмму запускают, вызвав по имени необходимое количество раз из любых мест программы.

## 10 класс | информатика

функции в математике, у которых для нескольких аргументов тоже вычисляется единственное значение).

Подпрограммы освободили разработчиков от рутинного повторения однотипного кода. Например, можно написать подпрограмму, которая будет выводить на экране домик, изображенный символами. Теперь для изображения его на экране хватит одной команды (вызова подпрограммы) (рис. 3.1.1).

**ПРИМЕР.**

**Рис. 3.1.1.** Код программы **PrintHouse** и результат

**ЗАПОМНИ****ВЫЗОВОМ ПОДПРОГРАММЫ**

называется обращение к ней по имени с целью ее использования. Вызов – это запуск подпрограммы.

Как видно на рисунке 3.1.1, в язык программирования была введена новая команда **PrintHouse**, расшифрованная прямо в теле программы. Для того, чтобы процедура заработала, в основной программе (или в другой процедуре) необходимо ее **вызвать** по имени.

Процедура начинается с ключевого **def** (от англ. *define* – определить). Затем записывается имя процедуры. После нее ставятся пустые скобки и двоеточие. Тело процедуры печатается с отступом. Процедура расположена выше основной программы, чтобы в момент вызова процедуры транслятор уже знал о ней. Обратите внимание, что при вызове процедуры, также как и в ее описании, после имени процедуры нужно ставить круглые скобки (в данном случае – пустые).



Чтобы начать программировать на Python, необходимо установить программную среду на свой компьютер. Ее можно взять с сайта разработчиков <https://www.python.org/downloads>. Для этого необходимо выбрать версию Python и загрузить бесплатный инсталляционный пакет, соответствующий операционной системе, установленной на вашем компьютере. Запустите полученный файл и следуйте инструкциям мастера установки.

С Python можно работать различными способами. Первый способ заключается в применении командной строки, когда программы и команды запускаются в консольном окне. Однако для создания и редактирования текстов программ удобнее использовать среду разработки IDLE, которая включена в дистрибутив Python. Она содержит интерпретатор команд **Python Shell** и специальный текстовый редактор **IDLE** для Python (рис. 3.1.2). В окне интерпретатора **Python Shell** можно выполнять те же команды, что и в консоли.



Рис. 3.1.2. Запуск IDLE

Кроме интерпретатора Python Shell среды IDLE можно воспользоваться различными интерактивными оболочками: IPython, Atom, Geany, PyCharm и др.

## СТРУКТУРА ПРОГРАММЫ С ПОДПРОГРАММОЙ

Подпрограммы, входящие в состав системы программирования, обычно называют **стандартными**, или **системными**. Все остальные подпрограммы, которые пользователь пишет самостоятельно или заимствует из каких-либо других источников, называют **нестандартными**, или **пользовательскими** (рис. 3.1.3).

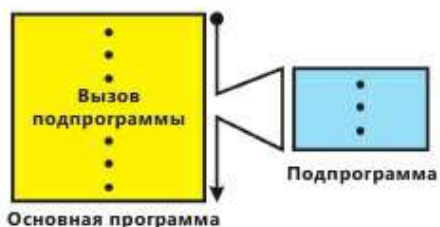


Рис. 3.1.3. Структура программы с подпрограммой



Рассмотрим программу, печатающую строку из 20 символов «\*», затем строку «**Title: Program 1**», затем опять строку из 20 символов «\*», потом строку «**The end**» и, наконец, опять строку из 20 символов «\*» (рис. 3.1.4):

```

starline0.py - C:/Pyt...
File Edit Format Run Options Window Help
print ("*"*20)
print ("Title: Program 1")
print ("*"*20)
print ("The end")
Ln: 4 Col: 4

Python 3...
File Edit Shell Debug Options Window Help
*****
Title: Program 1
*****
The end
*****
>>> |
Ln: 10 Col: 4
  
```

Рис. 3.1.4. Пример программы, печатающей строку заданных символов

Заметим, что в этой программе имеется фрагмент, повторяющийся три раза. В связи с этим его можно обозначить, например, **starline**, и определить один раз, а затем указывать только это имя (рис. 3.1.5).

```

starline.py - C:\PythonP...
File Edit Format Run Options Window Help
def starline():
    print ( "*"*20)
    #*****
starline()
print ("Title: Program 1" )
starline()
print ( "The end" )
starline()
Ln:

Python ...
File Edit Shell Debug Options Window Help
*****
Title: Program 1
*****
The end
|*****
>>>
Ln: 36 Col: 0
  
```

Рис. 3.1.5. Использование процедуры **starline**

## ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Переделайте программу про строки из звездочек так, чтобы строка состояла из символов «=», а ее длина была равна 70 символам.
2. Добавьте в программу процедуру **showmessage**, печатающую на экране сообщение «**Работает процедура**».
3. Добавьте в основную программу команду вызова процедуры **showmessage**.



1. Что такое подпрограммы? Для чего их применяют?

2. Как описываются процедуры в языке Python?



3. Достаточно ли одного формального описания процедуры в тексте программы для ее выполнения?

4. Как осуществляется вызов процедуры из основной программы?

5. Какие процедуры называют стандартными (системными)?

6. Что такое пользовательская процедура?



1. Напишите процедуру, которая будет печатать на экране изображенный символами рисунок 3.1.6.



```

      ^^
      ///()\\      #
      \\\|\\|\\|\\|      #
      #####
      #####
      #              #
      ==            ==
      %%%%%%%%%%%
  
```

Рис. 3.1.6. Результат программы

2. Напишите процедуру, которая будет печатать на экране справочную информацию о физических постоянных. Абсолютный ноль температуры равен  $-237,15^{\circ}\text{C}$ .



Ускорение свободного падения  $g = 9,806665 \text{ м/с}^2$ .  
Скорость света в вакууме  $c = 299\,792\,458(\pm 1,2) \text{ м/с}^2$ .



1. Напишите процедуру, которая будет печатать на экране таблицу квадратов числа 2.
2. Дано описание процедуры:

```
def Proc()
for i in range(1,11):
    for j in range(1,11):
        print("{:4d}".format(i*j),end=" ")
    print()
```

- 1) Что будет выведено на экран в результате ее выполнения?
- 2) Напишите программу, вызывающую процедуру **Proc**.

## 3.2. Параметры

Что называют подпрограммой?  
Как описать подпрограмму?  
Какие типы подпрограмм вы знаете?

- Механизм передачи параметров из основной программы в подпрограмму;
- способы передачи параметров по значению и по ссылке;
- об области видимости локальных и глобальных переменных.

**Аргумент** – Аргумент – Argument  
**Параметры** – Параметрлер – Parameters  
**Глобальный** – Жаһандық – Global

вы узнаете

ключевые слова

Во многих случаях полезно передавать данные в вызываемую процедуру, чтобы она могла с ними как-то взаимодействовать. Процедура **PrintHouse** при каждом вызове выполняет одни и те же действия. Чтобы



подпрограмма могла совершать не только одинаковые, но и однопипные действия (похожие алгоритмы, но с разными параметрами), был разработан механизм передачи параметров из основной программы в подпрограмму. В подпрограмме описываются некоторые абстрактные переменные (называемые **аргументами** или **параметрами**), фактические значения которых задаются только в момент вызова подпрограммы.

**ПРИМЕР 1.** Предположим, что в программе требуется многократно определять, является ли число простым.

Для этого необходимо определить, есть ли делители числа **N** в диапазоне от **2** до **N/2**. Если делители не найдены, **N** – простое число, в противном случае оно не является простым. Учтите, что в алгоритме имеются два выхода из цикла. Первый выход выполняется при естественном завершении цикла, а второй выход – досрочный. При нахождении хотя бы одного делителя нет необходимости продолжать цикл. Алгоритм, решающий эту задачу для переменной **N**, можно записать, как показано на *рис. 3.2.1*.

```
N=int(input())
simple=1
for i in range(2,N//2):
    if N%i==0:
        simple=0
        break
if simple==1:
    print(N, " простое число")
else:
    print(N, " не является простым")
```

**Рис. 3.2.1.** Алгоритм определения, является ли число **N** простым

Писать такой цикл каждый раз, когда нужно простое число, очень утомительно. Кроме того, легко сделать ошибку или опечатку, которую будет сложно найти. Поэтому лучше оформить этот вспомогательный алгоритм в виде процедуры. Но этой процедуре нужно передать аргумент – число, которое надо проверить, является ли оно простым. Программа получается такая (см. *рис. 3.2.2*).

Основная программа содержит всего одну команду – вызов процедуры **PrintSimple** с аргументом **37**. В заголовке процедуры в скобках записывают тип и внутреннее имя параметра (то есть имя, по которому к нему можно обращаться в процедуре).

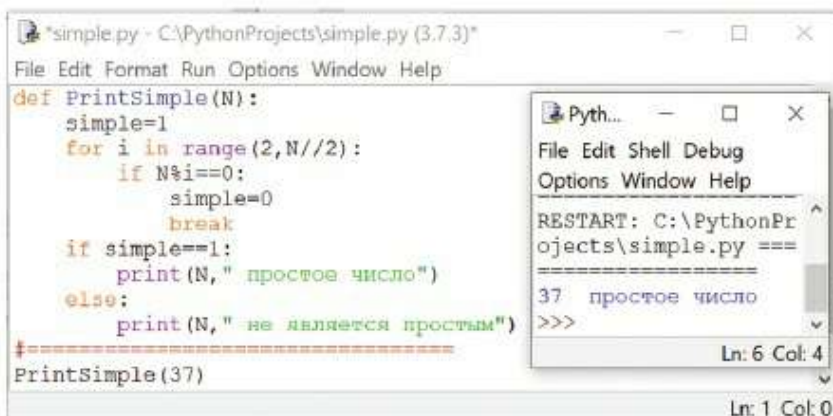


Рисунок 3.2.2. Пример процедуры с параметром

**ЗАПОМНИ****АРГУМЕНТ  
(ИЛИ ПАРАМЕТР)**

– это значение, которое передается подпрограмме из основной программы при ее вызове.

В процедуре объявлена **локальная** (внутренняя) переменная **simple**. Она известна только внутри этой процедуры, обратиться к ней из основной программы и из других процедур невозможно. **Глобальные** переменные объявляются вне тела какой-либо функции, и поэтому область видимости таких переменных распространяется на всю программу (рис. 3.2.3).

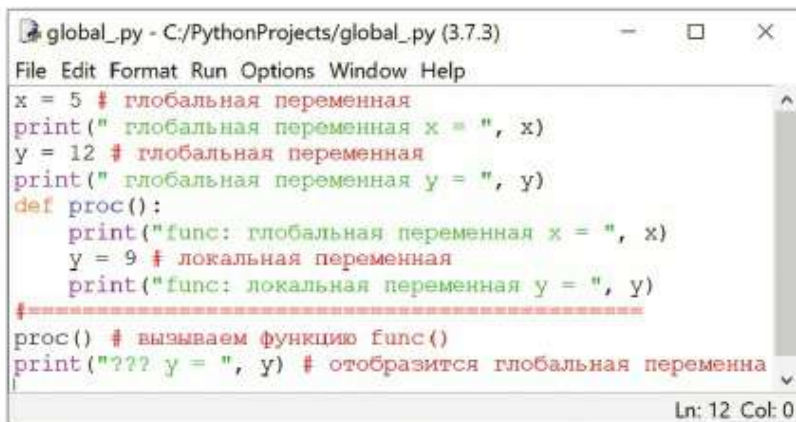
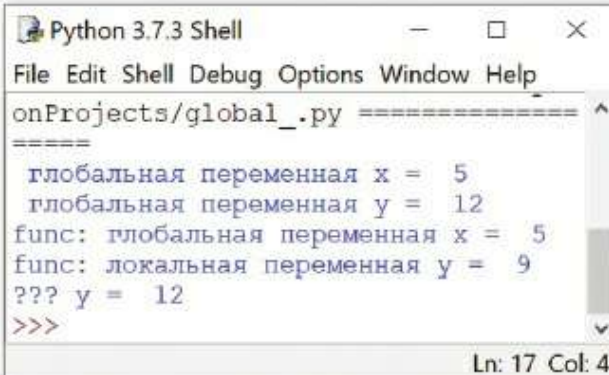


Рис. 3.2.3. Пример использования глобальной переменной

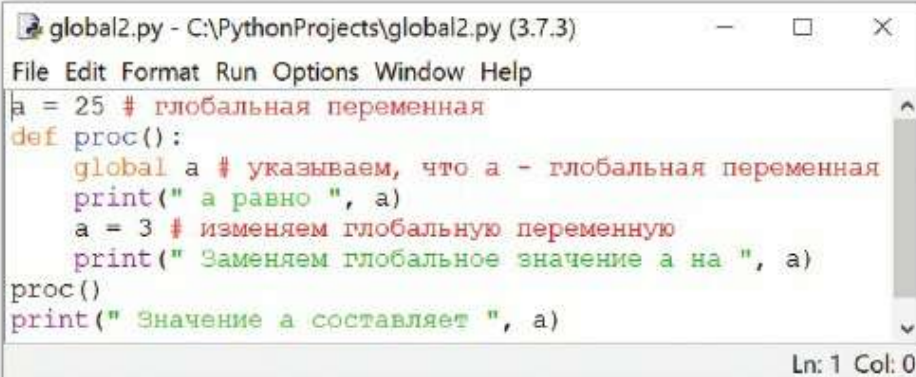
После выполнения программы получим следующий результат (рис. 3.2.4):



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
onProjects/global_.py =====
=====
    глобальная переменная x = 5
    глобальная переменная y = 12
func: глобальная переменная x = 5
func: локальная переменная y = 9
??? y = 12
>>>
Ln: 17 Col: 4
```

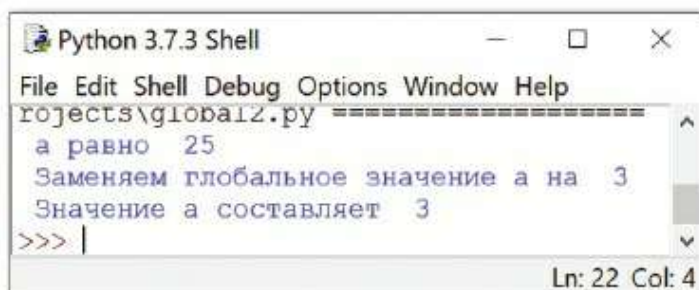
Рис. 3.2.4. Результат выполнения программы

Внутри процедуры мы обратились к глобальной переменной **x** и вывели ее значение на экран. Далее внутри функции создается локальная переменная **y**, причем ее имя совпадает с именем глобальной переменной – в этом случае при обращении к **y** выводится содержимое локальной переменной, а глобальная остается неизменной. Если мы хотим изменить содержимое глобальной переменной внутри функции, то используем ключевое слово **global** (рис. 3.2.5 и 3.2.6)



```
global2.py - C:\PythonProjects\global2.py (3.7.3)
File Edit Format Run Options Window Help
a = 25 # глобальная переменная
def proc():
    global a # указываем, что a - глобальная переменная
    print(" a равно ", a)
    a = 3 # изменяем глобальную переменную
    print(" Заменяем глобальное значение a на ", a)
proc()
print(" Значение a составляет ", a)
Ln: 1 Col: 0
```

Рис. 3.2.5. Пример использования ключевого слова **global**



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
projects\global12.py =====
а равно 25
Заменяем глобальное значение а на 3
Значение а составляет 3
>>> |
Ln: 22 Col: 4
```

**Рис. 3.2.6.** Результат использования ключевого слова **global**

Параметров может быть несколько, в этом случае они перечисляются в заголовке процедуры через запятую. Например, процедуру, печатающую строку, состоящую из **N** заданных символов, можно записать, как показано на *рис. 3.2.7*:



```
LineOfChar.py - C:\Pyt...
File Edit Format Run Options Window Help
def LineOfChar(N, ch):
    print(ch*N)
=====
LineOfChar(10, '@')
Ln: 1 Col: 0

Pyth...
File Edit Shell Debug
Options Window Help
@@@@@@@@@@@
>>>
Ln: 6 Col: 4
```

**Рис. 3.2.7.** Пример процедуры с несколькими параметрами и ее результат

При использовании параметров в процедуре каждый из них должен иметь уникальное имя. Когда программа вызывает процедуру, значения параметров присваиваются именам параметров слева направо.



**ПРИМЕР 2.** Рассмотрим пример программы, в которой осуществляется параллельный перенос отрезка на заданный вектор. Отрезок задан координатами своих концов (элементы списка с индексом **0** – абсцисса, с индексом **1** – ордината). Вектор сдвига  $\overline{OX}$  задан координатами точки **X** (точка **O** – начало координат).

Решение задачи может выглядеть так (*рис. 3.2.8*).





```

"shift.py - C:\PythonProjects\shift.py (3.7.3)"
File Edit Format Run Options Wind Python 3.7...
def shift(a, b, x):
    a[0] = a[0] + x[0]
    a[1] = a[1] + x[1]
    b[0] = b[0] + x[0]
    b[1] = b[1] + x[1]

C = [1, 2]
D = [-3, 2]
X = [1, 1]
shift(C, D, X)
print(C, D) # [2, 3] [-2, 3]
Ln: 9 Col: 0

Python 3.7...
File Edit Shell Debug Options
Window Help
TART: C:\PythonProjects\
shift.py =====
[2, 3] [-2, 3]
>>> |
Ln: 13 Col: 4
Ln: 9 Col: 0
  
```

**Рис. 3.2.8.** Программа параллельного переноса отрезка на заданный вектор

Локальные переменные – параметры **a**, **b** – связаны со значениями глобальных переменных **C** (равной [1, 2]) и **D** = [-3, 2]. Локальная переменная **X** связана со списком [1, 1]. В процедуре **shift** вычисляются новые значения элементов списков **a** = [2, 3] и **b** = [-2, 3]. Таким образом, изменяются значения глобальных переменных **C** и **D**.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Запустите программу и протестируйте ее для различных входных значений переменных **C** и **D**.
2. Измените программу так, чтобы вектор сдвига задавался переменной **Y**.



1. Что называют параметрами процедуры? Для чего они используются?
2. Какие переменные называются локальными? Где они описываются?
3. Какие переменные называются глобальными? Как осуществляется доступ к глобальным переменным внутри процедуры?
4. Как описываются процедуры, имеющие несколько параметров?



1. Напишите программу, выводящую на экран в столбик все цифры заданного числа, начиная с последней.



2. Напишите процедуру, печатающую на экране все делители заданного числа (в одну строчку).



3. Напишите процедуру, которая выводит на экран квадрат из звездочек со стороной **N**, где **N** – параметр процедуры (натуральное число).

4. Напишите процедуру, получающую в качестве параметра возраст человека в годах, и выводящую этот возраст со словом «год», «года» или «лет». Например, «61 год», «32 года», «15 лет».



Дана программа нахождения действительных корней уравнения  $ax^2 + bx + c = 0$ , ( $a \neq 0$ ).



```
from math import sqrt
print("Enter a, b, c")
a=int(input())
b=int(input())
c=int(input())
D=b*b-4*a*c
print("D=",D)
if D>0:
    x1=(-b-sqrt(D))/(2*a)
    x2=(-b+sqrt(D))/(2*a)
    print("x1=",x1," x2=",x2)
elif D==0:
    x1=(-b)/(2*a)
    print("x1=",x1)
else:
    print("No real roots")
```

Измените программу так, чтобы она использовала процедуру, которая выводит на печать все корни квадратного уравнения при заданных коэффициентах **a**, **b** и **c**. Вызовите эту процедуру из основной программы.

## 3.3. Пользовательские функции

Какие способы передачи параметров из основной программы в процедуру вы знаете? Чем отличаются локальная и глобальная переменные?

- Отличие функции от процедуры;
- что такое возвращаемое значение функции;
- назначение оператора `return`.

**Функция** – Функция – Function

**Возвращаемое значение** – Қайтып оралатын мәндер – Return value

**Результат** – Нәтиже – Result

вы узнаете

ключевые слова

Функции – один из самых важных компонентов языка Python. До этого мы использовали только стандартные функции языка программирования (например, **abs**, **sin**, **cos**). Теперь научимся создавать свои функции. Функция, как и процедура, – это вспомогательный алгоритм, который может принимать аргументы. Но, в отличие от процедуры, функция всегда возвращает **значение-результат**. Результатом может быть число, символ или объект другого типа. По окончании выполнения вызываемой функции осуществляется возврат значения в точку ее вызова.

Для передачи возвращаемого значения в вызывающую функцию используется оператор **return** (от англ. *вернуть*).

Оператор **return** выполняет следующие действия:

- вычисляет выражение, стоящее справа от слова **return**;
- прекращает выполнение тела функции;

– возвращает значение вычисленного выражения в качестве результата функции.



**ПРИМЕР 1.** Создадим программу, которая конвертирует заданное число **N** из десятичной системы счисления в системы счисления с основанием **2**, **5** и **7**. Для этого опишем функцию **Convert**. Младшая

### ЗАПОМНИ

В Python все подпрограммы называются **функциями**, но они могут быть двух типов: возвращающими или не возвращающими значение.



цифра в такой записи – это частное от деления числа на основание новой системы счисления. Если частное от деления не равно нулю, продолжаем деление и находим следующую цифру числа и т.д. Алгоритм, решающий эту задачу для переменной **N**, можно записать так (рис. 3.3.1):

```

convert.py - C:\PythonProjects\co
File Edit Format Run Options Win
def Convert(N,b):
    S=0
    p=1
    while N != 0:
        S+=N%b*p
        p*=10
        N=N//b
    return S
X=int(input("X="))
print(X, " (10) =", Convert(X,2), " (2) ")
print(X, " (10) =", Convert(X,5), " (5) ")
print(X, " (10) =", Convert(X,7), " (7) ")

Python 3.7...
File Edit Shell Debug Options
Window Help
X=33
33 (10) = 100001 (2)
33 (10) = 113 (5)
33 (10) = 45 (7)
>>> |
Ln: 19 Col: 4
Ln: 1 Col: 0
  
```

**Рис. 3.3.1.** Программа перевода числа из одной системы счисления в другую

Чтобы получить последнюю цифру числа, нужно взять остаток от деления числа на 10. Полученную цифру умножаем на вес разряда и суммируем с предыдущим значением. Затем последняя цифра отсекается, и мы переходим к следующей цифре. Цикл продолжается до тех пор, пока значение **N** не становится равно нулю.

Так же, как и в процедурах, в функциях можно объявлять и использовать локальные переменные. Они входят в область видимости только этой функции, для всех остальных функций и процедур они недоступны.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Можно ли с помощью функции **Convert** преобразовать числа в системы счисления с основанием 8, 16 и 20? Какие изменения необходимо для этого внести в программу?



В теле функции может быть несколько операторов **return**, после выполнения любого из них работа функции заканчивается.

**ПРИМЕР 2.** Описать функцию, возвращающую максимальное из двух целых чисел, заданных в качестве аргумента.

Функции, созданные в программе подобным образом, применяются точно так же, как и стандартные функции. Их можно вызывать везде, где может использоваться выражение того типа, который возвращает функция.

Имена функций в Python являются переменными, содержащими адрес функции. Этот адрес можно присвоить другой переменной и вызвать функцию с другим именем.

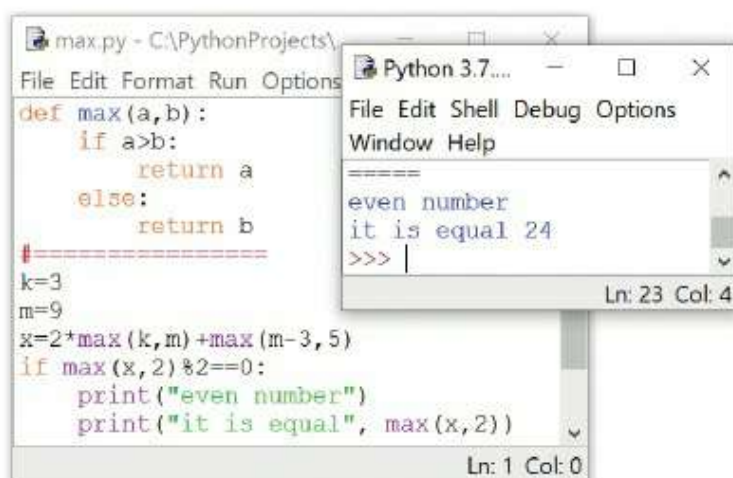


Рис. 3.3.2. Пример использования функции **max**

```
def summa(a, b):
    return a + b
F = summa
X = F(5, 12) # вызываем функцию с другим именем
```

Параметры функции могут принимать значения по умолчанию:

```
def summa(a, b=4):
    return a + b
```

## 10 класс | информатика

```
X = summa(3) # вместо y подставляется значение по
             # умолчанию
Y = summa(35, 28) # теперь значение второго параметра
                 # равно 28
```

Так как имя функции – обычная переменная, то мы можем передать ее в качестве аргумента при вызове функции:

```
def summa(a, b):
    return a + b
def func(f, x, y):
    return f(x, y)
z = func(summa, 10, 3) # передаем summa в качестве
                       # аргумента
```

Этот пример демонстрирует, как из функции **func()** можно вызвать функцию **summa()**.

При описании функции можно присваивать значения конкретным параметрам (использовать ключевые аргументы):

```
def func(x, y=4, z=15):
    print("x равно", x, ", y равно", y, ", z равно", z)
func(3, 9) # x=3, y=9, z=15
func(48, z=35) # x=48, y=4, z=35
func(z=50, x=100) # x=100, y=4, z=50
```

Ошибкой будет являться вызов функции, при котором не задан аргумент **x**, т.к. для него не указано значение по умолчанию.



1. Что такое функция? Чем функция отличается от процедуры?
2. Как описываются функции в тексте программы?
3. Можно по тексту программы определить, какое значение возвращает функция?
4. Как передать результаты вычислений из функции?
5. В каких целях можно использовать оператор **return**?
6. Перечислите основные преимущества использования функций в программе.



1. Опишите функцию **NOD** по алгоритму, представленному на блок-схеме (рис. 3.3.3).

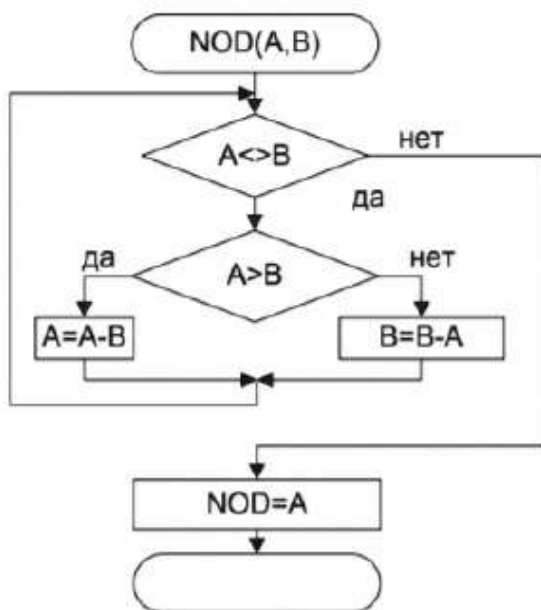


Рис. 3.3.3. Блок-схема алгоритма нахождения наибольшего общего делителя двух чисел

2. Запишите функцию возведения числа в квадрат.
3. Создайте функцию для вычисления площади треугольника, заданного тремя сторонами.
4. Напишите функцию **profit**, которая вычисляет доход по вкладу. Исходные данные для функции: величина вклада, процентная ставка (годовых) и срок вклада (количество дней).
5. Напишите функцию, которая сравнивает два целых числа и возвращает результат сравнения в виде одного из знаков:  $>$ ,  $<$  или  $=$ .
6. Напишите функцию, которая вычисляет факториал натурального числа **n**.



1. Заголовок функции имеет вид `def F(x, y, z=0):`  
Укажите правильные варианты вызова подпрограммы:

- A) `F(1, 2)`
- B) `F(25, z=24)`
- C) `F(z=12, x=150)`
- D) `F(30,40,50)`

2. Что будет выведено на экран в результате работы программы (рис. 3.3.4)? Почему?

- A) 1.    B) 2.    C) -3.    D) 24.

```
def F(x):
    return (x-1)*(x-3)
a=-3; b=3; d=0.1
t=a; M=a; R=F(a)
while t<b:
    if F(t)<R:
        M=t; R=F(t)
    t=t+d
print(M)
```

Рис. 3.3.4. Пример использования функции

3. Выполните трассировку программы:

```
def F(x):
    return 2*(x*x-1)*(x*x-1)+27
a=-20; b=20
M=a; R=F(a)
for t in range(a,b+1):
    if F(t)<=R:
        M=t; R=F(t)
print(M+R)
```



## 3.4. Рекурсия

Что такое возвращаемое значение функции?  
Опишите механизм передачи параметров из основной программы в подпрограмму.

– Что такое рекурсия;  
– правила описания рекурсивных функций;  
– что такое прямая и косвенная рекурсия.

**Рекурсия** – Рекурсия – Recursion

**База рекурсии** – Рекурсия базасы – Recursion basis

**Шаг рекурсии** – Рекурсия қадамы – Recursive step

Большая часть всех шуток о рекурсии касается бесконечной рекурсии, в которой нет условия выхода: чтобы понять рекурсию, нужно сначала понять рекурсию.

вы узнаете

ключевые слова

это интересно

Рекурсия – достаточно распространенное явление, которое встречается не только в областях науки, но и в повседневной жизни. Примеры рекурсии можно встретить в литературе и поэзии, в изобразительном искусстве и музыке, в окружающей природе. Пример рекурсии в поэзии:

Кто вечно хнычет  
И скучает,  
Тот ничего  
Не замечает.  
Кто ничего  
Не замечает,  
Тот ничего

Не изучает.  
Кто ничего  
Не изучает,  
Тот вечно хнычет  
И скучает.  
(Если скучно стало,  
Начинай сначала!)

Роман Сеф  
«Бесконечные стихи»

Примеры рекурсии в искусстве и окружающей природе (рис. 3.4.1–3.4.2):



**Рис. 3.4.1.** Цветная капуста имеет сложную рекурсивную форму



**Рис. 3.4.2.** Пример рекурсии в архитектуре

Рекурсия – мощный инструмент программирования. Она широко применяется при решении игровых и переборных задач. В программировании многие задачи решаются только с помощью рекурсии, или их решение иным способом значительно сложнее.

**ПРИМЕР.** Рассмотрим понятие рекурсии на классическом примере вычисления факториала  $n!$ . Как вы знаете, **факториал** числа  $n$  – произведение всех натуральных чисел до  $n$  включительно:

$$n! = 1 \times 2 \times \dots \times (n-1) \times n = \prod_{k=1}^n k$$

По определению полагают  $0! = 1$ .

Приведенное выражение можно преобразовать следующим образом:

$$n! = n \times (n-1) \times \dots \times 2 \times 1 = n \times (n-1)!$$

## ЗАПОМНИ

### РЕКУРСИВНАЯ ФУНКЦИЯ

– это функция, которая вызывает сама себя непосредственно или с помощью других функций.

То есть факториал числа  $n$  равен произведению числа  $n$  на факториал числа  $(n - 1)$ . В свою очередь, факториал числа  $(n - 1)$  можно представить произведением числа  $(n - 1)$  на факториал числа  $(n - 2)$  и т.д. Поэтому, если вычисление факториала  $n$  оформить в виде функции, то в теле этой функции будет команда вызова функции вычисления факториала числа  $(n - 1)$ , т.е. функция будет вызывать сама себя. Данный способ вызова называется

**рекурсией**, а функция, которая обращается сама к себе, называется **рекурсивной функцией**:

$$n! = \begin{cases} n! = 1, & \text{при } n = 0 \\ n \times (n-1)!, & \text{при } n > 0 \end{cases}$$

Рекурсивная функция обязательно должна содержать условие остановки рекурсии (базовый случай), чтобы не вызвать зацикливания программы.

Функция, вычисляющая факториал натурального числа **n!** (**n ≥ 0**), на Python определяется следующим образом:

```
def factorial(n):
    if n==0:
        return 1
    else:
        return n*factorial(n-1)
```

Когда **factorial()** вызывается с аргументом **0**, функция возвращает **1**. В противном случае она возвращает произведение **n \* factorial(n - 1)**. Для вычисления этого значения **factorial()** вызывается с **n - 1**. Это происходит, пока **n** не станет равно **0**.

Например, при передаче числа 2 у нас образуется следующая цепочка вызовов (см. рис. 3.4.3).

Ничего не умножается, пока мы спускаемся к **базовому случаю factorial(0)**. Затем мы начинаем подниматься обратно, по одному шагу (**шаг рекурсии**). Общее количество вложенных вызовов называют **глубиной рекурсии**. В случае с **factorial()** всего будет **n + 1** вызовов.

При каждом обращении процедуры к самой себе компилятор запоминает в стеке **адрес возврата**, чтобы продолжить работу программы, когда рекурсия закончится.

Если рекурсия бесконечная, вся свободная память компьютера (его ОЗУ) может быстро исчерпаться, а это приведет к краху работы программы. К недостаткам применения рекурсии относятся быстрое увеличение времени выполнения программы и переполнение стека.

Программы, в которых используются рекурсивные функции, отличаются простотой, наглядностью и компактностью текста. Однако за эту простоту приходится расплачиваться неэкономным использованием оператив-

## 10 класс | информатика

ной памяти. Кроме того, рекурсивный вызов функции иногда существенно замедляет работу программы.

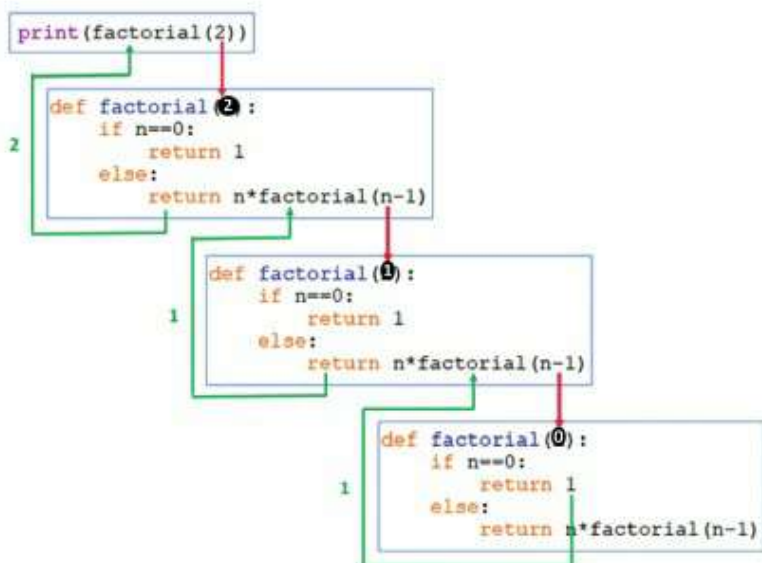


Рис. 3.4.3. Последовательность вызовов функции **factorial()**

## ЗАПОМНИ

### ШАГ РЕКУРСИИ (РЕКУРСИВНЫЙ ВЫЗОВ)

– обращение функции к себе самой, но с другим значением параметра, таким, который приближает к базовому случаю.

## ЗАПОМНИ

**СТЕК** (англ. *stack* – кипа, стопка)

– специально организованная область ОЗУ, используемая для временного сохранения данных или адресов.

## ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Напишите программу, вычисляющую факториал числа **n** с помощью рекурсивной функции.
2. Добавьте в программу команды, позволяющие пользователю вводить с клавиатуры значение **n**.
3. Вычислите значение факториала для **n = 5, 8, 15, 48**. Объясните полученный результат.





1. Что такое рекурсия? Приведите примеры рекурсии в природе.
2. Почему любое рекурсивное описание состоит из двух частей?
3. Объясните, как выполняется рекурсивная функция.
4. В каком случае рекурсия никогда не остановится? Докажите, что в рассмотренных задачах этого не случится.
5. Как вы понимаете понятие стека? Каким образом он используется при выполнении программ?
6. В каких случаях при использовании рекурсии может возникнуть переполнение стека?
7. В чем достоинства и недостатки рекурсивных функций? В каких случаях применение рекурсии оправдано?



1. Дано натуральное число  $n$ . Выведите все числа от 1 до  $n$ . Используйте рекурсивную функцию.
2. Вычислите сумму чисел в интервале, заданном вводимыми числами. Используйте рекурсивную функцию. (Подсказка: условием окончания рекурсии станет ситуация, когда верхняя граница на единицу больше нижней границы, то есть интервал задан двумя соседними целыми числами.)
3. Возведите заданное число в заданную степень. Используйте рекурсивную функцию. (Подсказка: здесь условием окончания рекурсии будет равенство нулю числа-степени. Обязательно следует предусмотреть вызовы функции для четной степени и нечетной степени.)
4. Реализуйте алгоритм Евклида для нахождения наибольшего общего делителя с помощью рекурсивной функции. Алгоритм Евклида находит наибольший общий делитель чисел  $A$  и  $B$  по следующему правилу:

$$NOD(A, B) = \begin{cases} A, & \text{если } B = 0 \\ NOD(B, A \% B), & \text{если } B \neq 0 \end{cases}$$



1. Исправьте ошибки в описании рекурсивной функции. Аргументируйте свой ответ.

```
def factorial(n):
    if n<=5:
        return 1
```



```
else:
    return n*factorial(n)
```

2. Исследуйте, что получится при вызове описанной ниже процедуры **Recursion (5)**. Почему?

```
def Recursion(n):
    if n>1:
        Recursion(n-1)
    print(n)
```

3. Дано описание рекурсивной функции **F**.

```
def F(n):
    if n>1:
        print(n)
        F(n-3)
        F(n//3)
```

Запишите подряд без пробелов и разделителей все числа, которые будут напечатаны на экране при выполнении вызова **F(9)**. Числа должны быть записаны в том же порядке, в котором они выводятся на экран.

4. Вызов представленной ниже рекурсивной функции приводит к появлению на экране чисел и точек. С каким минимальным натуральным аргументом **a** нужно вызвать эту функцию, чтобы в результате на экране появилось 5 точек (необязательно подряд, между точками могут встречаться числа)?

```
def F(a):
    if a<1:
        return 1
    if a%3==0:
        print("...")
        p=F(a//3)+F(a//4)
    else:
        print(".")
        p=F(a//4)
    print(p)
    return 2
```

## 3.5. Символьные строки

Какие типы данных Python вы знаете?  
Какой тип используется для хранения ASCII-кодов?

- Что такое символьная строка;
- способы инициализации строк;
- операции со строками;
- что такое срез строки.

**Символьная строка** –  
Символдық жол –  
Character string  
**Срез** – Бөлік –  
Slicing

Большая часть компьютеров в мире (и наибольшее время) занята обработкой текстовой информации. Ежедневно по всему миру отправляется 18,7 млрд текстов, не считая сообщений в приложениях.

вы узнаете

ключевые слова

это интересно

Управление текстовыми документами во многих сферах деятельности является очень важной задачей. Поэтому операции со строками являются необходимым элементом программирования.

### ЗАПОМНИ

#### СИМВОЛЬНАЯ СТРОКА

– это последовательность символов, расположенных в памяти в соседних ячейках. Для работы с символами во многих языках программирования есть переменные специального типа: символы и символьные строки.

### ИНИЦИАЛИЗАЦИЯ СТРОК

Для работы с текстом в Python предусмотрен специальный строковый тип данных **str** (от англ. *string* – строка). Python создает строковые объекты, если текст поместить в одинарные или двойные кавычки (рис. 3.5.1).

```
>>> 'hello'
'hello'
>>> "Welcome"
'Welcome'
>>>
```

Рис. 3.5.1. Примеры создания строковых объектов

## 10 класс | информатика

Для инициализации строковых переменных используют операцию присваивания или функцию ввода с клавиатуры – **input**:

```
s='Привет!'
s=input('Welcome')
```

Можно создать пустую строку:

```
s=''
```

Для создания многострочной строки необходимо заключить ее в три одинарные или двойные кавычки:

```
long_string='''Это очень, очень, очень, очень, очень
длинная строка'''
```

Если нужно использовать одинарные кавычки в вашей строке, то впишите строку в двойные кавычки и наоборот.

```
s1="I'm a Python programmer!"
s2='Слово "Python" обычно подразумевает змею'
```

В строках можно использовать специальные символы, которые записываются двумя служебными символами, рассматриваемыми Python как один:

```
\n - переход на новую строку
\t - знак табуляции
\\ - наклонная черта влево
\' - символ одиночной кавычки
\" - символ двойной кавычки
```

## ОПЕРАЦИИ СО СТРОКАМИ

**ЗАПОМНИ**

Оператор '+' используется для объединения (сцепления) строк. Эта операция иногда называется **конкатенацией**.

Для строк в Python определены **операции**:  
+ – конкатенации (объединения строк);

```
>>>'Almaty'+ ' city'
'Almaty city'
```

\* – умножения (повторения строки);

```
>>>'Abc' *3
'AbcAbcAbc'
```

[ ] – индексации (выполняет доступ к символу строки).



Каждый символ строки имеет свой порядковый номер (*индекс*). Нумерация символов начинается с нуля:

```
s = 'Def Python'
print(s[0])   #'D'
print(s[2])   #'f'
print(s[5])   #'y'
```

Так же можно получить элемент строки по отрицательному индексу, при этом отсчет пойдет с конца строки:

```
s = 'Def Python'
print(s[-1])  #'n'
print(s[-4])  #'t'
```

Строки в Python, как и числа, являются **неизменяемыми**. Попытка изменить какой-либо символ в строке приведет к ошибке (*рис. 3.5.2*).

```
>>> s='Move'
>>> s[0]='L'
Traceback (most recent call last):
  File "<pyshell#12>", line 1, in <module>
    s[0]='L'
TypeError: 'str' object does not support item assignment
>>>
```

**Рис. 3.5.2.** Ошибка, возникающая при попытке изменения S[0]

Если требуется изменить строку, то следует создать новую из фрагментов старой строки.

### СРЕЗЫ

Для того чтобы выделить часть строки (**подстроку**), в языке Python применяется операция получения среза – двух индексов, разделенных двоеточием:

```
>>> s='Python programming'
>>> s[7:11]
'prog'
```

s[7:11] – срез строки **s**, начиная с индекса **7**, заканчивая индексом **11** (не включая его).

## 10 класс | информатика

Со срезами можно производить различные манипуляции:

```
>>> s[:3] # с 0 индекса по 3-й не включительно
'Pyt'
>>> s[:] # вся строка
'Python programming'
>>> s[::2]
# третий аргумент задает шаг (по умолчанию один)
'Pto rgamn'
>>> s[::-1] # "обратный" шаг
'gnimmargorp nohtyP'
>>> s[:-1] # отсчет среза с конца строки
'Python programmin'
```



**ПРИМЕР 1.** Напишем программу виртуального собеседника, которая будет имитировать диалог с пользователем (рис. 3.5.3, а, б).

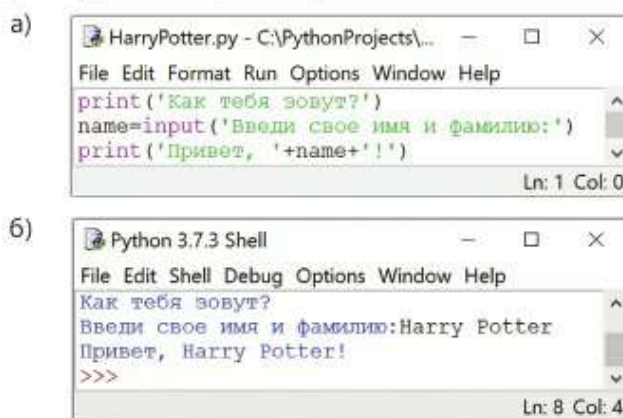


Рис. 3.5.3. Программа, имитирующая виртуального собеседника

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Добавьте в программу вопросы о месте жительства и наличии домашнего питомца.
2. Выведите на экран подходящие комментарии для возможных ответов. Например: «Это очень красивый город», «Возможно, я когда-нибудь побываю там» или «Какое милое существо!», «Не все любят животных в доме».
3. Дополните диалог новыми вопросами, проявив фантазию и креативность.

## СРАВНЕНИЕ СТРОК

К строкам можно применять операции сравнения. Оператор `==` возвращает **true**, если все символы обеих строк равны. Символы строк должны совпадать по регистру (рис. 3.5.4).

```
s1='информатика'
s2='программирование'
result=s1==s2           #False
result=s1=='Информатика' #False
result=s1=='информатика' #True
```

Рис. 3.5.4. Пример использования операции `==`

Операция `!=` возвращает **true**, если две строки не совпадают (рис. 3.5.5).

```
s1='информатика'
s2='программирование'
result=s1!=s2           #True
result=s2!='Программирование' #True
result=s2!='программирование' #False
```

Рис. 3.5.5. Пример использования операции `!=`

Результаты базовых операций сравнения `<`, `<=`, `>`, `>=` зависят от регистра и алфавитного порядка символов. Например, строка «**б**» больше строки «**а**», так как символ **б** по алфавиту идет после символа **а**. Строка «**а**» больше строки «**А**». Если первые символы строки равны, то сравниваются последующие символы (рис. 3.5.6):

```
s1='Орган'
s2='Оркестр'
result=s1>s2           #False
```

Рис. 3.5.6. Пример использования операции `>`

Условие `s1 > s2` ложно, то есть `s2` больше чем `s1`, так как при равенстве первых двух символов («**Ор**») третий символ первой строки («**г**») стоит в алфавите до третьего символа второй строки («**к**»), а код символа '**г**' меньше чем '**к**'.



**ПРИМЕР 2.** Создадим программу, которая проверяет знание пользователем предложенного стихотворения (рис. 3.5.7, а, б).

Нетрудно заметить, что если пользователь ввел с клавиатуры фразу «не помню», то программа завершает свою работу, иначе выводится следующая строка стихотворения. Досрочное завершение программы выполняется командой **exit()** из модуля **sys**.

<p>а)</p> <pre> 1  print('КОНКУРС ЧТЕЦОВ') 2  print('Продолжи стихотворение.') 3  print('Если затрудняешься, напиши: не помню') 4  print('') 5  print('У лукоморья дуб зеленый') 6  answer=input() 7  if answer=='не помню': 8      print('Не переживай. В следующий раз получится.') 9      sys.exit() 10 print('И днем и ночью кот ученый') 11 answer=input() 12 if answer=='не помню': 13     print('Не переживай. В следующий раз получится.') 14     sys.exit() </pre> <p>Ln: 15 Col: 14</p>	<p>б)</p> <pre> Python 3.7.3 Shell File Edit Shell Debug Options Window Help КОНКУРС ЧТЕЦОВ Продолжи стихотворение. Если затрудняешься, напиши: не помню  У лукоморья дуб зеленый Златая цепь на дубе том. И днем и ночью кот ученый не помню Не переживай. В следующий раз получится. &gt;&gt;&gt;   </pre> <p>Ln: 19 Col: 4</p>
---	---

Рис. 3.5.7. Пример программы, использующей операции сравнения



1. Что такое символьная строка?
2. Приведите примеры различных способов инициализации строк.
3. Можно ли обратиться к элементу строки с заданным номером? Как это сделать?
4. Какой индекс имеет первый символ строки?
5. Почему нельзя присвоить элементу строки с заданным номером новое значение?
6. Для чего применяется конкатенация строк?
7. Как выделить определенную часть строки, введенную пользователем?
8. Как изменить символы в строке?
9. Как удалить символы в строке?
10. Как вставить подстроку в заданную строку?
11. Какие значения получают строковые переменные после выполнения фрагмента программы (табл. 3.5.1)?



ТАБЛИЦА 3.5.1

```
str1='Hello, World'
str2=' '
str3=" " "Что отобразится'
в такой "строке" ? "'
str4='Это длинная\нстрока'
str5='123' * 4
```

Переменная	Значение
str1	
str2	
str3	
str4	
str5	

1. Напишите программу, имитирующую диалог пациента и доктора, выясняющего симптомы болезни.

2. Напишите программу, загадывающую пользователю загадки-обманки. Посчитайте, сколько верных ответов он дал. Примеры загадок-обманок:

1) Во дворе трещит мороз,  
Шапку ты надень на ... *голову (не нос)*.

2) Хоккеистов слышен плач  
Пропустил вратарь их ... *шайбу (не мяч)*.

3) Вышел зайчик погулять,  
Лап у зайца ровно ... *четыре (не пять)*.

3. Создайте программу, имитирующую тест Тьюринга, который предназначен для определения мыслительных способностей искусственного интеллекта. Примеры вопросов:

1) Какого цвета бурый медведь?

2) Все кошки любят рыбу. Мою кошку зовут Мурка. Что любит Мурка?

3) Еламан успокаивал Бакторе, так как он был расстроен. Кто был расстроен?

4) Большой шар проделал дыру в столе, потому что он был сделан из пенопласта. Что было сделано из пенопласта?

Искусственный интеллект пока плохо справляется с такими вопросами. Попробуйте выяснить, кем является пользователь: человеком или искусственным интеллектом?

4. Напишите программу, которая с помощью методов строковой обработки выделяет из слова «газообразование» анаграммы, например *газ, образование, образ, взор, нива, ...*



1. Какие значения получают переменные после инициализации? Аргументируйте свой ответ.

```
s1 = "s-1"
```

```
s2 = "s-2"
```

```
s3 = s1
```

1) `b = s2==s1`

2) `b = s2=="s-2"`

3) `s1 = "s1"; s2 = "s2"; b = s1 != s2`

4) `s1 = "abcd"; s2 = "de"`

```
b = s1 > s2
```

```
b = s1 < s2
```

5) `s1 = "abcd"; s2 = "ab"`

```
b = s1 >= s2
```

```
b = s1 <= s2; b = s2 >= "ab"
```

2. Опишите срезы строки `my_string='I like Python!'`, чтобы выделить следующие подстроки:

```
'I'
```

```
'I like Python'
```

```
'like'
```

```
'I like Python!'
```

```
'!nohtyP ekil I'
```

```
'Python!'
```

## 3.6. Функции и методы строковой обработки

Как задаются значения символьных строк?

Какие операции допустимы со строками?

- Функции и методы строковой обработки;
- как осуществляется преобразование из строкового типа в числовой и обратно.

**Длина** – Ұзындығы – Length

**Точечная нотация** – Нүктелік нотация – Point notation

**Подстрока** – Жоласты – Substring

вы узнаете

ключевые слова

При работе со строками, как правило, возникает необходимость выполнять их копирование, вставку, удаление или поиск.

Для работы со строками в Python предусмотрено большое число встроенных функций. Например, **len()** (от англ. *length* – длина). Данная функция вычисляет длину строки, которая передается ей в качестве параметра (рис. 3.6.1).

```
>>> s='processor'
>>> len(s)
9
```

Рис. 3.6.1. Пример использования функции **len()**

Для любого символа можно узнать его номер (код символа) с помощью функции **ord()**. И наоборот, определить символ по его числовому коду можно с помощью функции **chr()** (рис. 3.6.2).

```
>>> ord('D')
68
>>> chr(70)
'F'
>>>
```

Рис. 3.6.2. Пример определения кода символа и символа по коду

Числа могут быть преобразованы в строки с помощью функции **str()**.

Если строка является последовательностью знаков-цифр, то она может быть преобразована в целое число с помощью функции **int()**, а в вещественное – с помощью функции **float()** (рис. 3.6.3).

```
>>> str(123)
'123'
>>> int('456')
456
>>> float('3.1414926')
3.1414926
>>>
```

Рис. 3.6.3. Пример преобразования числа в строку и обратно

## ФОРМАТИРОВАНИЕ СТРОК

Форматирование в Python – мощный инструмент управления строками. Иногда возникают ситуации, когда нужно создать строку, подставив в нее некоторые данные, полученные в процессе выполнения программы (пользовательский ввод, данные из файлов и т.д.). Для этого используется метод **format()**. Например:

```
>>> 'Hello, {}!'.format('world')
'Hello, world!'
```

Строка формата содержит замещающие поля, заключенные в фигурные скобки {}. Всё, что не содержится в фигурных скобках, считается буквальным текстом, который копируется без изменений в выходные данные. А замещающие поля заполняются значениями аргументов функции **format**.

В приведенном примере для определения формата вывода строки используется запись `'Hello, {}!'.format('world')`. Это так называемая **точечная нотация**, которая означает, что метод **format** применяется к строковому объекту.

Когда в строку требуется подставить несколько значений, в форматной строке указывают необходимое количество пронумерованных замещающих полей (рис. 3.6.4).

```
a = "Python is as simple as {0}, {1}, {2}".format("a", "b", "c")
print(a) # 'Python is as simple as a, b, c'

b = "Python is as simple as {1}, {0}, {2}".format("a", "b", "c")
print(b) # 'Python is as simple as b, a, c'
```

Рис. 3.6.4. Пример использования функции **format()**

Для форматного вывода числовых данных используются спецификаторы формата **d** – для целых чисел, **f** – вещественных с фиксированной точкой и **e** – экспоненциальный формат (с плавающей точкой). Например:

```
>>> s = "{:5d}".format(123)
>>> s
'  123'
```



В результате число 123 будет записано по формату **d** (целое число) в пяти позициях, то есть в начале строки будут стоять два пробела.

```
x=54.321
s1 = "{:6.2f}".format(x) # s = " 54.32"
s2 = "{:10.2e}".format(x) # s = " 5.43e+01"
```

В **s1** будет записано значение переменной **x** в шести позициях с двумя знаками в дробной части. В **s2** – в 10 позициях с двумя знаками в дробной части.

### МЕТОДЫ СТРОКОВОЙ ОБРАБОТКИ

Методы **upper()** и **lower()** преобразуют строку соответственно в верхний и нижний регистр (рис. 3.6.5).

```
s="One Two Three"
s1=s.upper() #s1="ONE TWO THREE"
s2=s.lower() #s2="one two three"
```

Рис. 3.6.5. Пример использования методов **upper()** и **lower()**

Метод **split** (от англ. *split* – расщепить) используют для разбиения строки по заданному разделителю (по умолчанию – пробел).

```
>>> s='red blue orange white'
>>> s1=s.split()
>>> s1
['red', 'blue', 'orange', 'white']
>>> s = '/usr/local/bin'
>>> s.split('/')
['', 'usr', 'local', 'bin']
```

Метод **isdigit** определяет, содержит ли строка только цифры. Она возвращает **true**, если строка не содержит ничего, кроме цифр, и **false** – в противном случае (рис. 3.6.6).

```
s1="Computer"
print(s1.isdigit()) #False
s2="1024"
print(s2.isdigit()) #True
```

Рис. 3.6.6. Пример использования метода **isdigit()**

## 10 класс | информатика

Метод **find** используют для поиска подстроки в строке. Если подстрока найдена, то возвращается позиция первого вхождения. В противном случае возвращается **-1**. Функция **find** может вызываться с дополнительными двумя параметрами, определяющими начало и конец области поиска. Метод **rfind** (от англ. *reverse find*) используют для поиска последнего вхождения подстроки в строку (рис. 3.6.7).

```
s1="информатика"
s2="форма"
pos=s1.find(s2)           # 2
pos=s1.find("тик")       # 7
pos=s1.find("а", 7)      # 10
pos=s1.find("и", 3, 10)  # 8
pos=s2.find(s1)         # -1
pos=s1.rfind("и")       # 8
```

Рис. 3.6.7. Пример использования методов **find()** и **rfind()**

Метод **replace** заменяет в строке одну подстроку на другую:

```
>>> s='ABCDACFDA'
>>> s.replace('DA', 'NET')
'ABCNETCFNET'
```



**ПРИМЕР.** Напишем программу, которая заменит в строке все буквы **'a'** на **'b'**. Для этого определим длину строки и в цикле от первого до последнего символа будем проверять, являются ли они буквой **'a'**. Если очередной символ окажется равным **'a'**, то заменим его на **'b'** (рис. 3.6.8).

```
str1.py - C:\PythonProj...
File Edit Format Run Options Window Help
s = input( "Введите строку:" )
s1 = ""
for c in s:
    if c == "a": c = "b"
    s1=s1+ c
print ( s1 )
Ln: 1 C

Python 3.7.3 S...
File Edit Shell Debug Options
Window Help
1.py =====
Введите строку:abracadabra
bbrbcbdbbrb
>>> |
Ln: 7 Col: 4
```

Рис. 3.6.8. Программа замены в строке одного символа другим



В данном примере мы посимвольно перебираем строку. Для этого используем цикл **for** с **in s**. Он поочередно выделяет символы, входящие в строку **s**. Если очередной символ не равен **'a'**, то записываем его без изменения в результирующую строку. Иначе заменяем его на символ **'b'** и тоже добавляем в строку **res**.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Добавьте в программу команду, заменяющую **'b'** на **'a'**.
2. Что будет выведено на экран в результате выполнения фрагмента программы (рис. 3.6.9)?

```
str2.py - C:/PythonProj...
File Edit Format Run Options Window Help
s = input( "Введите строку:" )
res = ""
for c in s:
    if c == "a": c = "b"
    if c == "b": c = "a"
    res=res+ c
print ( res )
Ln: 4 Col: 24
```

Рис. 3.6.9. Фрагмент модифицированной программы замены в строке одного символа другим

3. Какие изменения нужно внести в код, чтобы результат работы программы выглядел следующим образом (рис. 3.6.10)?

```
Python 3.7.3 S...
File Edit Shell Debug Options
Window Help
2.py =====
Введите строку:abba
baab
>>>
Ln: 15 Col: 4
```

Рис. 3.6.10. Результат работы программы замены в строке одного символа другим

4. Внесите в программу изменения, учитывающие регистр символов (заменять не только строчные, но и заглавные буквы).
5. Исправьте программу так, чтобы осуществлялся поиск и замена не одного символа, а заданной подстроки.



1. Какие операции со строковыми типами вы знаете?

2. Как определить количество символов в строке?



3. Как определить, имеется ли в строке, введенной пользователем, определенный символ, и где он расположен в строке?



1. Подсчитайте, сколько раз в заданной строке встречается подстрока «abc».



2. Дан текст. Получите из этого текста новую строку, в которой отсутствуют пробелы.

3. Введите символьную строку и проверьте, является ли она *палиндромом* (словом, которое читается одинаково в обоих направлениях, например, «заказ»).

4. Зашифруйте строку, состоящую из русских букв, пробелов и знаков препинания: замените все буквы следующими за ними по алфавиту («я» на «а»), преобразуйте строчные буквы в прописные, прописные – в строчные.

5. Вводимый с клавиатуры текст переведите в последовательность точек и тире с помощью азбуки Морзе (рис. 3.6.11).

<b>А</b> --	<b>И</b> ..	<b>Р</b> ...	<b>Ш</b> ----
<b>Б</b> ----	<b>Й</b> .----	<b>С</b> ...	<b>Щ</b> ----
<b>В</b> ---	<b>К</b> -..	<b>Т</b> -	<b>Ъ</b> ----.
<b>Г</b> --.	<b>Л</b> ....	<b>У</b> ...	<b>Ы</b> ----
<b>Д</b> ...	<b>М</b> --	<b>Ф</b> ....	<b>Ь</b> ----
<b>Е</b> .	<b>Н</b> ..	<b>Х</b> ....	<b>Э</b> ....
<b>Ж</b> ....	<b>О</b> ---	<b>Ц</b> ----	<b>Ю</b> ----
<b>З</b> ---	<b>П</b> ....	<b>Ч</b> ----	<b>Я</b> ...

Рис. 3.6.11. Азбука Морзе

6. Что будет выведено на экран в результате выполнения фрагментов программы? Приведите собственные примеры входных данных и определите, что отобразится на экране.

```
str="ftp://dl.dropbox.com/u/Magick_py/py_magick3.pdf"
print(str.find('/')
```





```
print(str[4])
print(str.find('/', 8, 18))
print(str[8:18])
print(str.find('/', 8, 22))
print(str.rfind('/'))
print(str[2::2])
print(str.isdigit())
print(str[4])
print(str[4])

x=561.345
str="{:8.1f}".format(x)
print(str)
str="{:12.4e}".format(x)
print(str)
y=float(str)
print(y)
y=int(str)
print(y)
```

## 3.7.

### Файлы

Что называют файлом? Что понимать под именем файла?  
Какие типы файлов вы знаете?

- Типы файлов в программировании;
- методы открытия и закрытия файлов;
- как считать информацию из файла.

**Файл** – Файл – File

**Текстовый файл** – Мәтіндік файл – Text file

**Двоичный файл** – Екілік файл – Binary file

вы узнаете

ключевые слова

Большинство компьютерных программ работают с файлами, и поэтому возникает необходимость создавать, удалять, записывать, читать, открывать файлы.

В программировании различают файлы двух типов:

- **текстовые**, содержащие текст, разбитый на строки с помощью разделяющего символа окончания строки;

## 10 класс | информатика

– **двоичные**, содержащие данные любого формата и любой длины в двоичном виде.

Мы будем рассматривать только текстовые файлы.

При работе с файлом можно выделить следующие этапы:

- открытие файла;
- осуществление операций ввода-вывода в файл;
- закрытие файла.

### ОТКРЫТИЕ И ЗАКРЫТИЕ ФАЙЛА

Перед тем как прочесть или записать что-либо в файл, его следует открыть. Открытие файла осуществляется функцией **open**, которой нужно передать два параметра. Первый параметр имеет значение типа **str**, в котором записано имя открываемого файла. Второй – это значение типа **str**, которое равно **"r"**, если файл открывается для чтения данных (**read**), равно **"w"**, если на запись (**write**), при этом содержимое файла очищается, и **"a"** для добавления данных в конец файла (**append**). Если второй параметр не задан, то считается, что файл открывается в режиме чтения. Например:

```
my_file = open('file.txt')
fin = open('file1.txt', 'r')
fout = open('file2.txt', 'w')
fapp = open('file3.txt', 'a')
```

После того как работа с файлом закончена, важно не забывать его закрыть, чтобы освободить место в памяти. Делается это с помощью файлового метода **close()**. Например:

```
fin.close()
fout.close()
```

Работать с файлом (читать, записывать) после этого нельзя.

### ЧТЕНИЕ ИНФОРМАЦИИ ИЗ ФАЙЛА

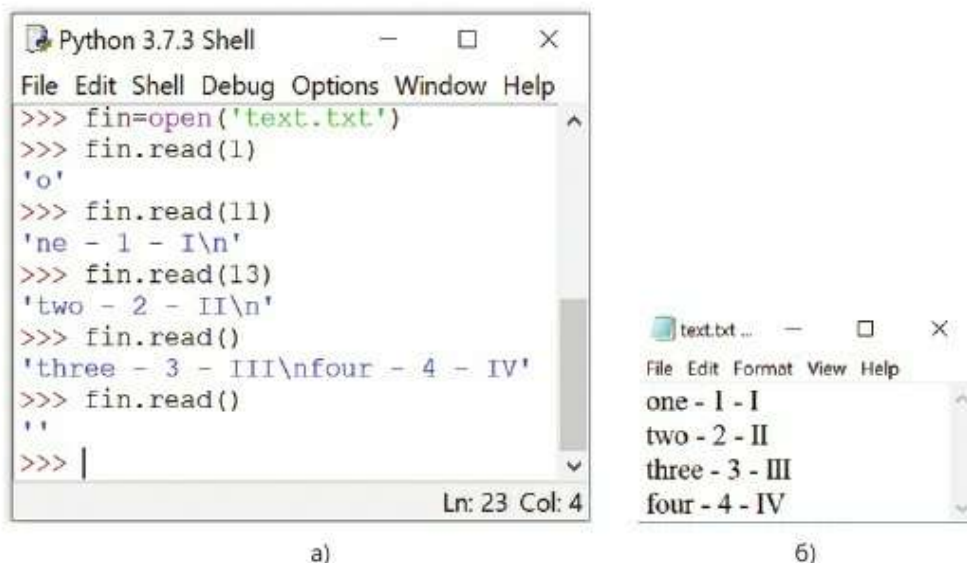
После открытия файла в режиме чтения из него можно считывать информацию. Чтение данных из файла может осуществляться разными способами:

- целиком (всё содержимое из файла);

- посимвольно (побайтно);
- построчно.

Получить все данные из файла можно с помощью файлового метода **read()**. Если методу **read()** передать целочисленный параметр, то будет считано не более заданного количества символов. Например, считывать файл побайтно можно с помощью метода **read(1)**.

**ПРИМЕР 1.** Пусть у нас имеется файл **text.txt** (рис. 3.7.1). Нужно считать из него данные.



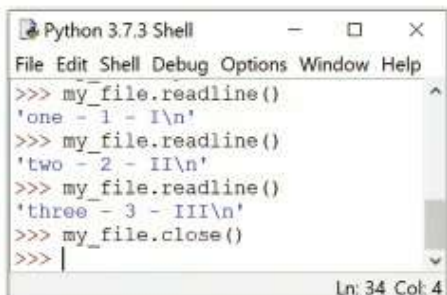
**Рис. 3.7.1.** Пример использования метода **read()**

Сначала считывается первый байт, который равен символу 'o'. Затем считанные следующие 11 байт выводят строку 'ne - 1 - I\n'. Аналогично выводим третью строку текста, состоящую из 13 символов (включая '\n'). Последующий вызов **read()** считывает весь оставшийся текст. После этого объект файлового типа **fin** становится пустым, и получить из него данные еще раз не получится. Для этого необходимо закрыть файл и открыть его снова для повторного чтения.

## 10 класс | информатика

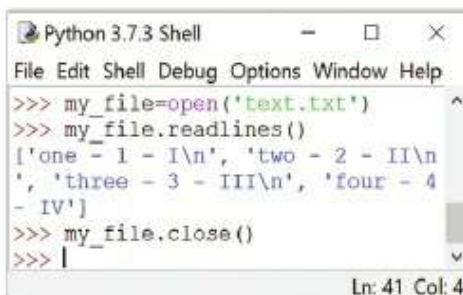
Для считывания файла построчно существует метод **readline()** (рис. 3.7.1, а).

Метод **readlines()** считывает сразу все строки и создает список (рис. 3.7.1, б).



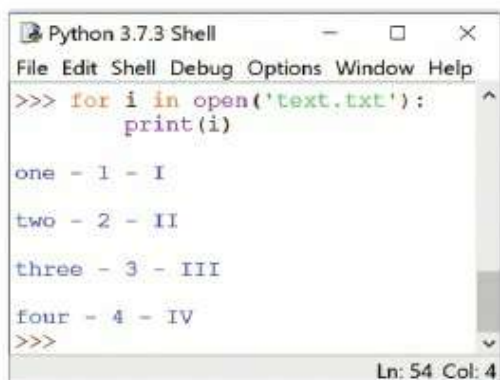
```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
>>> my_file.readline()
'one - 1 - I\n'
>>> my_file.readline()
'two - 2 - II\n'
>>> my_file.readline()
'three - 3 - III\n'
>>> my_file.close()
>>> |
Ln: 34 Col: 4
```

Рис. 3.7.1, а. Использование метода **readline()**



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
>>> my_file=open('text.txt')
>>> my_file.readlines()
['one - 1 - I\n', 'two - 2 - II\n', 'three - 3 - III\n', 'four - 4 - IV']
>>> my_file.close()
>>> |
Ln: 41 Col: 4
```

Рис. 3.7.1, б. Использование метода **readlines()**



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
>>> for i in open('text.txt'):
    print(i)
one - 1 - I
two - 2 - II
three - 3 - III
four - 4 - IV
>>>
Ln: 54 Col: 4
```

Рис. 3.7.1, в. Считывание данных в цикле без использования методов чтения

## ЗАПОМНИ

**ИТЕРИРУЕМЫЕ ОБЪЕКТЫ** предоставляют возможность поочередного прохода по своим элементам.

Объект файлового типа относится к так называемым итерируемым объектам. Поэтому считывать данные из них можно сразу в цикле без использования методов чтения.

Если нужно прочитать несколько данных из одной строки, используют метод **split()**. Он разбивает строку по пробелам (по умолчанию) или символам, которые передаются как аргументы, возвращает список выделенных фрагментов строки. Например:

```
>>> my_file=open('text.txt')
>>> blocks= my_file.readline().split()
```



```
>>> blocks
['one', '-', '1', '-', 'I']
```

В Python имеется оператор, применяя который можно упростить чтение и редактирование файлов. Оператор **with** автоматически открывает файл и, по окончании работы в нем, закрывает.

```
with open('text.txt') as my_file:
    for line in my_file:
        print(line)
```



**ПРИМЕР 2.** Определить, сколько в текстовом файле слов (рис. 3.7.2).

```
extractWords.py - C:\PythonProjects\extra...
File Edit Format Run Options Window Help
with open('data.txt') as file:
    text = file.read()
text = text.replace(".", "").replace(", ", "")
words = text.split()
print(len(words))
Ln: 8 Col: 0
```

Рис. 3.7.2. Программа подсчета количества слов

Открываем файл **'data.txt'** и связываем его с переменной **file**. Затем считываем содержимое файла в строковую переменную **text**. Метод **replace** удаляет знаки пунктуации из текста, чтобы разделителями слов остались только пробелы. Затем выделяем из текста слова с помощью метода **split**. Подсчитываем количество слов в списке **words** функцией **len**.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Добавьте в программу команду, печатающую выделенные слова.
2. Какие необходимо внести изменения в программу, если в тексте встречаются знаки препинания '?' и '!'?
3. Измените программу так, чтобы, кроме слов, в ней подсчитывались строки и буквы.

## 10 класс | информатика



1. Перечислите режимы доступа к файлу.
2. Как открыть и как закрыть файл?
3. Чем отличаются текстовые и двоичные файлы по внутреннему содержанию? Можно ли сказать, что текстовый файл – это частный случай двоичного файла?
4. Как вы думаете, почему открытый программой файл, как правило, блокируется, и другие программы не могут получить к нему доступ?
5. Что такое файловая переменная? Почему для работы с файлом используют не имя файла, а файловую переменную?
6. Может ли одна и та же файловая переменная быть использована для работы с несколькими файлами?
7. Как организуется чтение данных из файла?
8. Как организовать построчное считывание данных в файл? Приведите пример.



1. Прочитайте текст из файла и выведите только те строки, в которых есть слова, начинающиеся с буквы **A**.
2. В файле записаны данные о результатах сдачи экзамена. Каждая строка содержит фамилию, имя и количество баллов, разделенные пробелами: **<Фамилия> <Имя> <Количество баллов>**. Выведите фамилии и имена тех учеников, которые получили больше баллов.
3. Дан текст из файла, содержащий натуральные числа и слова. Напишите программу, которая формирует список из чисел, содержащихся в этой строке. Например, из строки **«mars2015 code7 2 b 39»** на выходе мы должны получить список **[2015, 7, 2, 39]**.



1. Считывает строку из файла `fin = open('file.txt')` команда:
  - A) `getline(fin)`
  - B) `read(fin)`
  - C) `fin.readline()`
  - D) `fin.read()`
2. Считывает символ из файла `fin = open('file.txt')` команда:
  - A) `fin.read(1)`
  - B) `read(fin)`
  - C) `fin.readline()`
  - D) `fin.read()`



3. Считывает всю информацию из файла `fin = open('file.txt')` команда:

- A) `readall(fin)`
- B) `fin.readline()`
- C) `read(fin)`
- D) `fin.read()`

4. Файл `data.txt` содержит текст (рис. 3.7.3).

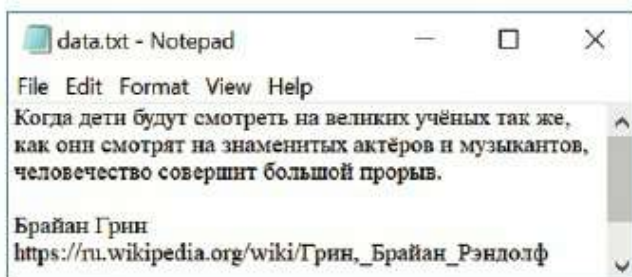


Рис. 3.7.3. Пример файла `data.txt`

Что будет выведено на экран в результате работы программы (рис. 3.7.4)? Почему?

```
filename = "data.txt"
with open(filename) as my_file:
    s = my_file.read()
    print(s)

with open(filename) as my_file:
    s = my_file.readlines()
    print(s)

with open(filename) as my_file:
    for s in my_file:
        print(s.strip())
```

Рис. 3.7.4. Программа, использующая различные способы чтения из файла

## 3.8. Работа с файлами

Какие типы файлов в Python вы знаете? Назовите методы открытия и закрытия файла. Перечислите способы чтения данных из файла.

- Способы записи информации в текстовый файл;
- способы проверки ошибок выполнения файловых операций;
- признак конца файла.

**Чтение из файла** – Файлдан оқу – Read from file

**Запись в файл** – Файлға жазу – Write to file

**Контроль ошибок** – Қатені бақылау – Error control

Поначалу словом file называли само устройство памяти, а не его содержимое. Например, диски IBM 350, использовавшиеся в машине IBM 305, назывались disk files.

вы узнаете

ключевые слова

это интересно

### ЗАПИСЬ ИНФОРМАЦИИ В ФАЙЛ

Данные выводятся в файл с помощью метода **write()**, который возвращает число записанных символов. В качестве параметра ему передается одна строка. Если файла с указанным именем в рабочем каталоге нет, то он будет создан, если файл с таким именем существует, то он будет **перезаписан**. Этот метод не выводит символ конца строки **'\n'** (как это делает функция **print** при стандартном выводе), поэтому для перехода на новую строку в файле необходимо явно вывести символ **'\n'**. Например:

```
>>> f1 = open('text1.txt', 'w')
>>> f1.write('Hello world!\n')
13
>>> f1.write('1 2 3\n')
6
>>> f1.close()
```

Файл, открытый на запись, нельзя прочитать. Для этого требуется его закрыть, а потом открыть на чтение.

Запись в файл можно осуществлять, используя определенный шаблон вывода. Например, запись в файл результата сложения двух чисел:

```
f1.write('{:d} + {:d} = {:d}\n'.format(x, y, x+y))
```



В данном примере вместо шаблонов `{:d}` последовательно подставляются значения параметров метода `format` (сначала `x`, затем `y`, и `x + y`).

С помощью метода `writelines()` можно записать в файл итерируемую последовательность (например, список строк или чисел).

```
>>> list = [0,1,2,3,4]
>>> f2 = open('text2.txt', 'w')
>>> f2.writelines('%s\n' %i for i in list)
>>> f2.close()
>>> open('text2.txt').read()
'0\n1\n2\n3\n4\n'
```

### ПРОВЕРКА ДОСТИЖЕНИЯ КОНЦА ФАЙЛА ПРИ ОПЕРАЦИЯХ ВВОДА

Часто при работе с файлами заранее неизвестно, сколько и каких значений хранится в текстовом файле. Приходится следить за тем, чтобы не пропустить момент достижения конца файла. В случае обнаружения конца файла методы `read()` и `readline()` возвращают пустую строку. Рассмотрим данный прием на примере построчного чтения текстового файла.

```
with open('file.txt') as my_file:
    while True:
        line = my_file.readline()
        if not line: break
        print(line)
```

В цикле `while` считываем строки, пока не будет достигнут конец файла. При получении переменной `line` пустой строки цикл чтения заканчивается с помощью оператора `break`.

### ПРОВЕРКА ОШИБОК ВЫПОЛНЕНИЯ ФАЙЛОВЫХ ОПЕРАЦИЙ

Файловые операции, например открытие и закрытие файлов, известны как один из наиболее вероятных источников ошибок. Для повышения надежности программ обычно выполняется проверка, успешно или нет завершилась файловая операция. После того, как предпринималась попытка открыть файл, необходимо убедиться в том, что файл открыт и готов для

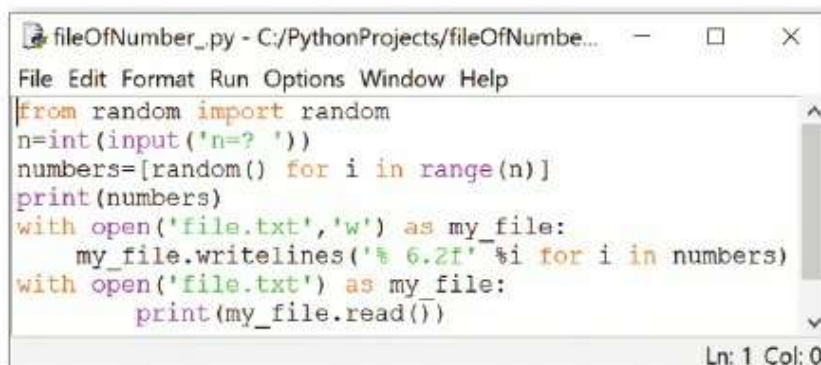
## 10 класс | информатика

чтения или записи. Диск может заполниться, пользователь может удалить используемый файл во время записи, файл могут переместить и т.д. Эти типы ошибок можно перехватить с помощью обработки исключений.

```
# Ошибка при открытии файла
try:
    with open('file.txt') as my_file:
        print(my_file.read())
except: # выполняется в случае ошибки
    print('Error opening file')
```

В данном примере мы помещаем обычный код в конструкции **try/except**. Если возникнет ошибка, выведем на экран сообщение.

**ПРИМЕР 1.** Создать текстовый файл **file.txt** и записать в него **n** вещественных чисел (рис. 3.8.1).



```
fileOfNumber_py - C:/PythonProjects/fileOfNumbe...
File Edit Format Run Options Window Help
from random import random
n=int(input('n=? '))
numbers=[random() for i in range(n)]
print(numbers)
with open('file.txt','w') as my_file:
    my_file.writelines('% 6.2f' %i for i in numbers)
with open('file.txt') as my_file:
    print(my_file.read())
Ln: 1 Col: 0
```

Рис. 3.8.1. Текст программы создания текстового файла

В результате работы программы будет создан текстовый файл **file.txt**, который можно просмотреть средствами обычного текстового редактора (рис. 3.8.2).

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Замените первую строку программы командой **from random import uniform**, а в третьей строке замените **random()** на **uniform(0,100)**. Что будет выведено в файл? Почему?

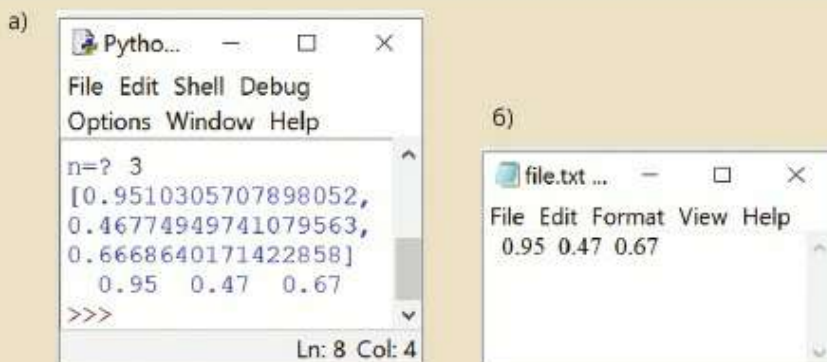


Рис. 3.8.2. Консольный и файловый вывод результата работы программы

2. Измените программу так, чтобы числа в файл записывались в столбик.
3. Какие нужно внести изменения в программу, чтобы вводились не случайные числа, а заданные пользователем?

**ПРИМЕР 2.** В текстовом файле **file.txt** хранятся вещественные числа, вывести их на экран и вычислить их сумму (рис. 3.8.3).

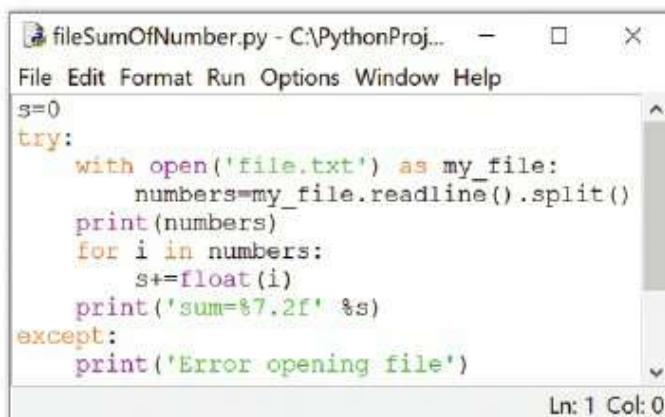
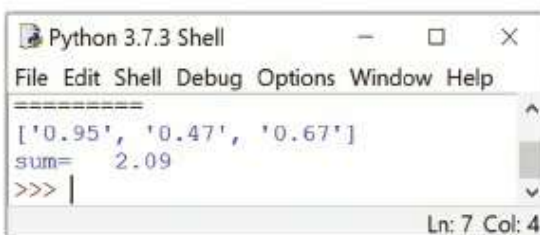


Рис. 3.8.3. Программа чтения чисел из файла

Помещаем программный код в конструкции **try/except**. Открываем файл **my\_file** в режиме чтения. Если открытие файла прошло некоррект-

но, то выводится сообщение об отсутствии такого файла, и программа завершает работу. Считываем из **my\_file** строку и формируем из нее список. Элементами списка будут подстроки, содержащие вещественные числа. В цикле перебираем элементы списка. Преобразуем их в вещественный формат и суммируем. Результат работы программы представлен на *рис. 3.8.4*.



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
=====
['0.95', '0.47', '0.67']
sum= 2.09
>>> |
Ln: 7 Col: 4
```

**Рис. 3.8.4.** Результат работы программы



1. Что представляет собой файл данных? Дайте определение.
2. Приведите пример стандартного потока ввода-вывода.
3. Какие возможности для работы с файлами предоставляет Python?
4. Как организуется чтение файла?
5. Как определить, что данные в файле исчерпаны?
6. В каких случаях нужно знать максимальное количество данных в файле, а в каких – нет?
7. Когда требуется открывать одновременно несколько файлов?



1. Считайте текст из файла и сохраните в другой файл только те строки, в которых есть слова, начинающиеся с буквы **A**.



2. Дан текстовый файл. Перепишите в другой файл все его строки, заменив в них символ «0» на символ «1» и наоборот.
3. В файле записаны целые числа. Найдите максимальное и минимальное число и запишите в другой файл.
4. Считайте текст из файла, заменив везде слово «самокат» на слово «самолет», и запишите в другой файл.



1. Фрагментом программы, записывающей строку 'Python' в файл 'file.txt, является:





- A)** `fin = open('file.txt','w')`  
`fin.write('Python')`  
`close(fin)`
- B)** `fout = open('file.txt')`  
`fout.write('Python')`  
`close(fin)`
- D)** `fout = open('file.txt','r')`  
`fout.writelines('Python')`  
`close(fout)`
- C)** `in = open('file.txt','w')`  
`fin.writelines('%s\n' %i for i in 'Python')`  
`close(fin)`

2. Фрагментом программы, записывающей строку список [1, 2, 3] в файл 'file.txt, является:

- A)** `fin = open('file.txt','w')`  
`fin.write([1, 2, 3])`  
`close(fin)`
- B)** `fout = open('file.txt')`  
`fout.write([1, 2, 3])`  
`close(fin)`
- D)** `fout = open('file.txt','w')`  
`fout.writelines([1, 2, 3])`  
`close(fout)`
- C)** `fin = open('file.txt','w')`  
`fin.writelines('%s\n' %i for i in [1, 2, 3])`  
`close(fin)`

3. В заданном файле целых чисел посчитайте количество компонент, кратных трем, и запишите результат во второй файл. Исправьте ошибки в представленном программном коде (рис. 3.8.5). Добавьте проверку на корректность выполнения файловых операций (try/except).

```
f1=open('file2.txt','w')
with open('file1.txt') as f2:
    while True:
        st = f1.readline().replace('\n','')
        print(st)
        if not st: break
        lst=st.split()
        for i in lst:
            if i%3==0:
                f2.write(i+' ')
```

Рис. 3.8.5. Программа файлового ввода/вывода

## 3.9. Методы сортировки

Какой тип данных называется массивом?

Как описываются массивы в Python?

Перечислите способы инициализации массивов.

– Что такое сортировка;  
– классификацию методов сортировки;  
– простые методы сортировки.

**Сортировка** –  
Сұрыптау – Sorting  
**Метод выбора** –  
Таңдау әдісі –  
Selection method  
**Пузырьковая сортировка** –  
Көпіршікті сұрыптау –  
Bubble Sort

Первые прототипы современных методов сортировки появились в XIX веке. К 1890 году для ускорения обработки данных переписи населения в США была создана сортировальная машина Холлерита, основанная на методах поразрядной сортировки.

вы узнаете

ключевые слова

это интересно

Во многих задачах бывает необходимо переставить элементы данных таким образом, чтобы они располагались в порядке возрастания или убывания. Такие данные называются **упорядоченными**, а процесс их получения – **сортировкой**. Сортировка используется при обработке больших объемов информации. Большинство процессов обработки данных выполняется проще, если данные упорядочены. Упорядоченные данные содержатся в телефонных книгах, оглавлениях, библиотеках, словарях, на складах, т.е. везде, где осуществляется поиск нужной информации.

Методы сортировки классифицируются на **внутренние** (когда данные размещаются в оперативной памяти) и **внешние** (когда данные размещены на внешних носителях). Внешняя сортировка применяется тогда, когда в работу вовлекается гораздо больше элементов, чем можно поместить в оперативную память.

### ЗАПОМНИ

Если в массиве данных есть одинаковые элементы, то говорят о сортировке по **НЕУБЫВАНИЮ** или по **НЕВОЗРАСТАНИЮ**.

Существует множество методов сортировки. Цель разработки различных способов заключается в том, чтобы добиться минимального числа сравнений и перемещений элементов. Методы сортировки делятся на **простые**, но медленно работающие, и **улучшенные** – более сложные, но эффективные.

## ПРОСТЫЕ СОРТИРОВКИ

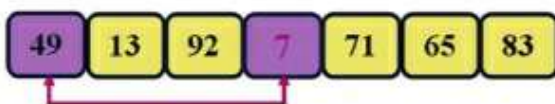
В простых сортировках при упорядочивании массива, состоящего из  $N$  компонент, выполняется  $C \cdot N^2$  действий, где  $C$  – некоторая константа.

## СОРТИРОВКА МЕТОДОМ ВЫБОРА

В сортируемой последовательности ищем минимальный элемент, который меняем местами с элементом в начале массива. Таким образом, первый элемент последовательности отсортирован. Оставшуюся часть массива рассматриваем как самостоятельный массив, в котором также выполним поиск наименьшего элемента и его обмен с первым элементом укороченного массива. Будем повторять эти действия до тех пор, пока массив не уменьшится до одного элемента.

**ПРИМЕР 1.** Отсортировать массив **49, 13, 92, 7, 71, 65, 83** методом выбора.

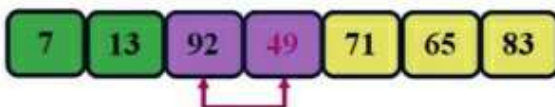
Наименьший элемент **7** меняем местами с первым.



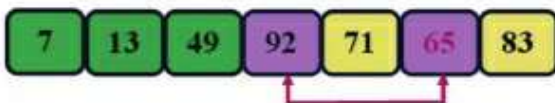
Первый элемент последовательности отсортирован.



В оставшемся неотсортированном массиве наименьший элемент **13**. Он уже стоит на первом месте укороченного массива, и перестановка элементов не осуществляется.



Минимальный элемент **49** меняем местами с первым элементом неотсортированной части массива **92**.



## 10 класс | информатика

Продолжаем аналогичные действия до тех пор, пока в неотсортированной части массива не останется один элемент.



Если в массиве  $N$  элементов, то нам придется  $N - 1$  раз находить номер  $nMin$  минимального элемента из  $A[i] \dots A[N]$  (неотсортированной части массива). Если  $i \neq nMin$ , то меняем местами элементы  $A[i]$  и  $A[nMin]$ . Поскольку поиск минимального элемента также выполняется в цикле, то алгоритм сортировки представляет собой вложенные циклы. Программа сортировки массива методом выбора представлена на рис. 3.9.1.

```

sortvibor2.py - C:\PythonProjects\sortvibor2...
File Edit Format Run Options Window Help
def sel_sort(array):
    for i in range(len(array) - 1):
        m = i
        j = i + 1
        while j < len(array):
            if array[j] < array[m]:
                m = j
                j = j + 1
        array[i], array[m] = array[m], array[i]

A = [49, 13, 92, 7, 71, 65, 83]
print(A)
sel_sort(A)
print('-----')
print(A)

```

```

Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
=====
[49, 13, 92, 7, 71, 65, 83]
-----
[7, 13, 49, 65, 71, 83, 92]
>>> |
Ln: 8 Col: 4

```

Рис. 3.9.1. Программа сортировки массива методом выбора

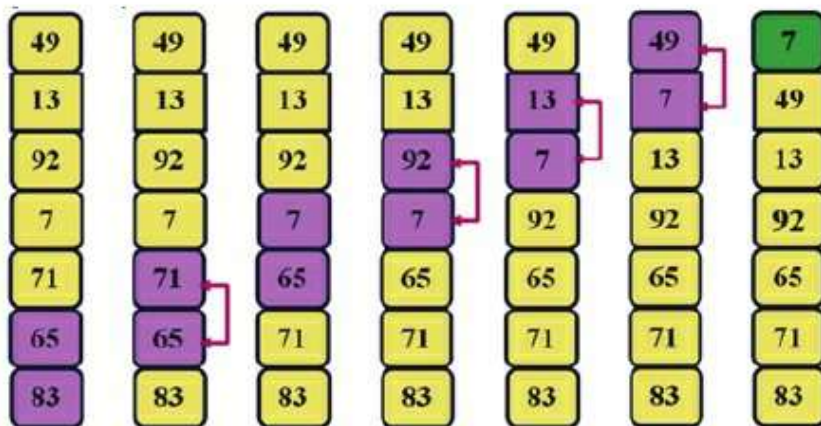


### ПУЗЫРЬКОВАЯ СОРТИРОВКА (СОСОРТИРОВКА ОБМЕНАМИ)

Алгоритм сортировки прямым обменом основан на принципе сравнения и обмена пары соседних элементов до тех пор, пока не будут отсортированы все элементы. Если рассматривать массивы как вертикальные, а не горизонтальные построения, то элементы можно представить пузырьками в сосуде с водой, причем вес каждого соответствует его значению. В этом случае при каждом проходе один пузырек как бы поднимается до уровня, соответствующего его весу. Поэтому данный метод получил название «пузырьковая сортировка».

**ПРИМЕР 2.** Отсортировать массив **49, 13, 92, 7, 71, 65, 83** методом пузырьковой сортировки.

Вначале сравниваем последний элемент с предпоследним. Если нижний элемент оказался меньше верхнего, то меняем их местами. Продолжаем таким же образом и рассматриваем следующую пару элементов и т.д. (рис. 3.9.2).



**Рис. 3.9.2.** Результат первого прохода алгоритма пузырьковой сортировки

Первая пара – **83** и **65** – расположена в правильном порядке. Изменений в массив не вносим. Во второй паре **65 < 71**. Меняем их местами. Третья пара – **65** и **7** – тоже расположена правильно. Четвертой паре (**7 < 92**) необходим обмен. Также меняем местами **7** и **13**, а затем – **7** и **49** (рис. 3.9.3).

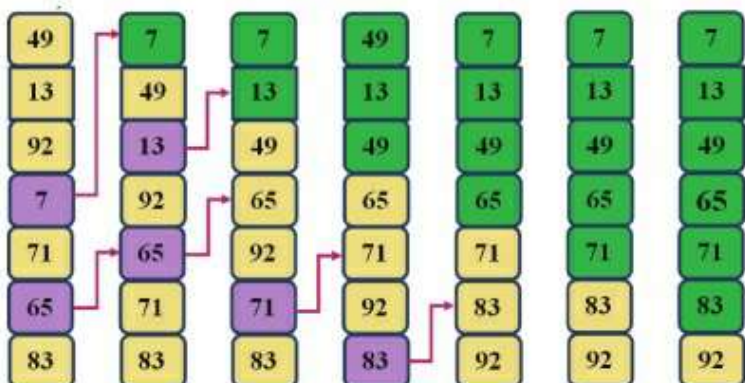


Рис. 3.9.3. Демонстрация перестановок алгоритма пузырьковой сортировки

В результате первого прохода алгоритма элемент **7** поднимается на самый верх, так как является наименьшим элементом массива. Это значит, что на следующих этапах его можно не рассматривать. Повторим ту же последовательность шагов для массива из  $N - 1$ ,  $N - 2$ ,  $N - 3$ , ...,  $2$  элементов. Рассмотрим программу, реализующую алгоритм пузырьковой сортировки (рис. 3.9.4).

```
N=7
A = [49, 13, 92, 7, 71, 65, 83]
print (A)
for i in range(N-1):
    for j in range(N-2,i-1,-1):
        if A[j] > A[j+1]:
            buff = A[j]
            A[j] = A[j+1]
            A[j+1] = buff
print (A)
```

Рис. 3.9.4. Программа алгоритма пузырьковой сортировки

В цикле **for j in range(N-2,i-1,-1)** для всех элементов от предпоследнего до  $i$ -ого выполняется проверка условия. Если текущий элемент **A[j]** больше следующего **A[j+1]**, то меняем их местами. За один проход цикла упорядочивается один элемент. Значит, таких циклов нужно сделать  $N - 1$ , т.е. на  $1$  меньше, чем количество элементов массива. Ведь если  $N - 1$  элементов поставлены на свои места, то оставшийся автоматически встает на свое

место. Поэтому полный алгоритм сортировки представляет собой вложенный цикл.



**ПРИМЕР 3.** Написать программу, которая выполняет пузырьковую сортировку массива с помощью пользовательской функции **bubble** (рис. 3.9.5, а, б).

```

bubble2.py - C:\PythonProject...
File Edit Format Run Options Window Help
def bubble(array,N):
    for i in range(N-2):
        swapped=False
        for j in range(N-2,i-1,-1):
            if array[j] > array[j+1]:
                buff = array[j]
                array[j] = array[j+1]
                array[j+1] = buff
                swapped=True
        if not swapped: break
A = [49, 13, 92, 7, 71, 65, 83]
print(A)
bubble(A,7)
print(A)
Ln: 11 Col: 0

```

а)

```

Python 3.7.3 Shell
File Edit Shell Debug Options Window
Help
C:\PythonProjects\bubble2.py
=====
[49, 13, 92, 7, 71, 65, 83]
[7, 13, 49, 65, 71, 83, 92]
>>>
Ln: 7 Col: 4

```

б)

**Рис. 3.9.5.** Программа пузырьковой сортировки (двумя файлами)

**bubble (array, N)** – это функция, которая принимает целочисленный массив и его размер, а затем выполняет пузырьковую сортировку элементов массива.

**bubble (A, 7)** – вызов функции с передачей ей массива **A** и его размера **7** в качестве параметров.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Создайте текстовый файл **array.txt**, содержащий произвольное количество целых чисел.
2. Замените консольный ввод элементов массива файловым.
3. Метод пузырьковой сортировки можно оптимизировать. На рисунке 3.9.3 видно, что два последних прохода не изменили порядок элементов, поскольку массив уже отсортирован. Очевидный способ улучшить данный алгоритм – это запоминать, производился ли на данном проходе обмен между элементами. Добавим в функцию **bubble** локальную логическую переменную **swapped**.



В начале каждого прохода алгоритма присвоим ей значение **false**, которое будет меняться на **true**, если производился обмен между элементами массива. Как только при очередном просмотре не происходит ни одного обмена между элементами **not swapped**, это означает, что массив отсортирован (рис. 3.9.6).

```
def bubble(array, N):
    for i in range(N-2):
        swapped=False
        for j in range(N-2, i-1, -1):
            if array[j] > array[j+1]:
                buff = array[j]
                array[j] = array[j+1]
                array[j+1] = buff
                swapped=True
        if not swapped: break
```

Рис. 3.9.6. Оптимизированная функция пузырьковой сортировки

4. Добавьте в функцию **bubble** консольный вывод элементов массива после каждого прохода алгоритма.



1. Объясните процесс сортировки.
2. В чем заключается идея сортировки методом выбора?
3. Почему метод пузырька получил такое название?
4. Каким образом можно оптимизировать пузырьковую сортировку?
5. Как нужно изменить приведенные алгоритмы, чтобы элементы последовательности были отсортированы по убыванию?



1. Создайте программу, сортирующую массив и определяющую количество различных чисел в нем.
2. Напишите программу, в которой пузырьковая сортировка выполняется с начала массива, т.е. наибольший элемент опускается в конец массива.
3. Напишите программу, которая выполняет неполную сортировку массива: ставит в начало массива три самых меньших по величине элемента в порядке возрастания (неубывания). Положение остальных элементов неважно.





4. Напишите программу сортировки массива по возрастанию последней цифры числа.
5. Напишите программу сортировки массива по убыванию суммы цифр числа.
6. Напишите программу сортировки массива, располагающей элементы первой половины массива по возрастанию, а второй – по убыванию.
7. Создайте программу, которая сортирует массив, а затем находит максимальное из чисел, встречающееся в массиве несколько раз.
8. Составьте программу следующего метода сортировки: сравниваются два соседних элемента  $A[i]$  и  $A[i + 1]$ . Если  $A[i] \leq A[i + 1]$ , то сдвигаются на один элемент вперед. Если  $A[i] > A[i + 1]$ , то осуществляется перестановка и сдвигаются на один элемент назад, если это возможно. Выполнять данные действия до тех пор, пока не будет пройден весь массив.



1. Объясните, для чего используется вложенный цикл в приведенных методах сортировки.
2. Сравните метод выбора и метод пузырька. В каком из них выполняется меньше перестановок?
3. Добавьте в представленную на рисунке 3.9.7 блок-схему команды оптимизированного варианта пузырьковой сортировки.

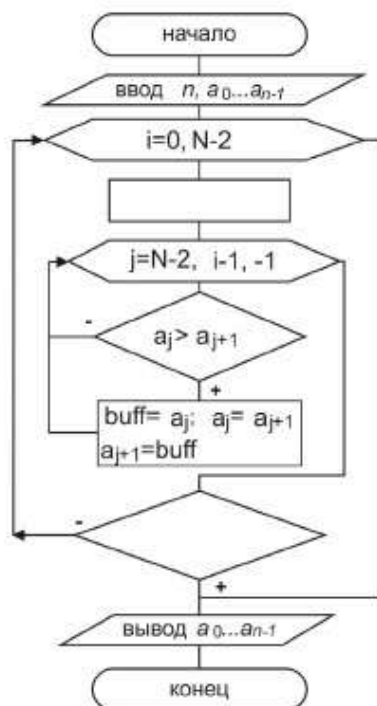


Рис. 3.9.7. Блок-схема алгоритма пузырьковой сортировки

## 3.10. Алгоритмы на графах

Что такое сортировка? Какие виды сортировок вы знаете? Какие сортировки называют простыми?

– Что такое граф;  
– основные понятия теории графов;  
– алгоритмы на графах.

**Граф** – Граф – Graph  
**Узел (вершина) графа** – Графтің түйіні (төбесі) – Node (vertex) of the graph  
**Ребро графа** – Граф қабырғасы – Graph edge

Первая работа по теории графов принадлежит Леонарду Эйлеру (1736 г.), хотя термин «граф» впервые ввел в 1936 году венгерский математик Денеш Кениг.

вы узнаете

ключевые слова

это интересно

### ЗАПОМНИ

**ГРАФ** (от греч. *grapho* – пишу) – это абстрактное представление множества объектов и связей между ними.

Теория графов является простым и мощным инструментом построения моделей и решения задач упорядочения объектов. На данный момент существует множество задач, где требуется построить некоторые сложные системы с помощью определенного

расположения их элементов. К ним относятся задачи теории сетевого планирования и управления, проблемы построения систем связи и исследования процессов передачи информации, выбор оптимальных маршрутов и потоков в сетях, календарное планирование промышленного производства, тактические и логические задачи, методы построения электрических сетей, задачи идентификации в органической химии и большой круг экономических задач.

**Граф** – это набор **узлов (вершин)** и связей между ними (**ребер**). Каждое ребро соединяет две вершины.

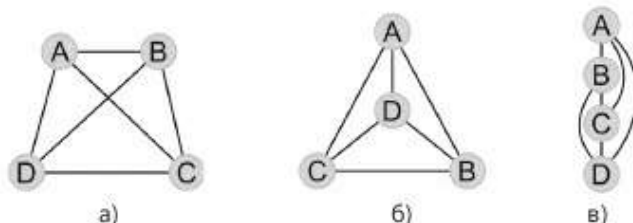


Рис. 3.10.1. Различные изображения одного и того же графа

Один и тот же граф может иметь различное изображение (рис. 3.10.1, а, б, в).

Две вершины, соединенные ребром, называют **смежными вершинами**. Вершины **A** и **B** на рис. 3.10.2 являются смежными, а **C** и **E** – нет. Количество ребер, исходящих из вершины, называют **степенью вершины**. Степень вершины **C** равна 2, а **D** – 3. Если какие-то две вершины соединены более чем одним ребром, то говорят, что граф содержит **кратные ребра** (ребра **BE**). Если ребро соединяет вершину саму с собой, то такое ребро называют **петлей (A)**. **Простой граф** не содержит петель и кратных ребер.

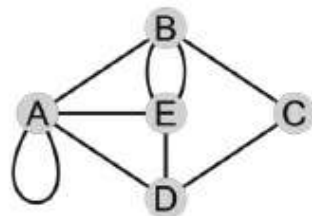


Рис. 3.10.2. Основные элементы графа

**Путем** на графе называется последовательность ребер, в которой конец одного ребра является началом следующего ребра **((A,B),(B,E),(E,D))**. Начало первого ребра называется началом пути, конец последнего ребра – концом пути. Если начало и конец пути совпадают, то такой путь называется **циклом ((A,B),(B,E),(E,D),(D,A))**.

Граф называют **ориентированным (орграфом)** (рис. 3.10.3, а), если ребра графа имеют направление (в этом случае они называются **дугами**), иначе граф – **неориентированный** (рис. 3.10.3, б).

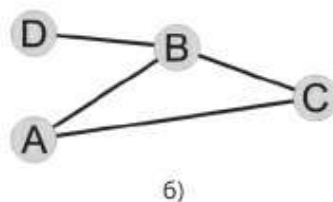
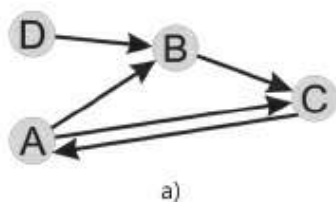


Рис. 3.10.3. Пример ориентированного и неориентированного графа

Граф называется **связным**, если между любыми двумя его вершинами есть путь. Дерево – это частный случай связного графа.

Очень часто рассматриваются графы, в которых каждому ребру приписана некоторая числовая характеристика – **вес**.

## ЗАПОМНИ

Граф называется **ДЕРЕВОМ**, если он связный и не имеет замкнутых путей – **ЦИКЛОВ**.

## 10 класс | информатика

Вес может означать длину дороги или стоимость проезда по данному маршруту. Соответствующие графы называются **взвешенными** (рис. 3.10.4).

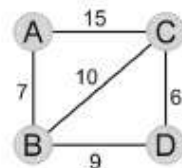
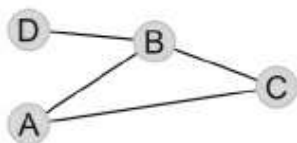


Рис. 3.10.4. Пример взвешенного графа

## СПОСОБЫ ОПИСАНИЯ ГРАФА

Выбор подходящей структуры данных для представления графа имеет принципиальное значение при разработке эффективных алгоритмов. При решении задач сведения об узлах и связях графа обычно записывают в виде специальной таблицы – **матрицы смежности** (рис. 3.10.5).

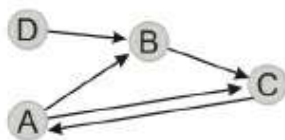


	A	B	C	D
A	0	1	1	0
B	1	0	1	1
C	1	1	0	0
D	0	1	0	0

Рис. 3.10.5. Пример графа и его матрицы смежности

Если между узлами **A** и **B** есть связь, то на пересечении строки **A** и столбца **B** записывают единицу. Ноль обозначает отсутствие связи. Матрица смежности симметрична относительно главной диагонали. В случае, когда граф содержит **петлю**, на главной диагонали будет записана единица.

Матрица смежности **ориентированного** графа не всегда симметричная. Если существует дуга из вершины **A** в вершину **B**, то на пересечении строки **A** и столбца **B** записывается единица. При отсутствии обратной дуги из **B** в **A** на пересечении строки **B** и столбца **A** записывается ноль (рис. 3.10.6).



	A	B	C	D
A	0	1	1	0
B	0	0	1	0
C	1	0	0	0
D	0	1	0	0

Рис. 3.10.6. Пример орграфа и его матрицы смежности



Информация о взвешенном графе хранится в виде **весовой матрицы**, содержащей веса ребер (рис. 3.10.7).

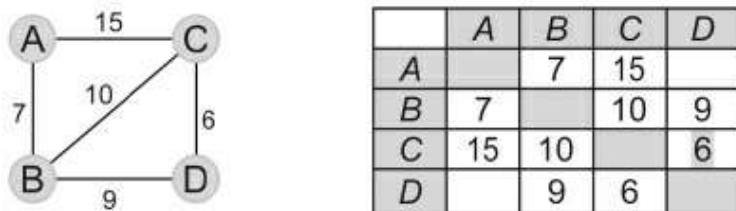


Рис. 3.10.7. Пример весовой матрицы смежности

В этом случае при отсутствии ребра можно хранить специальное значение, например **0**, **-1**. Во многих задачах удобно при отсутствии ребра хранить очень большое число ( $\infty$ ). В этом случае отсутствие ребра аналогично наличию ребра очень большой стоимости.

### ЗАДАЧА О КРАТЧАЙШЕМ ПУТИ

Задача о кратчайшем пути заключается в поиске самого короткого пути между двумя точками (вершинами) на графе. При этом сумма весов ребер найденного пути должна быть минимальна.

**ПРИМЕР 1.** Рассмотрим сеть автомобильных дорог, соединяющих города области. Числа на схеме (рис. 3.10.8) обозначают расстояния. Нужно найти кратчайший маршрут из города **A** в город **E**.

Обычно в таких случаях выбирают кратчайший маршрут до ближайшего города, в котором еще не были. В нашем случае в город **B** (длина 7), далее в **C** (длина 10), затем в **F** (длина 2) и следующий город **E** (длина 9). Длина найденного маршрута равна 28.

Алгоритм, который мы применили, называется «жадным». Это алгоритм, который на каждом шагу делает локально наилучший выбор в надежде, что итоговое решение будет оптимальным. «Жадный» алгоритм не всегда его находит.

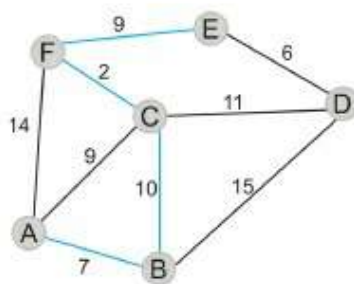


Рис. 3.10.8. Схема дорог между несколькими городами

## 10 класс | информатика

В нашем примере найденный путь не является оптимальным. Длина маршрута **A-C-F-E** равна 20, и он будет оптимальным.

Однако есть задачи, в которых «жадный» алгоритм всегда приводит к правильному решению. Вот одна из таких задач (ее называют **задачей Прима – Крускала** в честь Р. Прима и Д. Крускала, которые независимо предложили ее в середине XX века).

**ПРИМЕР 2.** Есть **N** городов, которые нужно соединить телефонной сетью с минимальными затратами. Затраты определяются длиной кабеля. Известно расстояние между городами. Требуется построить телефонную линию так, чтобы не было дублирования и длина была минимальной.

В теории графов задача Прима – Крускала имеет следующую формулировку: «Дан полный граф, состоящий из **N** вершин, и указаны длины его ребер. Найти остовное дерево минимальной длины (т.е. дерева, связывающего все вершины)».

Рассмотрим «жадный» алгоритм решения этой задачи:

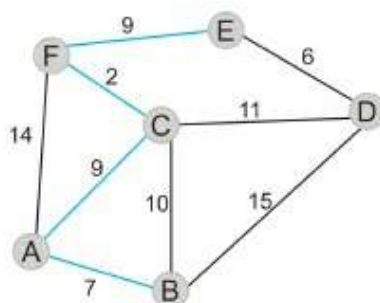
- 1) строятся все вершины для графа без ребер;
- 2) выбирается самое короткое ребро;
- 3) к дереву добавляется ребро минимального веса, которое еще не входит в дерево.

На *рис. 3.10.9* приведен пример нахождения минимального остовного дерева.

При решении этой задачи необходимо следить, чтобы новое ребро не образовывало цикла со старыми. Для этого до построения дерева окрасим каждую вершину **i** в отличный от других цвет **i**.

```
col = [i for i in range(N)]
```

Выберем очередное ребро минимальной длины (**iMin**, **jMin**), где **iMin** и **jMinj** имеют разные цвета. При этом вершина **jMinj** и все окрашенные в ее цвет (т.е. ранее с ней соединенные) вершины перекрашиваются в цвет **iMin**. Выбирая вершины разного цвета, мы исключаем возможность образования циклов. После выбора **N – 1** ребер все вершины получают один цвет.



**Рис. 3.10.9.** Пример минимального остовного дерева

Напишем программу, которая находит минимальное остовное дерево (рис. 3.10.10).

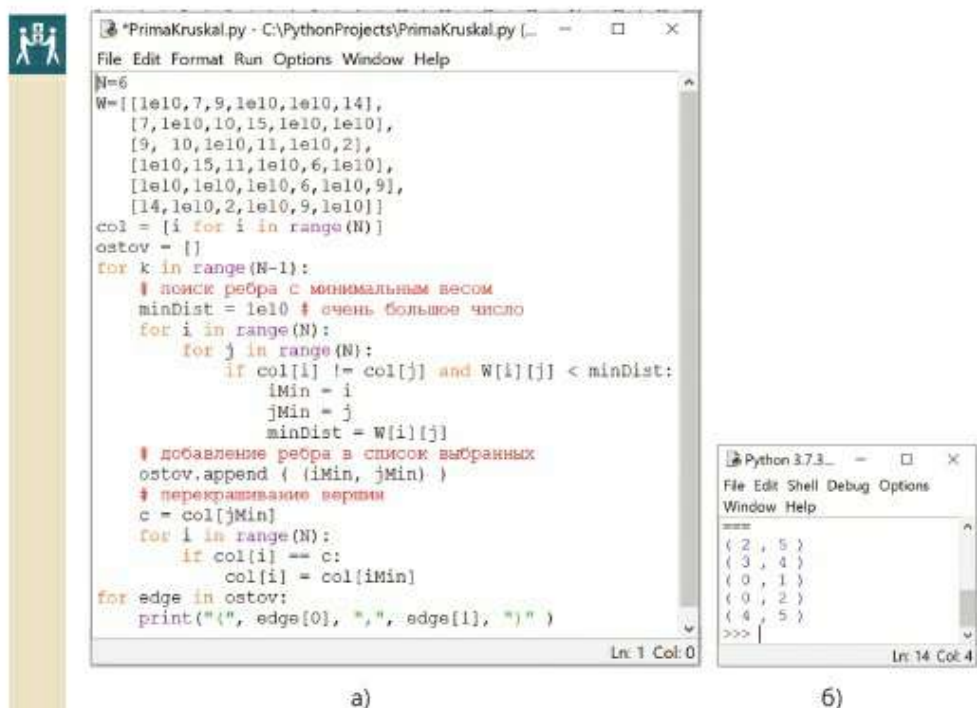


Рис. 3.10.10. Программа нахождения минимального остовного дерева

Здесь  $W$  – весовая матрица размера  $N \times N$ ;  $ostov$  – целочисленный массив из  $N - 1$  строк двух столбцов для хранения выбранных ребер (для каждого ребра хранятся номера двух вершин, которые оно соединяет). После окончания цикла остается вывести результат – ребра из массива  $ostov$ .

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Выполните программу для заданной матрицы смежности (рис. 3.10.11). Какие для этого нужно внести изменения в программу?

	A	B	C	D	E
A		2	10	8	16
B	2		9	1	
C	10	9		3	4
D	8	1	3		11
E	16		4	11	

Рис. 3.10.11. Весовая матрица

2. Измените программу так, чтобы массив заполнялся значениями из файла.



1. Что такое граф?
2. Как обычно задаются связи узлов в графах?
3. Что такое матрица смежности?
4. Что такое петля? Как определить ее в матрице смежности?
5. Что такое путь?
6. Какой граф называется связным?
7. Что такое орграф?
8. Как по матрице смежности отличить орграф от неориентированного графа?
9. Что такое взвешенный граф? Как может храниться в памяти информация о нем?
10. Что такое «жадный» алгоритм? Всегда ли он позволяет найти лучшее решение?



1. Напишите программу, которая строит минимальное остовное дерево графа, заданного весовой матрицей. Матрица вводится из файла.
2. Напишите программу, которая определяет кратчайший маршрут на графе, заданном весовой матрицей. Номера начальной и конечной вершин вводятся с клавиатуры, а весовая матрица графа вводится из файла.





1. Проанализируйте таблицу на рис. 3.10.12. Как называют такую таблицу? Какие особенности в таблице вы заметили?

	A	B	C	D	E
A		2	10	8	16
B	2		9	1	
C	10	9		3	4
D	8	1	3		11
E	16		4	11	

Рис. 3.10.12. Таблица данных о графе

2. Населенные пункты **A, B, C, D, E, F** связывают дороги, их длина (в километрах) приведена в таблице на рис. 3.10.13.

	A	B	C	D	E	F
A		3	5			15
B	3		3			
C	5	3		5	2	
D			5			3
E			2			7
F	15			3	7	

Рис. 3.10.13. Матрица смежности

Вычислите длину кратчайшего пути между пунктами **A** и **F**. (Передвигаться можно только по дорогам, указанным в таблице.)

1) 9. 2) 11. 3) 13. 4) 15.

## ТЕСТОВЫЕ ЗАДАНИЯ

## «АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ»



Знание



Понимание



Применение



Анализ



Синтез

**1. Дано описание рекурсивного алгоритма F.**

```
def F(n):
    if n>2:
        print(n)
        F(n-3)
        F(n-4)
```

**Чему равна сумма напечатанных на экране чисел при выполнении вызова F(10)?**

- A) 23.
- B) 33.
- C) 35.
- D) 10.
- E) 0.

**2. Переменные, область видимости которых распространяется на всю программу, называются:**

- A) локальными;
- B) глобальными;
- C) аргументами;
- D) параметрами-значениями;
- E) параметрами-переменными.

**3. Установите соответствие. Определите назначение операций и методов строковой обработки.****1**

str[2:7]

**A**

поиск первого (считая слева) вхождения подстроки s в строку str; аргументы 2 и 7 определяют начало и конец области поиска



2	<code>len(str)</code>	В	замена в строке <code>str</code> фрагмента (подстроки) '2' на фрагмент '7'
3	<code>str.replace('2','7')</code>	С	разбиение строки <code>str</code> по разделителю
4	<code>str.split(',')</code>	Д	извлечение среза – подстроки, содержащей символы строки <code>str</code> с номерами от 2 до 7
5	<code>str.find(s,2,7)</code>	Е	вычисление длины строки <code>str</code>



4. Верными командами открытия файла для ввода являются:

- A) `my_file = open('file.txt')`
- B) `fin = open('file1.txt', 'r')`
- C) `fout = open('file2.txt', 'w')`
- D) `fapp = open('file3.txt', 'a')`
- E) `file = open(file4.txt).`



5. Дана строка: `s = 'programming'`. Верными извлечениями срезов являются:

- A) `s[0:7]` #programming
- B) `s[3:5]` #gr
- C) `s[2:-2]` #ogrammi
- D) `s[-4:-2]` #mi
- E) `s[:6]` #progra
- F) `s[1:]` #rogramming
- G) `s[:]` #program



6. Определите, что будет напечатано в результате четвертого прохода алгоритма сортировки списка `[53, 48, 100, 7, -5, 0, 67]` методом выбора.

```
def sel_sort(array):
    for i in range(len(array) - 1):
        m = i
```



```

j = i + 1
while j < len(array):
    if array[j] < array[m]:
        m = j
        j = j + 1
array[i], array[m] = array[m], array[i]
print(array)

```

- A) [-5, 0, 7, 48, 53, 100, 67]4
- B) [-5, 0, 100, 7, 53, 48, 67]2
- C) [-5, 0, 7, 100, 53, 48, 67]3
- D) [-5, 48, 100, 7, 53, 0, 67]1
- E) [48, 53, 67, 100, -5, 0, 7]



7. Дан фрагмент программы:

```

size='height - {2}, length - {0}, width - {1}'
print(size.format(3, 6, 2.3))

```

Что будет выведено на экран?

- A) 'length - 3, width - 6, height - 2.3'
- B) 'length - 2.3, width - 3, height - 6'
- C) 'height - 2.3, length - 3, width - 6'
- D) 'height - 3, length - 6, width - 2.3'
- E) 'height - {2}, length - {0}, width - {1}'.



8. Дан фрагмент программы:

```

text='Делу время, потехе час.'
punct='., !?;:-'
n=punct.find(text[10])

```

Чему будет равно значение переменной n?

- A) 1.
- B) 10.
- C) 2.
- D) 7.
- E) 4.





9. Дана весовая матрица графа. Найдите кратчайший маршрут из вершины A в вершину F с помощью «жадного» алгоритма.

	A	B	C	D	E	F
A		2	4			
B	2		9	7		
C	4	9		8	1	
D		7	8		3	1
E			1	3		2
F				1	2	

Напишите найденный путь.



10. Напишите в ответе наименьшее значение входной переменной  $k$ , при котором приведенная программа выдает тот же ответ, что и при входном значении  $k = 10$ .

```
def F(n):
    return n*n*n
def G(n):
    return 2*n+3
k=int(input())
i=1
while F(i)<G(k):
    i+=1
print(i)
```

**ПРОЕКТНАЯ ДЕЯТЕЛЬНОСТЬ ЗА II ЧЕТВЕРТЬ**

**ЦЕЛЬ:** Продемонстрировать навыки по программированию в Python, приобретенные во II четверти.

**ЗАДАЧИ:**

1. Составить план написания программы.
2. Поставить задачи для реализации этого плана.
3. Сформулировать и реализовать алгоритмы решения этих задач.

**ЗАДАНИЯ**

1. Напишите процедуру, печатающую на экран запись заданного числа в римской системе счисления.
2. Напишите процедуру, печатающую на экран заданное десятичное число в восьмеричной системе счисления.
3. Напишите процедуру, печатающую на экран заданное десятичное число в шестнадцатеричной системе счисления.
4. Напишите процедуру, которая выводит на экран линию из **N** символов '-'. Натуральное число **N** передается в качестве параметра.
5. Напишите функцию, которая вычисляет максимальное из трех чисел.
6. Выступление спортсмена на соревнованиях оценивают **N** экспертов. Все они выставляют оценки в баллах (целые числа). Итоговая оценка рассчитывается как среднее арифметическое оценок экспертов, за исключением лучшей и худшей. Напишите функцию, которая вводит оценки экспертов и определяет итоговую оценку выступления спортсмена.
7. Напишите функцию, вычисляющую наименьшее общее кратное двух чисел.
8. Напишите функцию, вычисляющую сумму делителей числа. С ее помощью найдите все пары дружественных чисел, меньших 10 000. Дружественные числа – это два таких натуральных числа, что сумма всех делителей одного числа (меньших самого этого числа) равна другому числу, и наоборот.

9. Напишите функцию, вычисляющую сумму цифр числа. С ее помощью найдите все числа в диапазоне  $[0, N]$ , сумма цифр которых не меняется при умножении на 2, 3, 4, 5, 6, 7, 8 и 9 (например, число 9).
10. Напишите программу, заменяющую во введенной символьной строке все буквы «г» на буквы «д» и наоборот, как заглавные, так и строчные. При вводе строки «догДОГ» должна получиться строка «годГОД».
11. Введите символьную строку и проверьте, является ли она палиндромом.
12. Введите адрес файла и разбейте его на части, разделенные знаком '/'. Каждую часть напечатайте в отдельной строке.
13. Введите строку, в которой записано арифметическое выражение, например, «1 + 5 \* 3». Найдите значение этого выражения.
14. Преобразуйте строку, состоящую из фамилии, имени и отчества, в строку, содержащую инициалы и фамилию. Например, при вводе строки «Аубекова Мадина Аскаркызы» должно получиться «М.А. Аубекова».
15. Напишите функцию, изменяющую в имени файла имеющееся расширение на заданное (например, на .dat). Имя файла и новое расширение передаются функции в качестве параметра.
16. Напишите функцию, определяющую, сколько раз заданное слово входит в символьную строку.
17. Напишите программу, подсчитывающую количество пустых строк в текстовом файле.

## РЕКОМЕНДАЦИИ

При решении задач по составлению алгоритмов и программ следует придерживаться следующих рекомендаций:

- 1) вникнуть в формулировку задачи;
- 2) определить, какие данные являются исходными для решения поставленной задачи;
- 3) выяснить, что необходимо получить в результате решения задачи, каким образом должна быть представлена выходная информация;
- 4) разработать алгоритм решения задачи (для сложных задач алгоритм составляется в несколько этапов: сначала в обобщенном виде, а затем детализируется каждый из шагов);
- 5) необходимо осуществить трассировку алгоритма (на небольшом наборе исходных данных) и убедиться в его правильности;
- 6) алгоритм можно описать в словесном виде или блок-схемой;
- 7) после проверки правильности работы алгоритма на примере составить программу по разработанному алгоритму (закодировать алгоритм);
- 8) обязательно нужно провести тестирование работоспособности программы на контрольном примере с последующим анализом результатов;
- 9) при описании алгоритма и программы необходимо делать пояснения к выполненным шагам, к введенным переменным.



# 4.1. Введение в HTML

Как вы думаете, как создаются web-страницы?

– Что такое web-страница, язык HTML;  
– основные термины HTML;  
– как оформлять текст в web-документе;  
– как располагается текст относительно экрана браузера.

**Web-страница** – Web-парақ – Web page  
**Элементы** – Элементтер – Elements  
**Тег** – Тег – Tag

Первую web-страницу создал Тим Бернерс-Ли 6 августа 1991 года. Она содержала данные о том, что такое Всемирная паутина, как установить web-сервер, как скачать браузер и другую справочную информацию. Впоследствии эта страница была потеряна и вновь найдена благодаря имевшимся офлайновым копиям.

вы узнаете

ключевые слова

это интересно

Многие существующие web-страницы создаются частными лицами – это личные web-страницы. Информация, содержащаяся на личных web-страницах, бывает обычно следующая: контактная информация, хобби, биография, портрет, интересы, профессиональная деятельность, публикации, ссылки на другие web-страницы.

Перед тем, как создать web-страницу, сначала необходимо определить, что будет на ней размещено, и составить ее эскиз:

- информацию сгруппируйте по нескольким разделам;
- определите и введите текст, который должен находиться в каждом из разделов;
- определитесь с графикой и ее расположением на странице;
- определите, сколько примерно места потребуется, если будет включена графика, звук или видеоклипы.

Затем можно приступить к созданию web-страницы. Для ее создания используется язык HTML.

## ЗАПОМНИ

**HTML** (*HyperText Markup Language*

– язык разметки гипертекста) – стандартизированный язык разметки документов во Всемирной паутине. Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

## ОСНОВНЫЕ ТЕРМИНЫ HTML: ЭЛЕМЕНТЫ, ТЕГИ И АТТРИБУТЫ

### Что такое элементы?

Web-страница состоит из элементов. Элементы указывают, как определять структуру и содержимое объектов на странице. Элементы идентифицируются с помощью угловых скобок `<>`, окружающих имя элемента. Таким образом, элемент будет выглядеть так: `<a>`.

### Что такое теги?

Элемент – это набор из тегов и содержания. Если буква или специальное слово взято в угловые скобки `<` и `>`, то означает, что это **тег**. Название тегов можно набирать в любом регистре, т.е. прописными или строчными буквами, как удобно разработчику программы на HTML.

Существует два типа тегов – **парные** и **непарные**. Теги чаще всего идут парами: **открывающий** (`<tag>`) и **закрывающий** (`</tag>`). Между ними находится **содержание** или то, что попадает под действие этого элемента. С того места текста, где использован открывающий тег, он начинает влиять на текст до того места, где используется закрывающий тег. Непарный тег используется один без закрывающего тега, например, `<BR>`.

### ЗАПОМНИ

#### ТЕГ (от англ. tag)

– команда HTML, указывающая, какую операцию необходимо выполнить, например, вывести на экран текст, нарисовать линию или использовать графику.

### Что такое атрибуты и их значения?

Открывающий тег может содержать дополнительную информацию об элементе – **атрибуты** и **значения атрибутов**. Атрибуты являются свойствами, которые позволяют «настроить» тег.

Атрибуты определяются в открывающем теге после имени элемента. Формат для атрибутов состоит из имени атрибута со знаком равенства за ним, а затем в кавычках идет значение атрибута (рис. 4.1.1).

По выполняемым функциям теги делятся на структурные теги, теги формирования абзацев, символов и т.д.

### ЗАПОМНИ

#### АТТРИБУТЫ

являются свойствами, применяемыми для предоставления дополнительной информации об элементе.

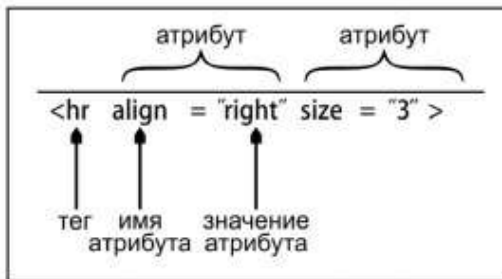


Рис. 4.1.1. Пример команды для создания красной линии, расположенной справа

### Структура документа HTML

HTML-документ находится внутри тега `<html>` и начинается с **заголовка** (англ. *head – голова*), затем идет **тело** (англ. *body*) (рис. 4.1.2). Внутри тега `<head>` лежит заголовок документа, который содержит название документа и некоторые параметры. В тег `<body>` заключается тело документа, которое содержит текст и теги, обрабатываемые браузером перед выводом. Чтобы задать заголовок страницы, необходимо ввести текст между тегами `<title> ... </title>`, которые располагаются в структурном теге `<head>`. Введенный текст будет отображаться в заголовке окна браузера, а также в результатах поиска поисковых систем.

### ЗАПОМНИ

#### ФОРМАТИРОВАНИЕ ТЕКСТА

– это выбор средств для его изменения, таких как редактирование шрифта или использование разнообразных эффектов.



Рис. 4.1.2. Структура HTML-кода



## Оформление текста в web-документе

В таблицах приведены теги управления внешним видом web-документа (табл. 4.1.1) и теги форматирования символов, текста (табл. 4.1.2).

ТАБЛИЦА 4.1.1

## Теги управления внешним видом web-документа

Тег	Комментарий
<p>	Тег «параграф» предназначен для формирования абзаца, который разделяется пустой строкой и не начинается с красной строки
 	Одиночный тег означает переход на новую строку
<HR>	Одиночный тег рисует горизонтальную линию через весь экран
<HR SIZE=" ">	Установка высоты (толщины) линии
<HR WIDTH=" ">	Установка ширины линии, указывают в процентах или пикселях, например: <HR WIDTH="50%"> или <HR WIDTH="325">
<HR COLOR="... ">	Задание определенного цвета линии. <b>Примечание:</b> цвета рассмотрим позже
<HR ALIGN="... ">	Выравнивание линии по значению: – <i>left</i> (по левому краю); – <i>center</i> (по центру); – <i>right</i> (по правому краю)
<H1>...</H1> <H2>...</H2> <H3>...</H3> <H4>...</H4> <H5>...</H5> <H6>...</H6>	Обозначают заголовки документа. <H1> – текст с самым большим размером шрифта (высший уровень), <H6> – с самым малым (низший уровень). <b>Примечание:</b> размер <H3> задается по умолчанию, т.е. если не определен уровень заголовка

Для форматирования текста HTML-документов предусмотрены теги **физического** и **логического** форматирования.



Теги физического форматирования прямо указывают на то, какие свойства шрифта надо изменить (например, сделать жирным), а теги логического форматирования указывают, например, поисковой системе, какой это текст (например, важный).


ТАБЛИЦА 4.1.2

## Теги форматирования текста

Теги физического форматирования	Теги логического форматирования
<code>&lt;b&gt;Текст жирным шрифтом&lt;/b&gt;</code>	<code>&lt;strong&gt;Важный фрагмент текста&lt;/strong&gt;</code>
<code>&lt;i&gt;Текст курсивом&lt;/i&gt;</code>	<code>&lt;em&gt;Акцентированный фрагмент&lt;/em&gt;</code>
<code>&lt;s&gt;Перечеркнутый текст&lt;/s&gt;</code>	<code>&lt;del&gt;Удаленный, зачеркнутый текст &lt;/del&gt;</code>
<code>&lt;u&gt;Подчеркнутый текст&lt;/u&gt;</code>	<code>&lt;ins&gt;Добавленный текст, которого раньше не было, подчеркивание &lt;/ins&gt;</code>

Существует множество программных продуктов, облегчающих работу по созданию HTML-файлов. Все средства работы с HTML можно разделить на три категории: редакторы тегов, шаблоны и программы-преобразователи. Начнем работать в редакторе тегов **Notepad++**, его можно скачать по адресу <https://notepad-plus-plus.org/download/v4.5.6.html>.

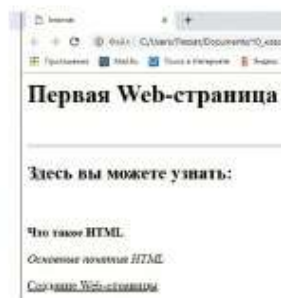


1. Создадим новую папку (каталог) в Windows для хранения всех компонентов вашей страницы.
2. Щелкнем мышью на кнопку **Пуск** (в левом углу рабочего стола).
3. Щелкнем мышью на значке  `notepad++.exe`.
4. Откроется редактор **Notepad++**.
5. Создадим новый документ, и курсор окажется в первой строке. Создадим HTML-документ, набрав в первой строке тег `<html>`. Введем несколько строк с различными тегами и содержанием (код программы на HTML), как на *рис. 4.1.3, а*. При необходимости используйте буфер обмена.



Тег `<meta charset="utf-8">` устанавливает кодировку UTF-8, чтобы в браузере правильно отображалась кириллица.

```
<html>
<head>
  <meta charset="utf-8">
  <title>Internet</title>
</head>
<body>
  <h1>Первая Web-страница</h1>
  <br>
  <hr>
  <h2>Здесь вы можете узнать:</h2>
  <br>
  <p><b>Что такое HTML</b></p>
  <p><i>Основные понятия HTML</i></p>
  <p><u>Создание Web-страницы</u></p>
</body>
</html>
```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.1.3. Пример структуры HTML

**6.** Сохраним текстовый файл. Файлу необходимо присвоить расширение **html**. Для этого необходимо в меню редактора **Notepad++** выбрать команду **Файл – Сохранить как...** В диалоговом окне в строке напротив записи **Имя файла** наберем имя файла **primer** с расширением **html**, как это сделано на *рис. 4.1.4*.

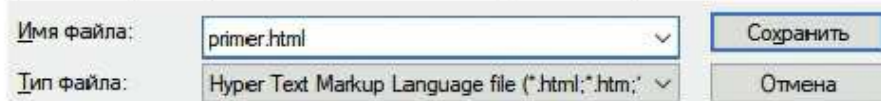


Рис. 4.1.4. Сохранение файла

**ПРИМЕЧАНИЕ:** имя файла и его расширение необходимо записывать только строчными буквами, не используя пробелы в имени.

**7.** Теперь необходимо запустить файл **primer.html**. В открывшемся браузере страница должна выглядеть, как на *рис. 4.1.3, б*.

**ПОЭКСПЕРИМЕНТИРУЙТЕ**

Поэкспериментируйте с примером (рис. 4.1.3). Попробуйте понять, для чего нужен каждый тег.

Если в браузере при просмотре вид web-страницы выглядит по-другому, чем хотелось, то ее всегда можно отредактировать, вернувшись в текстовый редактор **Notepad++**.

Расположение текста меняется путем включения атрибутов в тег **<p>** (табл. 4.1.3).

**ТАБЛИЦА 4.1.3****Теги выравнивания текста**

Тег	Комментарий
<code>&lt;P ALIGN="LEFT"&gt;</code>	Абзац выравнивается по левому краю экрана
<code>&lt;P ALIGN="RIGHT"&gt;</code>	Абзац выравнивается по правому краю экрана
<code>&lt;P ALIGN="CENTER"&gt;</code>	Абзац выравнивается по центру экрана
<code>&lt;P ALIGN="JUSTIFY"&gt;</code>	Абзац выравнивается по ширине экрана

**ПРИМЕЧАНИЕ:** текст по умолчанию автоматически располагается относительно левого края экрана, т.е. если в теге **<p>** не указывать атрибут.



1. Какая информация располагается на web-странице?
2. Что такое HTML, HTML-файл, тег, содержание?
3. Какие структурные теги вы знаете? Дайте им характеристику.
4. Какие теги форматирования текста вы знаете? Дайте им характеристику.
5. Какие теги управления внешним видом web-страниц вы знаете? Дайте им характеристику.
6. Назовите основные действия по созданию web-страницы.
7. Как располагают текст относительно экрана браузера?

**ПОЭКСПЕРИМЕНТИРУЙТЕ**

Примените для созданной web-страницы теги расположения текста.



1. Дан код на HTML:

**<P ALIGN="CENTER">Текст выровнен по центру</p>**

Укажите, где в коде: *элементы, открывающий тег, закрывающий тег, атрибут, значение атрибута, содержание.*

2. Создайте HTML-документ, отображающий на экране браузера следующую информацию: **Фамилия, имя, отчество. Название школы. Номер класса.**

3. Создайте HTML-документ, отображающий на экране браузера следующую информацию (рис. 4.1.5).

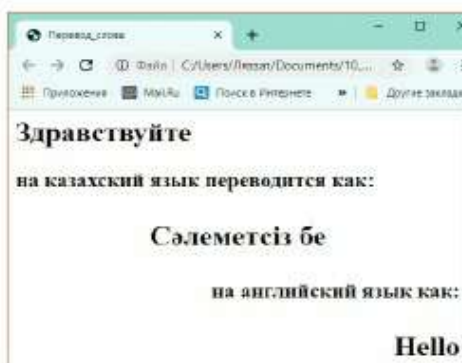


Рис. 4.1.5. Пример выравнивания текста



Найдите ошибки.

```
< html
<head>
<title>Привет, мир!<title>
</head>
<h1>Привет, мир!</h>
<p>Это веб-страница.</p>
</body>
</html>
```



Исследуйте другие редакторы HTML. Сравните преимущества и недостатки HTML-редактора и ввода вручную в **Блокнот**.



## 4.2. Дополнительные возможности форматирования текста в HTML

Как создать web-страницу с разными эффектами?

Как форматировать текст, изменяя его шрифт, цвет, размер.

**Шрифт** – Қарпін – Font  
**Размер** – Өлшемі – Size  
**Цвет** – Түс – Colour  
**Фон** – Фон – Background

Язык HTML изначально был задуман и создан лишь как средство структурирования и форматирования документов, которое не привязано к средствам воспроизведения. Текст с разметкой HTML в идеале должен был единообразно воспроизводиться на разном оборудовании.

вы узнаете

ключевые слова

это интересно

Для использования дополнительного форматирования текста рассмотрим тег **<font>** (табл. 4.2.1).

Описание тега **<font>**

ТАБЛИЦА 4.2.1

Тег	Комментарий
<code>&lt;font size=" " &gt; ...&lt;/font&gt;</code>	Устанавливает размер текста в пределах от 1 (наименьший) до 7 (наибольший). <b>Примечание:</b> если не определять размер текста, то браузер отобразит его размером <code>&lt;font size="3"&gt;...&lt;/font&gt;</code>
<code>&lt;font face=" " &gt; ...&lt;/font&gt;</code>	Устанавливает шрифты различного начертания (гарнитура)
<code>&lt;font color=" " &gt; ...&lt;/font&gt;</code>	Устанавливает цвет текста

### УПРАВЛЕНИЕ РАЗМЕРОМ ТЕКСТА

**Задание относительно размера шрифта с помощью тега **<font>****

Чтобы привлечь внимание к определенному фрагменту web-страницы,

## 10 класс | информатика

используют метод относительного задания размеров, например, вывод первой буквы фразы более крупным шрифтом, чем остальные.

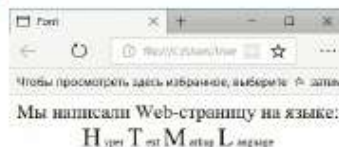
В теге **<font>** вместо определенного номера значения атрибута **size** имеется возможность задать размер шрифта относительно, например **+3** или **-7**. Тогда web-браузер прибавляет (или вычитает) указанное число к номеру шрифта, который используется по умолчанию (размер 3).

Выполним пример создания текста с заглавными прописными буквами (рис. 4.2.1) на компьютере.



1. Откроем программу **Notepad++**.
2. Введем код программы на HTML, как на рис. 4.2.1, а.
3. Сохраним файл, как **primer2.html** в своей папке.
4. Запустим **primer2.html**. Должно получиться, как на рис. 4.2.1, б.

```
<html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Font</title>
  </head>
  <center>
    <font SIZE="5">Мы написали Web-страницу на языке: </font>
    <font size="6">H</font>
    <font size="-1">yper</font>
    <font size="6"> T</font>
    <font size = "-1">ext</font>
    <font size="6"> M</font>
    <font size = "-1">arkup</font>
    <font size="6"> L</font>
    <font size = "-1">anguage</font>
  </center>
</body>
</html>
```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.2.1. Создание текста с заглавными прописными буквами

## ШРИФТЫ РАЗЛИЧНОГО НАЧЕРТАНИЯ

Можно использовать множество различных шрифтов, установленных на компьютере для текстового редактора.

Приведем примеры: `<font FACE= "Arial">Интернет</font>`  
`<BR><font FACE= "Coronet">Интернет</font>`

## УПРАВЛЕНИЕ ЦВЕТОМ ТЕКСТА

Цвета задаются шестнадцатеричным кодом из шести знаков, перед ним нужно установить знак # или можно указать буквенное значение цвета.

Зададим, например, текст красного цвета:

```
<font COLOR="#FF0000">Текст красного цвета</font>
```

где **#FF0000** – код красного цвета,

или `<font COLOR="red">Текст красного цвета </font>`,

где **red** – название цвета.

**ПРИМЕЧАНИЕ:** цветом по умолчанию является черный, он будет использован, если не определен цвет текста.

Названия цветов на английском языке можно найти по адресу [https://puzzleweb.ru/html/colors\\_html.php](https://puzzleweb.ru/html/colors_html.php).

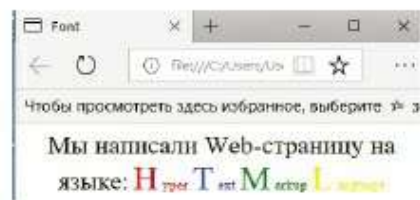
Изменим цвет текста. Поэкспериментируем с файлом **primer2.html**. Напишем код программы, как на *рис. 4.2.2, а*. Сохраним файл как **primer3.html** в своей папке. Результат должен получиться, как на *рис. 4.2.2, б*.

```
<html>
<html>
<head>
  <meta charset="utf-8">
  <title>Font</title>
</head>
  <center>
<font size="5">Мы написали Web-страницу на языке:
</font>
  <font size="6" color = "#FF0000">H</font>
  <font size="-1" color = "red">yper</font>
  <font size="6" color = " #0000FF"> T</font>
  <font size = "-1" color = "blue">ext</font>
  <font size="6" color = " #008000 "> M</font>
```

```

<font size = "-1" color = "green">arkup</font>
<font size="6" color = " #FFFF00"> L</font>
<font size = "-1" color = " Yellow" >anguage</font>
</center>
</body>
</html>

```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.2.2. Создание цветного текста

При вводе текста в web-страницу необходимо слегка выделить определенные его фрагменты, используя при оформлении текста специальные символы. Язык HTML предлагает ряд возможностей (табл. 4.2.2. и 4.2.3).

ТАБЛИЦА 4.2.2

## Теги дополнительного форматирования

Тег	Комментарий
<SUB>...</SUB>	Выводит текст в виде нижнего индекса
<SUP>...</SUP>	Выводит текст в виде верхнего индекса
<S>...</S>	Зачеркнутый текст

ТАБЛИЦА 4.2.3

## Специальные символы

Символ	Название	Запись в HTML
§	Знак параграфа	&sect
°	Знак градуса	&deg
"	Прямая кавычка	&quot

**ПРИМЕЧАНИЕ:** запись в HTML необходимо осуществлять строчными буквами. Если знак градуса будет записан **&DEG** вместо **&deg**, то на экране браузера вместо символа ° отобразится **&DEG**.





Выполним пример (рис. 4.2.3) использования верхнего и нижнего индексов на компьютере и сохраним как **primer4.html**.

```
<html>
<head>
<meta charset="utf-8">
<title>NS</title>
</head>
<body>
<b>Перевод чисел из двоичной системы
счисления в десятичную:</b>
101101<sub>2</sub> = 45<sub>10</sub>
<br><b>Единицы измерения информации</b>
<br>1 килобайт = 1 Кб = 2<sup>10</sup>
байт = 1024 байта
</body>
</html>
```

<p><b>Перевод чисел из двоичной системы счисления в десятичную:</b> <math>101101_2 = 45_{10}</math>  <b>Единицы измерения информации</b>  1 килобайт = 1 Кб = <math>2^{10}</math> байт = 1024 байта</p>
---

а) Вид в Notepad++

б) Вид в браузере

**Рис. 4.2.3.** Использование верхнего и нижнего индексов

### ИЗМЕНЕНИЕ ЦВЕТА С ПОМОЩЬЮ ТЕГА <BODY>

Тег **<body>** задает цвет фона и цвет основного текста документа. Для этого в тег **<body>** необходимо записать два ключевых слова:

- bgcolor="..." устанавливает цвет фона;
- text=" " задает цвет текста.



Выполним пример (рис. 4.2.4) на компьютере, используя файл **primer2.html**. На темно-зеленом фоне запишем текст белого цвета. Сохраним как **primer5.html**.

```
<html>
<head>
<meta charset="utf-8">
<title>Font</title>
```



```

</head>
<body bgcolor="#008000" text="#FFFFFF"> <center>
<font size="5">Мы написали Web-страницу на языке: </font>
<font size="6" >H</font>
<font size="-1" >yper</font>
<font size="6" > T</font>
<font size = "-1">ext</font>
<font size="6" > M</font>
<font size = "-1" >arkup</font>
<font size="6" > L</font>
<font size = "-1" >anguage</font>
</center>
</body>
</html>

```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.2.4. Создание темно-зеленого фона

## СОЗДАНИЕ БЕГУЩЕЙ СТРОКИ

В качестве бегущей строки (БС) принимается заголовок текста или строка, требующая особого внимания от посетителей данной web-страницы (табл. 4.2.3).

ТАБЛИЦА 4.2.3

### Тег создания бегущей строки и его атрибуты

Тег	Комментарий
MARQUEE>...</MARQUEE>	Создает БС
ALIGN="..."	Располагает БС по вертикали относительно текста, в котором она расположена. Принимает значения: – TOP – по верхнему положению; – MIDDLE – посередине; – BOTTOM – по нижнему положению
BGCOLOR="..."	Задаёт цвет фона БС



1. Перечислите возможности тега `<font>`.
2. Какой размер шрифта у данного текста?



```
<font SIZE=+4>Возможности HTML</font>!
```

3. Каким образом браузер отобразит текст, заданный шрифтами различного начертания? Что произойдет с текстом, если такие шрифты на персональном компьютере посетителя не установлены?
4. Какие специальные символы предусмотрены в HTML?
5. Как можно изменить цвет текста и фона с помощью тега `<body>`?
6. Какие способы задания цвета существуют в HTML?
7. Чем отличается тег `<font>` от тега `<body>` при задании цвета текста?
8. Как создать БС на web-странице? Перечислите и дайте характеристику атрибутам БС.



1. Создайте HTML-документ, отображающий на экране браузера следующую информацию (рис. 4.2.5).

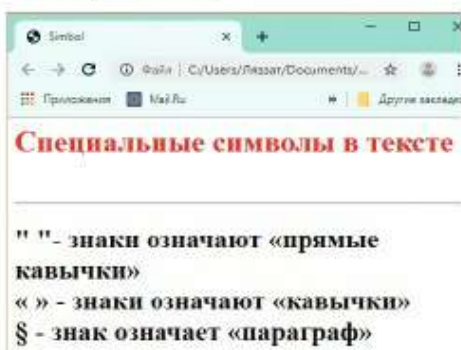


Рис. 4.2.5. Специальные символы в тексте

**ПРИМЕЧАНИЕ:** заголовок текста выполнен в виде бегущей строки (БС). Для БС заданы следующие параметры: **текст БС перемещается слева направо; задан фон для БС; число повторов анимации БС равно трем; задан стиль БС: ALTERNATE.**

2. Создайте HTML-документ, отображающий на экране браузера следующую информацию (рис. 4.2.6).

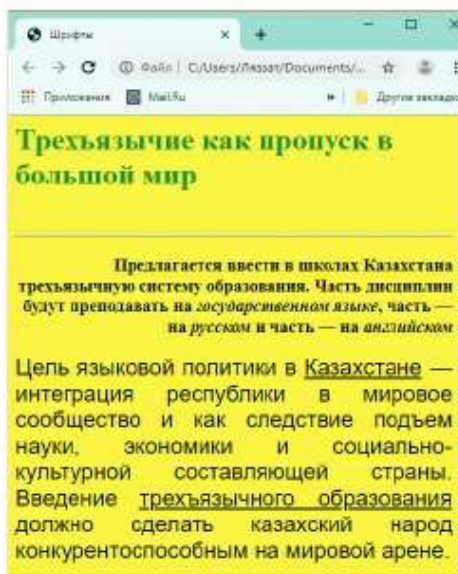


Рис. 4.2.6. Вид web-страницы в окне браузера



Найдите ошибки.

```
< html>
<head><title> Форматироваие текста</title></head>
<body>
<P ALIGN= "CENTER "><B><FONT COLOR='W066FF" SIZE=
="4"><BIG>&sect 1. Первый параграф</BЮ</FONT></B>
<BOD Y BGCOL OR = "#008000 " TEXT = "HFFFAFA " Face=
"Times Roman">
<FONT Colors "#008000 " >
<BR>Темно-зеленый текст, заданный тегом FONT</font>
</html>
```



Проведите исследование на тему «Форматирование текста в HTML». Рассмотрите еще теги, которые используются для форматирования текста.



## 4.3. Вставка таблиц и создание списков в HTML

Как вы думаете, для чего данные объединяются в списки и таблицы?

– Как создать маркированный и нумерованный списки на веб-странице;  
– как создать таблицы с помощью HTML.

**Нумерованный список** –  
Нөмірленген тізім –  
Numbered list  
**Маркированный список** –  
Белгіленген тізім –  
Bulleled list  
**Ячейка** – Ұяшық – Cell  
**Таблица** – Кесте – Table

На пути к вершине горы Эверест есть доступ к высокоскоростному Интернету.

вы узнаете

ключевые слова

это интересно

### СОЗДАНИЕ HTML-СПИСКОВ

#### ЗАПОМНИ

**СПИСКОМ** называется взаимосвязанный набор отдельных фраз или предложений, который начинается с маркера или цифры.

Списки могут быть **вложенными**. Например, во вложенных списках маркер первого уровня – круг, второго – окружность, третьего – квадрат.

Изменить вид маркера списка можно с помощью стиля **list-style**. О стилях в HTML вы узнаете на следующих уроках.

ТАБЛИЦА 4.3.1

#### Теги для создания списков

Тег	Комментарий
<ul>...</ul>	Обрамление маркированного списка
<ol>...</ol>	Обрамление нумерованного списка
<li>...</li>	Обрамление каждого элемента списка



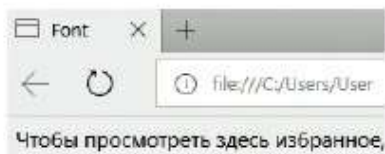
Выполним пример создания **маркированного** списка (рис. 4.3.1) на компьютере и сохраним файл как **primer6.html** в своей папке.



```

<html>
<head>
  <meta charset="utf-8">
  <title>List1</title>
</head>
<body>
  <ul>
    <li>Абай Кунанбаев
      <ul>
        <li>&laquo;Слова назидания&raquo;;</li>
        <li>&laquo;Ты зрачок глаз моих&raquo;;</li>
        <li>&laquo;Двадцать стихотворений&raquo;;</li>
      </ul>
    </li>
    <li>Мухтар Ауэзов
      <ul>
        <li>&laquo;Путь Абая&raquo;;</li>
        <li>&laquo;Племя младое&raquo;;</li>
        <li>&laquo;Выстрел на перевале&raquo;;</li>
      </ul>
    </li>
  </ul>
</BODY>
</HTML>

```



- Абай Кунанбаев
  - "Слова назидания"
  - "Ты зрачок глаз моих"
  - "Двадцать стихотворений"
- Ауэзов М
  - "Путь Абая"
  - "Племя младое"
  - "Выстрел на перевале"

а) Вид в Notepad++

б) Вид в браузере

Рис. 4.3.1. Маркированный список

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Измените пример (рис. 4.3.1), создав **нумерованный** список.

## СОЗДАНИЕ ТАБЛИЦЫ В HTML

Для лучшего представления информации можно использовать таблицы. Существует тег-контейнер **table**, в котором размещается содержимое таблицы. Контейнер **tr** используется для обозначения строки, контейнер **th** – для обозначения заголовков столбцов (строк), а контейнер **td** – для размещения данных в ячейках. Заголовки выделяются полужирным шрифтом и располагаются по центру в своих ячейках. Данные выравниваются по левой стороне ячейки и имеют обычный шрифт. По умолчанию вокруг ячеек таблицы рамок нет.

ТАБЛИЦА 4.3.2

## Атрибуты тегов таблицы

Атрибут	Описание значения атрибута
align="..."	Содержимое ячеек выравнивается в зависимости от значения атрибута (left, center, right, justify)
background="URL"	Задаёт фон ячеек с помощью URL изображения
bgcolor="цвет"	Задаёт цвет фона ячеек строки
bordercolor="цвет"	Изменяет цвет рамки ячейки
width = "..."	Определяет ширину всей таблицы или ячейки в пикселях либо в процентах от ширины окна браузера
height="..."	Определяет высоту всей таблицы или ячейки в пикселях либо в процентах от высоты окна браузера
border="..."	Устанавливает ширину рамки таблицы в пикселях, например, border=2



Выполним пример создания двух таблиц без рамки и с рамкой (рис. 4.3.2) на компьютере, сохраним файл как **primer7.html** в своей папке. Результат должен выглядеть, как на рис. 4.3.2, б.

```
<html>
<head>
<meta charset="utf-8">
<title>Table1</title>
</head>
```



```

<body>
<font color = "red"><h3>Таблица без рамок</h3> </font>
<table>
  <tr>      <th>Имя</th> <th>Фамилия</th>      </tr>
  <tr>      <td>Айнур</td> <td>Исаева</td>      </tr>
  <tr>      <td>Аскар</td> <td>Ахметов</td>      </tr>
</table>
<font color = "red"><h3>Таблица с рамками </h3> </font>
<table border=1>
  <tr>      <th>Имя</th> <th>Фамилия</th>      </tr>
  <tr>      <td>Айнур</td> <td>Исаева</td>      </tr>
  <tr>      <td>Аскар</td> <td>Ахметов</td>      </tr>
</table>
</body>
</html>

```

### Таблица без рамок

Имя Фамилия  
Айнур Исаева  
Аскар Ахметов

### Таблица с рамками

Имя	Фамилия
Айнур	Исаева
Аскар	Ахметов

а) Вид в Notepad++

б) Вид в браузере

Рис. 4.3.2. Таблицы без рамки и с рамкой

## ОБЪЕДИНЕНИЕ ЯЧЕЕК

Для соединения двух смежных ячеек в одном столбце нужно использовать атрибут **rowspan** тега **<th>** или **<td>**, например **<td rowspan=2>**.

Для объединения двух смежных ячеек в одной строке нужно использовать атрибут **colspan** тех же тегов, например **<td colspan=2>**.





Выполним пример создания объединения ячеек в таблице (рис. 4.3.3) на компьютере. Сохраним файл как **primer8.html** в своей папке.

```
<html>
<head>
  <meta charset="utf-8">
  <title>Table1</title>
</head>
<body>
  <table border>
    <tr align="center">
      <td rowspan="2"></td> <td colspan="2">
        <font color="red"><b>хорьки </b></font> </td>
      </tr>
      <tr>
        <td >вес</td> <td align="center">
          >размер</td> </tr>
      <tr align="center">
        <td><font color="gray">самцы</font></td>
        <td>1.2 - 2.5 кг</td> <td>до 70см</td>
      </tr>
      <tr align="center">
        <td><font color="pink">самки</font></td>
        <td>0.7 - 1.0 кг</td> <td>до 40см</td>
      </tr>
    </table>
  </body>
</html>
```

	хорьки	
	вес	размер
самцы	1.2 - 2.5 кг	до 70см
самки	0.7 - 1.0 кг	до 40см

а) Вид в Notepad++

б) Вид в браузере

Рис. 4.3.3. Объединение ячеек



1. Назовите основной структурный элемент таблицы.
2. Что такое список? Перечислите его разновидности.
3. В чем отличие маркированного списка от нумерованного?
4. Чем отличаются теги `<th>...</th>` от `<td>...</td>`?





5. Для чего необходимо объединение ячеек?
6. Как выделить ячейку цветом?
7. Как поменять цвет границы ячейки?
8. Какие атрибуты используются для выравнивания текста внутри ячейки?



1. Создайте маркированный и нумерованный списки.

1. Шрифты:
  - размер;
  - цвет;
  - к набору;
  - индексы.
2. Заголовки:
  - с 1-го по 6-й уровень.
3. Пункты:
  - выравнивание;
  - разрыв линий в параграфе;
  - использование переформатирования.

2. Создайте новую web-страницу. Разместите на странице таблицу.
  - 1) Таблицу выровняйте по центру страницы.
  - 2) Задайте цвет фона таблицы **"#E2E2E2"**.
  - 3) Задайте ширину таблицы **400 пикселей**.
  - 4) Задайте расстояние между границей ячейки и ее содержимым **20 пикселей**.
  - 5) Задайте толщину границы **4 пикселя**, цвет границы – **"#008000"**.
  - 6) Второму ряду таблицы задайте цвет **"#FFFFCA"**.
  - 7) Задайте цвет ячейки 3 **"#FFCACA"**, цвет ячейки 5 – **"#ECFFEC"** (см. рис. 4.3.4).

Сохраните файл в своей папке как файл **table2.html**.

1	2	3
4	5	6
7	8	9

**Рис. 4.3.4.** Пример форматирования заднего фона таблицы



### 3. Создайте новую web-страницу.

- 1) В ней создайте таблицу, состоящую из пяти рядов и пяти колонок.
- 2) Объедините ячейки таблицы (рис. 4.3.5).
- 3) Сохраните страницу в своей папке как файл **table3.html**.

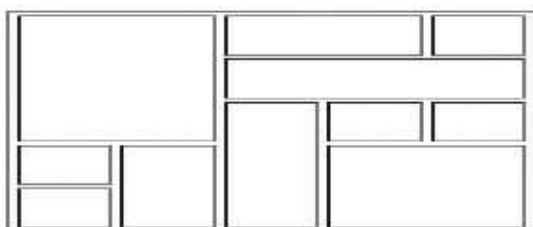


Рис. 4.3.5. Пример объединения ячеек



Составьте кроссворд на тему «Таблицы и списки в HTML».



Выполните проектную работу: создайте web-страницу (рис. 4.3.6), используя теги для создания таблицы и другие изученные теги.

The screenshot shows a web browser window with the address bar displaying 'C:\Users\Lesson\Documents\10\_lesson\HTML\_урок3\table3.html'. The page content includes:

- Учебник HTML для начинающих**
- О чем и для кого эта книга**
- Автор:** Семенихин Алексей Александрович
- Рассчитан данный учебник** именно для начинающих "с нуля" людей, которые только только начали привносить навыки в удивительный мир веб-разработки, поэтому старался писать его как можно более развернуто, что называется "подробнее" с минимальным количеством аббревиатур.
- Дополнительная литература**
- Полосинский Е.** «Создание HTML» ISBN 5-8459-0466-3
- Гончаров А.** «Создание HTML» ISBN 5-89302-143-7
- Полосин В.** «Создание Web-страниц в Web-сайтах» ISBN 5-89302-143-7
- Вана Валентина** «HTML. Структурный» ISBN 978-5-388-00215-0
- Учебник HTML**
  - Описание
  - Что такое HTML язык
  - Редактируем текст
  - Рисуем
  - Таблицы
  - Ссылки
- И ещё...**
  - Фреймы
  - Настройка меню
  - Мета теги
  - Звук
  - И снова таблицы...и
  - Формы
  - Текст, логика и файлы
- Дополнительные ссылки:**
- Каждая практическая задача сопровождается ссылкой на адрес <http://www.webmaster.ru/html/ur3a01.html>.
- Все задания могут выполняться с помощью, советам и комментариями по e-mail: [valentinalex@yandex.ru](mailto:valentinalex@yandex.ru) – С уважением, Валентина Вана Валентина, куда Вас не формирует <http://forum.webmaster.ru/index.php>

Рисунок 4.3.6. Проектная работа «Создание таблицы»

## 4.4. Вставка изображения на web-страницу

Как вы думаете, можно ли редактировать изображение на web-странице?

– Как вставить изображение на web-страницу;  
– какие атрибуты есть у тега для вставки изображения;  
– как сделать фон с изображением.

**Изображение** – Кескін – Image  
**Ссылка** – Сілтеме – Link  
**Адрес** – Мекенжай – Address

Смайлы, которые сегодня используются в повседневном общении в социальных сетях и мессенджерах, появились около 30 лет назад. Автором первого в мире смайла считается программист Скотт Фальман.

вы узнаете

ключевые слова

это интересно

### КАК ВСТАВИТЬ ИЗОБРАЖЕНИЯ НА WEB-СТРАНИЦУ

Для вставки изображений на страницу применяют непарный тег **<img>**. Стандартные форматы изображений – это **GIF**, **PNG** и **JPEG**.

**JPEG** (*Joint Photographic Experts Group*) лучше всего использовать для хранения фотографий или полутоновых картинок, не содержащих текста.

**GIF** (*Graphics Interchange Format*) – для анимации.

**PNG** (*Portable Network Graphics*) – формат для всего остального (иконки, кнопки и др.).

Тег **<img>** не имеет значения без соответствующих атрибутов. К нему необходимо добавить соответствующие атрибуты. Первый, самый главный атрибут этого тега, – это атрибут **src** (англ. *source* – источник), который указывает на место расположения самого файла изображения.

Код вставки изображения в HTML-страницу имеет такой вид:

```

```

Если файл **image.jpg** лежит в той же папке, что и файл страницы, то достаточно указать имя изображения. Если файл изображения находится в другом месте, необходимо написать полный путь к нему.

Приведем несколько примеров указания адреса файла с изображением.



`` – в папке, где расположен HTML-документ, вложена папка **myfoto**, в которой расположен файл **foto.jpg**.

`` – фотография размещена на уровень выше от документа.

Так же можно указать место фотографии, ссылаясь на определенный интернет-ресурс. Для этого в атрибут **src** нужно вставить URL-адрес изображения: ``

## КАК СДЕЛАТЬ ФОН С ИЗОБРАЖЕНИЕМ

**Фоновое изображение** представляет собой графический файл с рисунком, который несколько раз повторяется и заполняет всё окно браузера любого размера. В теге `<body>` с помощью атрибута **background** задается название файла графики, используемой в качестве фоновой:

`<body background="fon.jpg">`



Выполним пример вставки изображения (рис. 4.4.1) на компьютере.  
1. Продолжим создание страницы, которая была создана на рис. 4.3.1.

```
<html>
<head>
  <meta charset="utf-8">
  <title> 0 казахских просветителях</title>
</head>
<body background="fon.jpg">
<marquee behavior="slide"><font color=red>
  <font size = 30> <b> Наша гордость!</b></font>
</marquee>
  <dl>
    
    <font size = 24> <dt>Абай (Ибрагим) Кунанбаев (1845-
1904) </dt> </font>
    <h2><dd>
      Великий казахский поэт, композитор, просветитель,
мыслитель, общественный деятель, основоположник казахской
письменной литературы и её первый классик, реформатор
культуры в духе сближения с русской, а также европейской
культурой на основе просвещённого Ислама.
    </dd>
```



```

<br><EM>Произведения Абая Кунанбаева </EM>
<ol>
  <li>&laquo;Слова назидания&raquo;</li>
  <li>&laquo;Ты зрачок глаз моих&raquo;</li>
  <li>&laquo;Двадцать стихотворений&raquo;</li>
</ol></h2>

<dt> <font size = 24>Ауэзов Мухтар Омарханович (1897–
1961) </font></dt>
<h2> <dd>
  Один из наиболее широко известных казахских
  писателей, советский драматург и учёный. Академик АН
  Казахской ССР, председатель Союза писателей Казахстана.
  Его двухтомный роман «Путь Абая» вошёл в «Библиотеку
  всемирной литературы».
</dd>
<br><EM>Произведения Ауэзова М. </EM></b>
<ol>
  <li>&laquo;Путь Абая&raquo;</li>
  <li>&laquo;Племя младое&raquo;</li>
  <li>&laquo;Выстрел на перевале&raquo;</li>
</ol></h2>
</dl></font> </body>
</html>

```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.4.1. Вставка изображений



2. В папку рядом с HTML-файлом поместим файлы с изображениями А. Кунанбаева **Abai.jpg**, М. Ауэзова **Mukhtar.jpg** и с изображением фона **fon.jpg**.

3. Откроем программу **Notepad++**.

4. Затем нужно открыть **primer6.html**. Вставим текст (данные о прожекторах можно взять из <https://www.wikipedia.org/>) в нужное место.

5. Введем недостающий код программы на HTML (см. рис. 4.4.1, а).

6. Сохраним файл как **primer9.html** в своей папке.

7. Запустим **primer9.html**. Должно получиться, как на рис. 4.4.1, б.

**ПРИМЕЧАНИЕ:** если изображение расположено на вашем компьютере, то просто вставить его на страницу в Интернете не получится. Для этого сначала изображение нужно переместить в какое-нибудь место в Интернете (на файловый хостинг, например). В коде страницы нужно указать полный URL-адрес данного места с изображением.

Приведем атрибуты для тега **<img>** (табл. 4.4.1).

ТАБЛИЦА 4.4.1

## Атрибуты тегов изображения

Атрибут	Описание значения атрибута
width = "..."	Ширина картинка в пикселях или процентах
height = "..."	Высота картинка в пикселях или процентах. < img src = "Abai.jpg" width = "50" height = "20" > < img src = "Mukhtar.jpg" width = "10%" height = "5%" >
alt = "..."	Выводит альтернативный текст. < img src = "Abai.jpg" alt = "Фото Абая" >
title = "..."	Добавление всплывающей подсказки к изображению. < img src = "Abai.jpg" alt = "Фото Абая" title = "Абай Кунанбаев" >
align = "..."	Определяет режим выравнивания изображения относительно других элементов HTML-страницы (left, right, center). < img src = "Mukhtar.jpg" align = "left" > < p > Ауэзов Мухтар Омарханович < /p >
vspace = "..."	Создание верхнего и нижнего отступов

hspace = "..."	Создание боковых отступов (слева и справа).  < p> Ауэзов Мухтар Омарханович < /p>
border = "..."	Создание черной рамки вокруг изображения. Определяет толщину рамки в пикселях. Например, < img src="Abai.jpg" border="5">

Для прекращения обтекания картинки текстом можно использовать тег **<br>** с атрибутом **clear** (табл. 4.4.2).

ТАБЛИЦА 4.4.2

Атрибут тега **<br>**

Атрибут	Описание значения атрибута
clear	Прекращение обтекания текстом изображения, может принимать три значения (left, right, all)

**ПРИМЕЧАНИЕ:** использовать чужие изображения можно только в том случае, если получено разрешение от владельца или если они взяты с сайтов с бесплатными картинками.

## ПОЭКСПЕРИМЕНТИРУЙТЕ

Поэкспериментируйте с нашим сайтом о казахских просветителях, добавьте различные атрибуты для изображений.



1. Чем отличаются форматы **GIF** и **JPEG**?
2. Каким образом можно задать рамку для изображения?
3. Как используется альтернативный текст?
4. Каким образом задаются ширина и высота изображения?
5. Как задаются отступы от изображения?
6. Как расположить текст рядом с изображением?
7. Что такое пиктограммы? Расскажите об их назначении и видах.
8. Как создаются фоновые изображения на web-странице?





1. Создайте HTML-документ, отображающий на экране браузера информацию, показанную на *рис. 4.4.2*.



Рис. 4.4.2. Вид web-страницы в окне браузера

2. Внесите изменения в текст файла **primer9.html**: используйте атрибуты графики **ALT**, **BORDER**, **ALIGN**, **HEIGHT**, **WIDTH**, **VSPACE**, **HSPACE**. Обратите внимание на размер графического файла (в байтах), потому что это влияет на время загрузки web-страницы. Просмотрите, как изменится ваша web-страница в браузере.

3. Добавьте на первую страницу (**primer.html**) свою фотографию вместо строки **Первая web-страница**. Используя атрибуты **width** и **height**, уменьшите и увеличьте размер изображения в три раза. Заметна ли потеря качества изображения при увеличении?



Найдите ошибку в коде.

```
<html>
<head> <title>Imagel<title> </head>
```



```
<body>
  <p></p>
</body>
</html>
```



Напишите эссе на тему «Форматы изображений на web-странице. Искажение растрового изображения при масштабировании».

## 4.5. Вставка гиперссылки на web-страницу

Как вы думаете, для чего нужны гиперссылки?

- Как создать ссылку на другую страницу, web-страницу;
- как создать ссылку внутри страницы;
- как сделать ссылкой изображение;
- как сделать ссылку на электронную почту;
- как задать цвета всех ссылок на странице.

**Ссылка** – Сілтеме – Link

**Адрес** – Мекенжай – Address

**Электронная почта** – Электрондық пошта – Email

То, что сегодня называется Интернетом (протокол HTML), на самом деле является результатом исследования по ядерным разработкам. Изначально целью было придумать способ обмена документами. Он заключался в использовании гиперссылок. Из этого проекта и родилась Всемирная сеть.

вы узнаете

ключевые слова

это интересно

### ГИПЕРССЫЛКИ В HTML

HTML-документ отличается наличием гиперссылок (или просто ссылок) на другие сайты, документы, файлы и т.д.

Гиперссылка может ссылаться на следующие объекты и элементы: приложение, файл, каталог, заголовок, примечание, текст,

### ЗАПОМНИ

#### ГИПЕРССЫЛКА

(англ. *hyperlink*)

– часть web-документа, которая ссылается на другой элемент на этой же странице либо на любой другой.

изображение и др. Эти объекты располагаются на локальном диске или в компьютерной сети, в Интернете.

Ссылки разделяют на **внешние** и **внутренние**, а также на **текстовые** и **графические** (изображения).

### ВНЕШНИЕ ССЫЛКИ

Ссылки, которые ведут за «пределы» HTML-страницы, называются **внешними**. Ссылки на каталог, на файл, на сайт, на страницу являются **внешними** по отношению к исходной странице, и все создаются одним и тем же способом.

HTML-ссылки определяются тегами `<a> ... </a>` (сокращение от англ. *anchor* – якорь).

Текстовые ссылки в браузере отображаются как текст (по умолчанию он выделен синим цветом и подчеркнут), а графические – в качестве объекта с картинкой, по которому нужно щелкнуть для перехода. При нажатии по ссылке браузер загружает страницу, указанную в атрибуте **href**. Например, рис. 4.5.1.

<code>&lt;a href="http://www.cprm-game.kz/"&gt; Ссылка на сайт &lt;/a&gt;</code>	<a href="http://www.cprm-game.kz/">Ссылка на сайт</a>
<code>&lt;a href="http://www.cprm-game.kz/beginner.html"&gt;Ссылка на страницу &lt;/a&gt;</code>	<a href="http://www.cprm-game.kz/beginner.html">Ссылка на страницу</a>
<code>&lt; a href = "http://www.cprm-game.kz/files/p_Bigfoot2017.pdf" &gt; Ссылка на страницу &lt; /a&gt;</code>	<a href="http://www.cprm-game.kz/files/p_Bigfoot2017.pdf">Ссылка на файл</a>
<code>&lt;a href="http://www.cprm-game.kz/"&gt; &lt;img src="smile.png"&gt; &lt;/a&gt;</code>	

а) Вид в Notepad++

б) Вид в браузере

Рис. 4.5.1. Создание ссылки на web-странице

#### ПОЭКСПЕРИМЕНТИРУЙТЕ

В тег `<a>` одновременно поместите и картинку, и текст (рис. 4.5.2).



[Текст к картинке](#)

Рис. 4.5.2. Ссылка в виде картинки и текста

## ВНУТРЕННИЕ ССЫЛКИ

Внутренние ссылки ведут на различные части этой же страницы.

Для осуществления навигации внутри одной страницы сайта необходимо предварительно разметить ее якорями и затем ссылаться на них. Для разметки используйте атрибут **name**.

```
< a name="имя якоря">текст< /a>
```

Якорь надо расположить в тех местах страницы, в которые желает попасть пользователь после клика по ссылке.

Имя нужного якоря с обязательным символом решетки (**#**) перед ним указывается в атрибуте **href** внутренней ссылки вместо адреса.

```
<a href="#имя якоря">текст </a>
```



Продолжим создание страницы о казахских просветителях.

1. Откроем в **Notepad++** файл **primer9.html**.
2. Разметим четыре первых подзаголовка атрибутом **name**, присвоив им значения «1», «2», «3», «4» соответственно (рис. 4.5.3).

<code>&lt;a name = "1"&gt; Абай (Ибрагим) Кунанбаев &lt;/a&gt;</code>
<code>&lt;a name = "2"&gt; Произведения Абая Кунанбаева&lt;/a&gt;</code>
<code>&lt;a name = "3"&gt;Ауэзов Мухтар Омарханович &lt;/a&gt;</code>
<code>&lt;a name = "4"&gt; Произведения Ауэзова М.&lt;/a&gt;</code>

Рис. 4.5.3. Вид в Notepad++

3. Добавим в начале или в конце страницы (на ваше усмотрение) следующий код (рис. 4.5.4, а).

<code>&lt;a href = "#1"&gt; Кунанбаев &lt;/a&gt;</code>	<a href="#">Кунанбаев</a>
<code>&lt;a href = "#2"&gt; Произведения Кунанбаева&lt;/a&gt;</code>	<a href="#">Произведения Кунанбаева</a>
<code>&lt;a href = "#3"&gt;Ауэзов &lt;/a&gt;</code>	<a href="#">Ауэзов</a>
<code>&lt;a href = "#4"&gt; Произведения Ауэзова&lt;/a&gt;</code>	<a href="#">Произведения Ауэзова</a>

а) Вид в Notepad++

б) Вид в браузере

Рис. 4.5.4. Внутренние ссылки

4. Сохраним файл **primer9.html** в своей папке.
5. Запустим данный файл. Должно получиться, как на рис. 4.5.4, б.



Нажмите на любую из четырех верхних ссылок и переместитесь к соответствующему заголовку, адрес страницы в браузере изменится на **http://../primer9.html/#1**.

Есть анкоры, которые используются в браузерах по умолчанию, например, нажатие по ссылке **#top** автоматически прокрутит страницу в самый верх – в начало сайта.

### ССЫЛКА НА ЭЛЕКТРОННУЮ ПОЧТУ

Для создания ссылки на электронную почту необходимо вместо URL-адреса в значении атрибута **href** набрать адрес электронной почты с указанием протокола (**mailto:**). Когда вы выберете такую ссылку, откроется окно почтовой программы и в поле **Кому** отобразится указанный адрес электронной почты. HTML-код:

```
<a href="mailto:lyazzat_r@gmail.com">Моя почта </a>
```

**ПРИМЕЧАНИЕ:** если ваш браузер не реагирует на ссылку **Моя почта**, то необходимо добавить почтовое приложение или **Outlook** как приложение по умолчанию: **Панель управления – Программы – Задание программ по умолчанию**. Из списка выбрать программу для работы с почтой, после этого ссылка будет работать.

### ЗАДАНИЕ ЦВЕТА ВСЕХ ССЫЛОК НА СТРАНИЦЕ

Цвета ссылок задаются в качестве атрибутов тега **<body>** (табл. 4.5.1). Значения используются по умолчанию, если необязательные атрибуты не указаны.

Атрибуты тега **<body>**

ТАБЛИЦА 4.5.1

Атрибут	Описание значения атрибута
link	Определяет цвет ссылок на web-странице (цвет по умолчанию синий, #0000FF)
alink	Цвет активной ссылки. Цвет ссылки меняется при нажатии на ней кнопки мыши. Цвет по умолчанию красный, #FF0000
vlink	Цвет уже посещенных ссылок. Цвет по умолчанию фиолетовый, #800080



Выполним пример создания цвета ссылок (рис. 4.5.5) на компьютере и сохраним файл, как **primer10.html** в своей папке.

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Цвет ссылок</title>
  </head>
  <body link="red" vlink="#cecece" alink="#ff0000"
    bgcolor = "olive">
    <p><h1><a href="content.html">Содержание сайта</a></h1>
  </p>
  </body>
</html>
```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.5.5. Изменение цвета гиперссылки



1. Что такое гиперссылка, и какова ее роль на web-страницах?
2. Перечислите и охарактеризуйте основные теги гиперссылок.
3. Каким образом осуществляется ссылка на местные web-страницы?
4. Каким образом осуществляется ссылка на любое место в WWW?
5. Как можно изменить цвет гиперссылки?
6. Как используется графика в качестве гиперссылки?
7. Как осуществляется переход по метке внутри одного HTML-документа?
8. Как осуществляется переход по метке, созданной в другом HTML-документе?



1. Создайте HTML-документ, отображающий на экране браузера информацию, показанную на рисунке 4.5.6, чтобы при наведении мыши на рисунок появлялось название посуды, а при нажатии на рисунок открывалась соответствующая ссылка.



Рис. 4.5.6. Вид web-страницы в окне браузера

2. Изображение на первой странице сделайте гиперссылкой: при нажатии на нее в новом окне должен открываться ее полноразмерный вариант. Сделайте задний фон страницы светло-желтым (#FFFFD9).

3. Измените файл **primer9.html** так, чтобы произведения Абая Кунанбаева и Мухтара Ауэзова стали гиперссылками на соответствующие страницы с произведениями этих писателей (найдите такие страницы в Интернете), еще непосещенным задайте оливковый цвет (#808000), а уже посещенным – красный (#FF0000).



Найдите ошибки в коде.

```
<html>
<head>
  <title>Ссылка<title>
</head>
<body>
  <p><a href=" images/sample.gif"><a></p>
</body>
</html>
```



Изучите полезные советы по созданию гиперссылок и познакомьтесь с изображениями-картами как разновидностью меню, т.е. изображениями, чувствительными к клику по кнопке мыши. Напишите эссе на тему «Карты ссылок».

## 4.6. Использование каскадной таблицы стилей (CSS)

Насколько важен внешний вид web-страниц?

– Что такое CSS, основные термины языка CSS;  
– что такое селекторы;  
– свойства и значения языка CSS;  
– какие бывают селекторы.

**Стиль** – Стиль – Style  
**Селектор** – Селектор – Selector  
**Свойства** – Қасиеттер – Properties  
**Значение** – Мән – Value  
**Класс** – Жік (Сынып) – Class

Для верстки web-страницы можно применять не одну, а сразу несколько таблиц, поэтому W3C (*World Wide Web Consortium – Консорциум Всемирной паутины*) рекомендует таблицы стилей называть «каскадными таблицами стилей».

вы узнаете

ключевые слова

это интересно

С помощью HTML можно определять содержимое и структуру web-страниц, но для создания визуального стиля и внешнего вида страницы лучше воспользоваться CSS.

Например, при применении CSS убирается подчеркивание у ссылок, границами таблицы могут быть пунктирные линии или меняется курсор мыши, шрифты, цвета задаются лишь один раз, а использовать их можно многократно. То есть для создания красивой и удобной web-страницы необходимо изучить язык CSS.

### ЗАПОМНИ

**CSS** (*Cascading Style Sheets*)

– технология управления внешним видом элементов web-страницы.

### ОСНОВНЫЕ ТЕРМИНЫ CSS

Основными терминами являются **селекторы**, **свойства** и **значения**. Чтобы определить, какие элементы HTML должны быть стилизованы, нужно на них указать и потом задать стиль.

Стиль состоит из набора правил отображения в парах **Свойство – Значение**, и к каким элементам их применять (**селектор**). Правила набираются внутри фигурных скобок и разделяются точкой с запятой. Свойство отделяется от значения двоеточием (рис. 4.6.1).

Рассмотрим эти термины более подробно.





Рис. 4.6.1. Структура синтаксиса CSS

## СЕЛЕКТОРЫ

При добавлении **элементов** на web-страницу они могут быть оформлены с помощью CSS. **Селектор** определяет, на какой именно элемент или элементы в HTML нацеливаться и применить к ним стили (такие как цвет, размер и положение). Селекторы могут включать в себя комбинацию различных показателей для выбора уникальных элементов.

Селекторы, как правило, связаны со значением атрибута (как значения **id** или **class**) или именем элемента (как **<h1>** или **<p>**).

В CSS селекторы сочетаются с фигурными скобками {}, которые охватывают стили, применяемые к выбранному элементу.

Например, мы хотим выбрать каждый абзац на странице или выбрать только один конкретный абзац. Селектор нацелен на все элементы **<p>**.

```
p { ... }
```

## СВОЙСТВА

Как только элемент выбран, свойства определяют стили, которые будут к нему применены. Имена свойств идут после селектора, внутри фигурных скобок {...} и непосредственно перед двоеточием. Например, это свойства, которые мы часто используем, такие как **background**, **color**, **font-size**, **height** и **width** и другие часто добавляемые свойства.

Например, определим свойства **color** и **font-size**, применяемые ко всем элементам **<p>**.

```
p { color: ...;
    font-size: ...; }
```

## ЗНАЧЕНИЯ

Чтобы задать поведение свойства, необходимо указать его **значение**. Значения определяются как текст между двоеточием и точкой с запятой.

Например, выбираем все элементы **<p>**, устанавливаем им оливковый цвет и задаем размер 16, т.е. значение свойства **color** как **olive**, а значение свойства **font-size** как **16** пикселей.

```
p { color: olive;
    font-size: 16px; }
```

## ПОДКЛЮЧЕНИЕ CSS

Чтобы подключить CSS с HTML, нужно указать из HTML на CSS-файл. Стили CSS включаются в HTML-документ тремя способами: как **внутренние** стили, как стили **уровня документа**, как **внешние** стили.

## ВНУТРЕННИЕ СТИЛИ

Когда нужно указать стили конкретного единственного элемента, используются внутренние стили. Внутренний стиль набирается в атрибуте **style** и относится только к содержимому этого тега. По приоритету внутренний стиль имеет более высокий уровень, чем остальные стили.

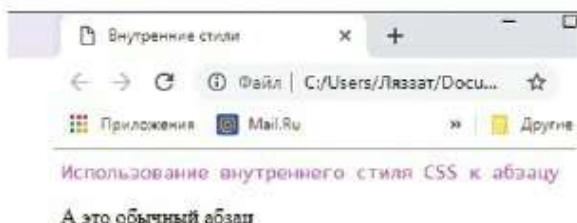


Выполним пример изменения размера, шрифта, цвета абзаца с помощью применения внутреннего стиля (рис. 4.6.2).

1. Откроем программу **Notepad++**.
2. Введем код программы на HTML, как на рис. 4.6.2, а.
3. Сохраним файл как **primer11.html** в своей папке.
4. Запустим **primer11.html**. Должно получиться, как на рис. 4.6.2, б.

```
<html>
<head>
  <meta charset="utf-8">
  <title>Внутренние стили</title>
</head>
<body>
  <p style="font-size: 120%; font-family: monospace;
color: #cd66cc">
    Использование внутреннего стиля CSS к абзацу</p>
  <p> А это обычный абзац </p>
</body>
</html>
```

а) Вид в **Notepad++**



### б) Вид в браузере

Рис. 4.6.2. Применение внутреннего стиля

## СТИЛИ УРОВНЯ ДОКУМЕНТА (ГЛОБАЛЬНЫЕ)

Стили применяются ко всему документу и записываются внутри тега `<style>...</style >`, который помещают в тег `<head>...</head>` в документе HTML.



Выполним пример указания двух стилей для пунктов списка `<li>`: один для случая, когда он находится в маркированном списке `<ul>`, и второй, когда он находится в нумерованном списке `<ol>` (рис. 4.6.3).

1. Откроем программу **Notepad++**.
2. Введем код программы на HTML, как на рис. 4.6.3, а.
3. Сохраним файл как **primer12.html** в своей папке.
4. Запустим **primer12.html**. Должно получиться, как на рис. 4.6.3, б.

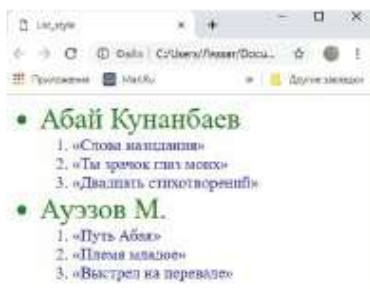
```
<html>
  <head>
    <meta charset="utf-8">
    <title>List_style</title>
  <style>
    ul li{font-size: 36px;color:green;}
    ol li{font-size: 20px;color:blue;}
  </style>
</head>
<body>
<ul>
<li>Абай Кунанбаев
```



```

<ul>
  <li>&laquo;Слова назидания&raquo;</li>
  <li>&laquo;Ты зрачок глаз моих&raquo;</li>
  <li>&laquo;Двадцать стихотворений&raquo;</li>
</ul>
</li>
<li> Мухтар Ауэзов
  <ul>
    <li>&laquo;Путь Абая&raquo;</li>
    <li>&laquo;Племя младое&raquo;</li>
    <li>&laquo;Выстрел на перевале&raquo;</li>
  </ul>
</li>
</ul>
</body>
</html>

```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.6.3. Стили списка

Такой способ указания стилей используется, когда нужно применить одинаковые стили сразу к множеству HTML-элементов (тегов) в одном документе.

## ВНЕШНИЕ (СВЯЗАННЫЕ) СТИЛИ

Стили сохраняют в отдельном файле с расширением **.css**. Для добавления данного стиля на страницу HTML включаем тег

```
<link rel="stylesheet" type="text/css" href="адрес_стиля">
```

Преимуществом внешних стилей является использование одного стиля сразу в нескольких страницах HTML.

Общие для всего сайта стили хранят во внешних файлах, но при включении в HTML-код данные стили влияют сразу на множество тегов в нескольких документах. Например, на всех страницах сайта надо поменять фон



и размер текста. Для этого создается один внешний стиль CSS, в котором задается новый фон и размер текста.

Если же CSS-файл находится на другом сайте, то нужно указать его полный URL-адрес.



Выполним пример использования класса в качестве селектора.

1. Откроем программу **Notepad++**.
2. Введем код программы на HTML, как на *рис. 4.6.4*.

```
<style>
.cll {text-decoration: underline; font-size: 80%}
A.cll {text-decoration: none;}
</style>
```

Рис. 4.6.4. Селектор-класс

3. Сохраним файл как **style1.css** (*рис. 4.6.5*).

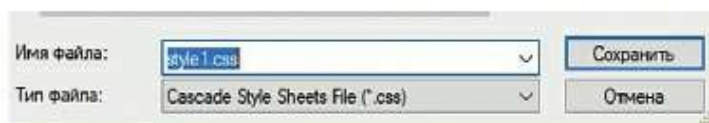


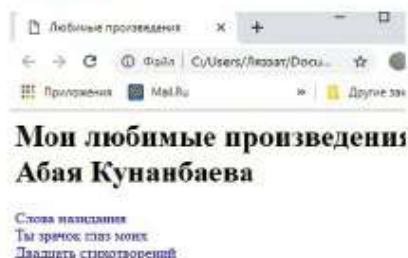
Рис. 4.6.5. Окно сохранения файла

4. Создадим саму страницу **primer13.html**.
5. Создадим новый файл в программе **Notepad++**.
6. Введем код программы на HTML, как на *рис. 4.6.6, а*.

```
<html>
<head>
  <meta charset="utf-8">
  <title> Любимые произведения</title>
  <link rel="stylesheet" type="text/css"
href="style1.css">
</head>
<body>
  <h1 class="cll">Мои любимые произведения Абая
Кунанбаева</h1>
  <a href="http://silkadv.com/ru/node/2001" class="cll">
Слова назидания</a>
```



```
<br>
<a href="https://www.stihi.ru/2012/03/05/2908"
class="c11"> Ты зрачок глаз моих</a>
<br>
<a href="http://abai-inst.kz/rus/?p=98" class="c11">
Двадцать стихотворений</a>
</body>
</html>
```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.6.6. Использование внешнего стиля

6. Сохраним файл как **primer13.html** в своей папке.

7. Запустим **primer13.html**. Должно получиться, как на рис. 4.6.6, б.

Когда браузер клиента откроет данную страницу, тогда он загрузит и файл **style1.css**, а также применит все созданные стили CSS к документу.

Таким образом, применение CSS дает возможность оформлять документ отдельно от его содержимого. В примере приемы оформления находятся в файле **style1.css**, а само содержание – в файле **primer13.html**. Такой способ очень упрощает редактирование сайта.



1. Почему таблицы стилей CSS называются каскадными?
2. Как задать значение свойству стиля? Приведите пример.
3. Какое свойство задает цвет фона?
4. Как в CSS обозначаются комментарии?
5. Как убрать подчеркивание для всех ссылок страницы?
6. Что означает свойство **CLEAR**?
7. Возможно ли смешать стили, например внутренний и внешний?
8. Как задается зеленый цвет для всех элементов, имеющих класс **GREEN**?



1. Разработайте внешний CSS-файл, в котором необходимо увеличить размер шрифта, задать для тега **body** фон со свойством **background-color** и границу толщиной **6 px**. Включите его во все страницы сайта.
2. Цвет фона главной страницы сделайте отличной от цвета других страниц сайта.
3. Для ссылок на внутренние страницы (в меню) и внешних ссылок создайте два разных класса стилей. В теги **<a>** на страницах добавьте атрибут **class**.



Какой цвет будет у текста на web-странице в данном примере?

```
<html>
  <head>   <title>Цвет текста</title>
    <style>
      HTML { color: black; }
      BODY { color: red; }
      P { color: green; }
    </style>
  </head>
  <body>
    <p style="color: blue;"><span style="color:
olive;">Текст</span></p>
  </body>
</html>
```



Отредактируйте файл **primer9.html** так, чтобы использовались три способа создания стиля для любых элементов страницы.

**Тестовые вопросы для самопроверки**

1. Какое стилевое свойство подойдет для задания цвета заголовка синим?  
**A)** font-color. **B)** Color. **C)** font-family. **D)** Text. **E)** font-size.
2. Что такое стиль?  
**A)** Способ сокращения HTML-кода за счет переноса части данных в другой файл.  
**B)** Язык разметки гипертекстовых документов.  
**C)** Набор правил форматирования элементов web-страницы.  
**D)** Метод преобразований текстовых документов в HTML.  
**E)** Технология, представляющая собой разные приемы для верстки HTML-кода.
3. Как расшифровывается аббревиатура CSS?  
**A)** Colorful Style Sheets.  
**B)** Cascading Style Sheets.  
**C)** Computer Style Sheets.  
**D)** Creative Style Sheets.  
**E)** Common Style Sheets.
4. Сайт имеет более ста HTML-страниц, имеющих одинаковое стилевое оформление. Какой способ подключения CSS подходит лучше всего?  
**A)** Связанные стили.  
**B)** Глобальные стили.  
**C)** Блочные стили.  
**D)** Внутренние стили.  
**E)** Экспорт стиля.
5. Какой HTML-код применяется для подключения внешнего CSS-файла?  
**A)** <style>mystyle.css</style>  
**B)** <style>@mystyle.css</style>  
**C)** <link rel="stylesheet" href="http://htmlbook.ru/mystyle.css">  
**D)** <link>@import url(mystyle.css)</link>  
**E)** <stylesheet>mystyle.css</stylesheet>
6. Какой атрибут используется для определения внутреннего стиля?  
**A)** Style. **B)** Class. **C)** Styles. **D)** Font. **E)** Link.



## 4.7. CSS-селекторы и свойства

Что вы знаете о профессии дизайнера?

– Какие бывают селекторы;  
– CSS-свойства: текст, размеры, шрифты, фон, цвета, границы, позиции.

**Шрифт** – Қарпін – Font  
**Позиция** – Позиция – Position  
**Границы** – Шекаралар – Borders

Согласно данным компании Netcraft, в настоящее время есть более 227 млн сайтов, которые содержат более 65 млрд web-страниц.

вы узнаете

ключевые слова

это интересно

Вспомним, что в стилях CSS пара **Свойство: Значение** задается в фигурных скобках, а перед скобкой указывается **селектор**, т.е. элемент, к которому их необходимо применять.

```
h6 { color: blue; }
```

*селектор      свойство: значение*

Используя свойства и значения, можно отформатировать текст любой сложности средствами CSS. В справочнике по атрибутам CSS можно найти нужные свойства (например, <http://htmlbook.ru>).

Особенность CSS заключается еще в том, что некоторые свойства наследуются от родительского элемента к дочернему. Например, если для тега **<body>** задано свойство **font-size**, то его наследуют все элементы на странице. Если свойство размера задается в процентах, то оно будет вычислено, исходя из значения для родительского элемента.

### КАКИЕ БЫВАЮТ СЕЛЕКТОРЫ

В качестве селектора используют:

– **Название тега.** В этом случае стиль будет применен ко всем таким тегам.

Рассмотрим пример. Нужно задать такой стиль для ссылок: 11-й размер шрифта и без подчеркивания.

```
A { font-size: 11pt;
    text-decoration: none; }
```

– **Несколько названий тегов через запятую.** Например, необходимо все заголовки сделать зелеными.

## 10 класс | информатика

```
h1, h2, h3, h4, h5, h6 { color: green; } /* делаем все
заголовки зелеными*/
/*...*/ – это комментарий.
```

– **Несколько названий тегов через пробел.**

Например, ссылка вложена в таблицу. Необходимо увеличить размер ее шрифта на 20% от базового.

```
TABLE A { font-size: 120%; }
```

– **Идентификатор.** Идентификаторы нацелены только на один уникальный элемент. В CSS идентификаторы обозначаются после знака #, затем в HTML-коде идентификатор используется как значение атрибута **id**, **id="идентификатор"**.

Например, идентификатор выберет только элемент, содержащий атрибут **id** со значением **home**.

```
#home { font-size: 200% }
```

В HTML-коде записываем **id="идентификатор"**

```
<div id="home">Соединим HTML и CSS</div>
```

Значение атрибута **id** может быть использовано только один раз на странице, независимо от типа отображаемого элемента. Желательно резервировать **id** для важных элементов.

– **Классы** позволяют выбрать элемент на основе значения атрибута **class**. Селекторы классов выбирают определенную группу элементов, а не все элементы одного типа.

Классы применяют одинаковые стили сразу к разным элементам, используя то же значение атрибута **class** для нескольких элементов.

В CSS классы обозначаются с точкой впереди, за которой следует значение атрибута класса.

```
.cl1 {text-decoration: underline; font-size: 80%}
```

В HTML-коде селектор класса выбирает все элементы, содержащие значение **cl1** атрибута **class**, включая **элементы <div>** и **<p>**.

```
<div class="cl1">...</div>
```

```
<p class="cl1">...</p>
```

Например, зададим цвет элементу. В качестве параметра может выступать как шестнадцатиричное, так и буквенное значение цвета.

```
.red { color: yellow}
```



Откроем **primer9**. Используя CSS, зададим заголовок со следующими параметрами: размер – 16 pt, полужирный, выравнивание по центру. Для остального текста – размер 12 pt, межстрочный интервал – полуторный, красная строка – 1,5 см. В качестве селектора используем класс (рис. 4.7.1) и сохраним файл, как **primer14.html** в своей папке.

```
<html>
<head>
<meta charset="utf-8">
<style>
.clh2 {color: Blue; text-decoration: none; font-size: 16
pt;
font - weight : bold;text - align: center;
}
.clp {font-size: 12 pt; line-height: 1.5;text indent:
20px;
}
</style>
</head>
<body>
<h2 class="clh2">Абай (Ибрагим) Кунанбаев (1845–1904)</
h2>
<p class="clp">
Великий казахский поэт, композитор, просветитель,
мыслитель, общественный деятель, основоположник казахской
письменной литературы и ее первый классик, реформатор
культуры в духе сближения с русской, а также европейской
культурой на основе просвещенного ислама.</p>
</body>
</html>
```

а) Вид в Notepad++

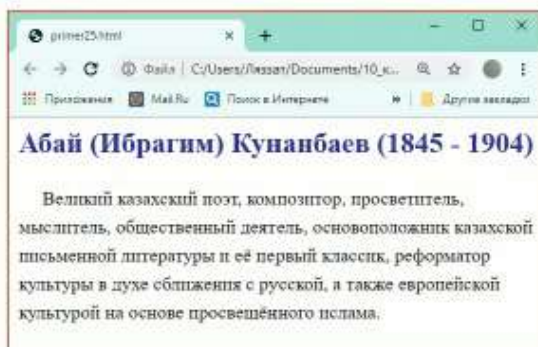


Рис. 4.7.1. Форматирование текста с CSS

### СОЗДАНИЕ, ИСПОЛЬЗОВАНИЕ И НАСТРОЙКА ОТОБРАЖЕНИЯ ФОНА В CSS

CSS предоставляет множество инструментов для работы с фоном (табл. 4.7.1), которые позволяют создать действительно уникальный и гармоничный для вашего сайта фон.

ТАБЛИЦА 4.7.1

#### CSS-свойства: фон, границы

Свойства	Описание значения	Синтаксис
background-color: ...	Заполняет элемент цветом. Значение <b>color</b> задает цвет элемента, а значение <b>transparent</b> – прозрачный фон	<b>.col {background – color : Red;}</b>
background-image: ...	Позволяет использовать изображение в качестве фона и принимает значения: – <i>none</i> – фоновое изображение не используется; – <i>url</i> – позволяет задать путь к изображению	<b>.cat {background – image: url(img/cat1.gif);}</b>
border	Соединяет параметры: – border-top – верхняя граница; – border-right – правая граница; – border-bottom – нижняя граница; – border-left – левая граница	<b>.bor5 {border-left-style:solid; border-right-style:solid; border-width:2px;}</b>



Свойства	Описание значения	Синтаксис
border-style: ...	Устанавливает стиль для границ. Значения: – solid – границы отображаются сплошной линией; – dashed – пунктирной линией; – dotted – двойной сплошной линией; – double – точками	<b>aside { border-style: solid dotted dashed double; }</b>
border-color: ...	Задаёт цвет границы	<b>.bor1 { border-style:solid; border-color:red;}</b>
border-width: ...	Задаёт толщину границы	<b>.bor2{ border-style:solid; border-width:2px;}</b>

Если объявлено только **одно** значение стиля, все четыре границы будут использовать этот стиль. Если **два стиля** – верхняя и нижняя границы будут использовать первый стиль, а правая и левая границы будут использовать второй стиль. Если **три стиля**, то верхняя граница будет использовать первый стиль, правая и левая границы будут использовать второй стиль, а нижняя граница будет использовать третий стиль.

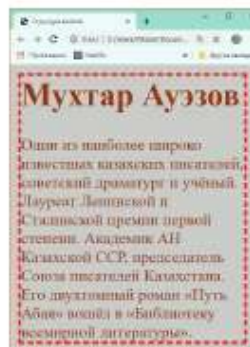


Выполним пример задания фона и границы текста (рис. 4.7.2) и сохраним файл как **primer15.html** в своей папке.

```
<html>
<head>
  <meta charset="utf-8">
  <title>Структура файлов</title>
  <style>
    body {
      background: #D3D3D3; /* Цвет фона */
      color: #8B4513; /* Цвет текста */
      border: 3px dashed RED; }
  </style>
</head>
```



```
<body>
  <h1>Мухтар Ауэзов</h1>
  <p> Один из наиболее широко известных казахских писателей, советский драматург и учёный. Лауреат Ленинской и Сталинской премии первой степени. Академик АН Казахской ССР, председатель Союза писателей Казахстана. Его двухтомный роман «Путь Абая» вошёл в «Библиотеку всемирной литературы».</p>
</body>
</html>
```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.7.2. Задание фона и границы текста

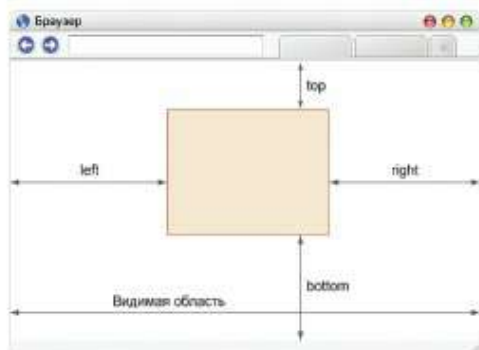
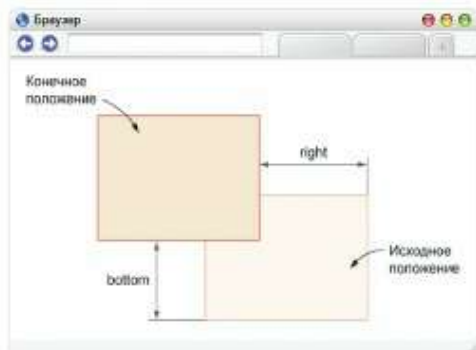
## CSS-ПОЗИЦИОНИРОВАНИЕ (POSITION)

Мощнейшим инструментом форматирования информации, расположенной на странице, является **Position**.

ТАБЛИЦА 4.7.2

### CSS-свойства: позиция

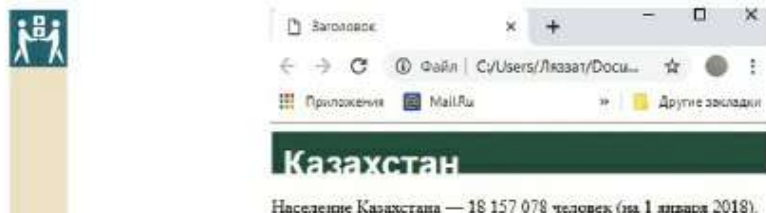
Свойства	Описание значения	Синтаксис
Position: ...	<p>Позволяет определить способ позиционирования элемента. Параметры значения:</p> <ul style="list-style-type: none"> <li>– <b>static</b> – позиционирование не применяется;</li> <li>– <b>absolute</b> – позиционирование элемента будет производиться в абсолютных величинах (рис. 4.7.3);</li> <li>– <b>relative</b> – позиционирование будет производиться относительно нормального положения элемента на странице (рис. 4.7.4);</li> <li>– <b>fixed</b> – задает жесткую позицию элемента</li> </ul>	<pre>pos {   position :   static; }</pre>

Рис. 4.7.3. Значения свойств (**absolute**)Рис. 4.7.4. Значения свойств (**relative**)

Выполним пример использования относительного позиционирования (рис. 4.7.5) на компьютере и сохраним файл как **primer16.html** в своей папке.

```
<html>
<head>
  <meta charset="utf-8">
  <title>Заголовок</title>
  <style type="text/css">
    H1 {
      font: bold 2em Arial, Tahoma, sans-serif;
      color: #fff; background: #375D4C;
      padding: 0 10px;
    }
    H1 SPAN {
      position: relative; /* Относительное позиционирование */
      top: 0.3em; /* Сдвигаем вниз */
    }
  </style>
</head>
<body>
  <h1><span>Казахстан</span></h1>
  <p>Население Казахстана – 18 157 078 человек (на 1
  января 2018). </p>
</body>
</html>
```

а) Вид в Notepad++



б) Вид в браузере

Рис. 4.7.5. Использование относительного позиционирования



1. Какое свойство нужно использовать, чтобы поместить линию под текстом каждого заголовка на web-странице?
2. Укажите разницу между записями **#N** и **.N**.
3. Назовите различие относительного позиционирования от абсолютного.
4. Где находится начало оси координат при абсолютном позиционировании?
5. Как вы понимаете понятия «родительский элемент» и «дочерний элемент»?
6. Айман хочет сделать сиреневый цвет фона web-страницы. Выберите правильное из трех значений свойства **background**, которое подойдет лучше всего (**#CBD1E8**; **#33F**; **#B6B7BE**).



1. Откройте файл **primer9.html** и отредактируйте его так, чтобы заголовков списка имел фон цвета **#7CFC00**, сами списки имели фон цвета **blue**, а цвет текста – **white**. Например, как на *рис. 4.7.6*.



Рис. 4.7.6. Пример оформления текста





2. Выполните задания.

- 1) На странице создайте сплошную границу коричневого цвета.
- 2) На странице создайте границу красного цвета слева и справа.
- 3) На странице создайте сплошную границу розового цвета сверху и пунктирную границу голубого цвета слева, справа и снизу.
- 4) На странице создайте границу оранжевого цвета толщиной один пиксель.
- 5) На странице создайте границу красного цвета толщиной два пикселя сверху и снизу и пунктирную границу серого цвета слева и справа.
- 6) На странице создайте сплошную границу цвета **#40E0D0** и толщиной три пикселя.
- 7) На странице создайте пунктирную границу цвета **#000080** и толщиной два пикселя снизу и сплошную границу цвета **#080000** и толщиной один пиксель сверху.
- 8) Поместите изображение в раздел заголовка. Измените высоту и ширину изображения и определите его положение в правом верхнем углу. Например:

```
img { position: absolute; left: 80%; }
```



1. Для цвета фона web-страницы и текста Тогжан установила соответствующие цвета **#B0C4DE** и **#A52A2A** и в стилях написала следующий код, но нужные цвета не проявились. Найдите ошибку.

```
body { background-color: #B0C4DE  
color: #A52A2A }
```

2. Найдите ошибку в следующем стиле.

```
img { float: right; border-width: 3; display:  
block }
```

3. Задан код **<p><span>Стили CSS</span></p>**. Каким будет размер текста, если задан следующий стиль?

```
BODY { font-size: 24pt; }  
P { font-size: 50%; }  
SPAN { font-size: 1.5em; }
```



Оцените, какой дизайн имеют страницы сайта вашей школы, и напишите, что вы сделали бы для их усовершенствования.

## 4.8.

## Разметка web-страницы

Как вы думаете, можно ли группировать объекты на web-странице?

– Что такое разметка web-страницы;  
– как создать нижний колонтитул, главное меню и прочие навигационные блоки.

**Верхний или нижний колонтитул** – *Үстіңгі немесе төменгі деректеме* – Header or footer

**Навигация** – Навигация – Navigation

**Блок** – Блок – Block

**Меню** – Мәзір – Menu

Основной причиной того, что главная страница Google имеет такой «голый» вид, является тот факт, что основатели Google... не знали языки разметки HTML и хотели быстро сделать простой интерфейс.

вы узнаете

ключевые слова

это интересно

Для создания полноценных web-страниц, которые обычно состоят из нескольких логически независимых частей, необходима разметка страницы.

**Разметка страницы** – это расстановка меток в коде web-страницы. При создании разметки и оформлении страницы ее разделяют на отдельные функциональные части.

Например, на страницах сайта, кроме основной информации, обычно присутствуют шапка, нижний колонтитул, главное меню и другие навигационные блоки (рис. 4.8.1).

Верхний и нижний колонтитулы обозначаются тегами **<header>** и **<footer>** соответственно.

**Верхний колонтитул** (шапка) предназначен обычно для заголовка, вводной информации о документе, навигационных ссылок, формы поиска, логотипа и т.п. Таким же образом **нижний колонтитул** включает информацию, которая размещается в конце документа. Например, данные об авторе, дату создания и другую подобную информацию.

## ЗАПОМНИ

## КОЛОНТИТУЛ

(фр. *colonne* – столбец и лат. *titulus* – надпись, заголовок)

– строка, расположенная на краю полосы набора и содержащая заголовок, имя автора, название произведения, части, главы, параграфа и т.д.

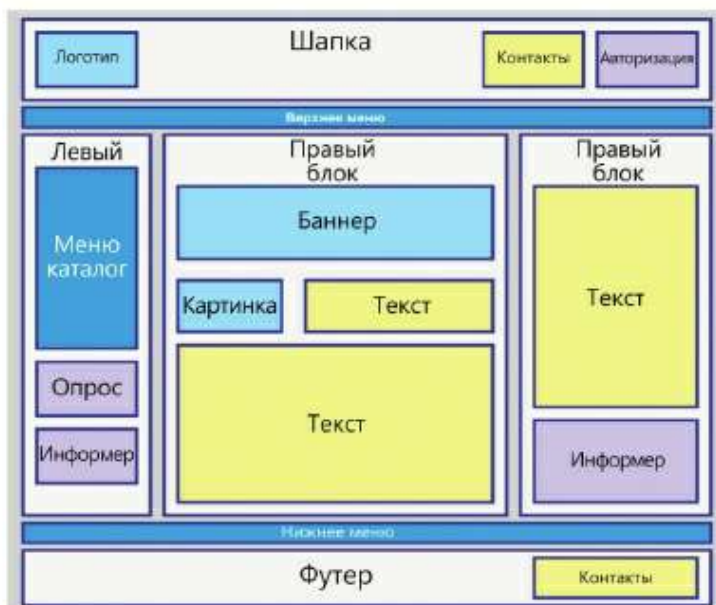


Рис. 4.8.1. Макет web-страницы

## ЗАПОМНИ

### КОНТЕНТ

– информационное содержание сайта (тексты, графическая, звуковая информация и др.), а также книги, газеты, сборника статей, материалов и др.

## ПРИКРЕПЛЕНИЕ ТАБЛИЦЫ СТИЛЕЙ CSS

Контент для блоков можно добавить между соответствующими тегами. А форму, расположение, цвет и другие внешние проявления блоков задаются с помощью файла CSS.

Представим, что имеются следующие селекторы: `body`, `container`, `header`, `naviga-`

`tion`, `menu`, `content`, `footer`.

Рассмотрим новые свойства:

- **margin** определяет местоположение блока. Точкой отсчета при этом может выступать как окно браузера, так и границы других элементов web-страницы;
- **width** – ширина блока;
- **height** – высота;

– **float** – свойство, которое позволяет позиционировать элементы селектора главным образом по левую или по правую сторону.



Выполним пример создания колонтитулов и навигационных ссылок на компьютере.

1. Создадим в **Notepad++** новый файл **style2.css**.

2. Теперь добавим код в файл стилей **style2.css**.

```
body { background: #f3f2f3;
        color: #000000;
        font-family: Trebuchet MS, Arial, Times New
Roman;
        font-size: 15px;}
#container { background: #FFFFFF;
margin: 30px auto;
width: 900px;
height: 1000px;}
#header {background: #838283;
height: 200px;
width: 900px;}
#navigation {background: #a2a2a2;
width: 900px;
height: 20px;}
#menu {background: #333333;
float: left;
width: 200px;
height: 550px;}
#content {background: #d2d0d2;
float: right;
width: 700px;
height: 550px;}
#footer {background: #838283;
height: 180px;
width: 900px;}
img {float:right; /* Выравнивание по левому краю */
margin: 7px 7px 7px 0; /* Отступы вокруг кар-
тинки */
border-style:solid;
border-color:blue;}
```





3. Создадим новый файл **primer17.html**.

4. Откроем программу **Notepad++**.

5. Введем код программы на HTML:

```
<html>
<head>
  <title>Разметка в HTML и CSS</title>
<link rel="stylesheet" type="text/css" href="style2.css"
/>
</head>
<body>
      <div id="container">
        <div id="header">
          <MARQUEE BEHAVIOR="slide"><font color=red>
            <font size = 30> <b> Наша гордость!</b></font></MARQUEE>
          </div>
          <div id="navigation">
<a href="#1">Кунанбаев </a>
<a href="#2"> Произведения Кунанбаева</a>
<a href="#3"> Ауэзов </a>
<a href="#4"> Произведения Ауэзова M</a>
          </div>
          <div id="menu">
<font color=white> 0 просветителях
<BR> Произведения</font>
          </div>
          <div id="content">
<dl>
      <font size = +2> <dt><a name="1"> Абай (Ибрагим) Кунан-
баев </a> (1845 - 1904)</dt></font>
      
      <font size = +1>
      <dd>
```

*Великий казахский поэт, композитор, просветитель, мыслитель, общественный деятель, основоположник казахской письменной литературы и ее первый классик, реформатор культуры в духе сближения с русской, а также европейской культурой на основе просвещенного Ислама.*



```

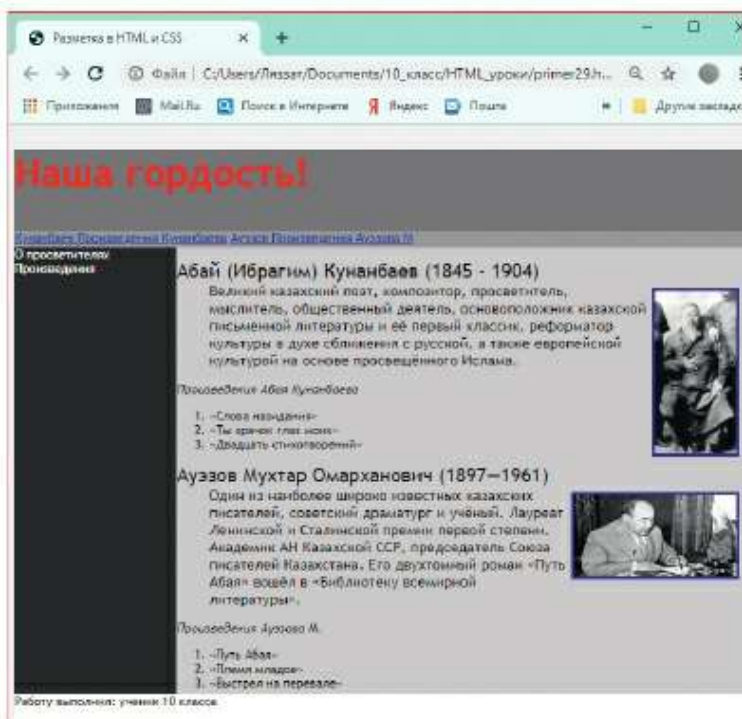
</dd> </font>
  <br><EM><a name="2">Произведения Абая Кунанбаева
</a> </EM>
  <ol>
    <li>&laquo;Слова назидания&raquo;</li>
    <li>&laquo;Ты зрачок глаз моих&raquo;</li>
    <li>&laquo;Двадцать стихотворений&raquo;</li>
  </ol>
  <dt> <font size = +2><a name="3">Ауэзов Мухтар Омар-
ханович </a> (1897–1961) </font></dt>
  
<font size = +1>
  <dd>
Один из наиболее широко известных казахских писателей,
советский драматург и ученый.
Лауреат Ленинской и Сталинской премии первой степени.
Академик АН Казахской ССР, председатель Союза писателей
Казахстана. Его двухтомный роман «Путь Абая» вошел
в «Библиотеку всемирной литературы».
</dd></font>
<br><EM><a name="4">Произведения Ауэзова М.</a> </EM>
</b>
  <ol>
    <li>&laquo;Путь Абая&raquo;</li>
    <li>&laquo;Племя младое&raquo;</li>
    <li>&laquo;Выстрел на перевале&raquo;</li>
  </ol>
</dl>
</div>
<div id="clear"></div>
<div id="footer">Работу выполнил: ученик 10 класса
</div>
</div>
</body>
</html>

```

## а) Код программы

6. Сохраним файл как **primer17.html** в своей папке.

7. Запустим **primer17.html**. Должно получиться, как на рис. 4.8.2, б.



### б) Вид в браузере

Рис. 4.8.2. Разметка страницы с CSS

## ПОЭКСПЕРИМЕНТИРУЙТЕ

Поэкспериментируйте с этим примером. Попробуйте изменить цвет, размер шрифта, цвет фона у блоков, размер блоков, поставить изображение слева и т.д.



1. Какую пару **Свойство – Значение** необходимо использовать для элемента, чтобы он был горизонтально центрирован в своем контейнере?
2. В **body** поместили контейнерный элемент со значением **width** больше 100%. Изменится ли поведение холста web-страницы?
3. Какое еще свойство необходимо задать для этого элемента, при применении значения **float** к элементу?



1. Создайте шаблон страницы (рис. 4.8.3) со следующими логическими частями: **HEADER** (шапка сайта), **SIDEBAR** (левая колонка), **CONTENT** (правая колонка), **FOOTER** (подвал).

2. Заполните шаблон своим содержанием, например, напишите о хобби, о любимой музыке, фильме, передаче и т.д.

3. Создайте разметку страницы с помощью таблицы шириной 800 пикселей, которая выровнена по центру экрана. Таблица состоит из четырех строк и одного столбца. Первая строка – это заголовок страницы, вторая строка – меню web-страницы, а в последней строке находится содержание сайта.

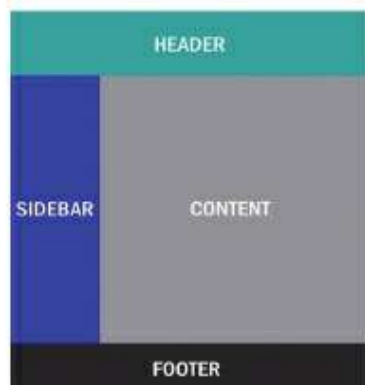


Рис. 4.8.3. Шаблон страницы



Найдите ошибку в коде.

```
<html>
<head>    <title>Блоки (div)</title>
<style>
  body    {margin: 10px;}
  div     {border- solid 1px black;}
  .top    (position: absolute; height: 50px width: 100%;)
  .right  {position: relative; top: 95px; right: 5px;
width: 150px; }
</style>
</head>
<body>
  <div class="" right"> RIGHT</div>
  <div class="" top">TOP
</body>
</html>
```



Разработайте небольшой web-сайт, который состоит из трех страниц. Разметку страниц производите с помощью блоков:





- пусть на страницах размещаются изображения;
- на страницах должны присутствовать ссылки на другие страницы сайта;
- наполнение для сайта можно найти в Интернете (на одну из данных ниже тем).
  1. «Алматы – южная столица».
  2. «О любимом спортсмене».
  3. «О любимой профессии».
  4. «О любимой музыкальной группе».
  5. «Об универсиаде Казахстана».
  6. Свободная тема, но предварительно обсудите ее с учителем.

## 4.9. Использование скриптов на web-странице

Как вы думаете, можно ли программировать действия на web-странице?

– Что такое скрипты;  
– варианты внедрения сценариев в HTML-код;  
– какой синтаксис у языка JavaScript;  
– понятие операторов.

**Скрипт** – Скрипт – Script  
**Интерактивный** – Интерактивті – Interactive  
**Окно** – Терезе – Window  
**Программа** – Бағдарлама – Program  
**Идентификатор** – Идентификатор – Identifier (ID)

Полноценный динамический язык программирования JavaScript разработал Брендан Айк, сооснователь проектов Mozilla, Mozilla Foundation и Mozilla Corporation.

вы узнаете

ключевые слова

это интересно

### ВНЕДРЕНИЕ СЦЕНАРИЕВ В HTML

Для того чтобы увеличить функциональность и иметь возможность взаимодействия с пользователями сайта в HTML, используются **скрипты**.

Для написания скриптов используются скриптовые языки, которые различаются по своему синтаксису, сферам применения и возможностям.

**JavaScript (JS)** – это язык сценариев, который позволяет сделать HTML-страницу **интерактивной**, т.е. «умеющей общаться» с пользователем (например, отклик при нажатии кнопок или при вводе данных в формы, динамические стили).

**ПРИМЕЧАНИЕ.** Иногда в браузере выполнение сценариев JavaScript отключено, и соответственно многие элементы дизайна недоступны. Как включить сценарии JavaScript (на Opera, Internet Explorer и Mozilla FireFox), описано на сайте <https://www.pc-shporgalka.com/read-article165.html>.

Чтобы поместить сценарий JavaScript в текст web-страницы, необходимо использовать дескриптор **<script>**. Но код JavaScript описывается между тегами **<script>** **</script>**. Код HTML внутри этих тегов размещать нельзя.

Закрывающий тег **</script>** указывает на то, что далее снова следует HTML-код.

Когда браузер считывает HTML-страничку и видит **<script>**, он первым делом считывает и выполняет код, а только после этого продолжает считывать страницу дальше.

Для добавления сценария на страницу HTML используется дескриптор **<script type="text/javascript">** **</script>**.

Атрибуты дескриптора:

- **language** в ранних версиях использовался для определения языка и версии;
- **type** – атрибут, пришедший на замену language. Он сообщает браузеру, какой язык используется внутри дескрипторов;
- **src** определяет URL внешнего исходного JavaScript-файла, сценарий которого прикрепляется к HTML-странице.

Включение JavaScript в HTML возможно двумя способами: включение в HTML-файл и создание внешних скриптов.

### ПОДКЛЮЧЕНИЕ JAVASCRIPT В HTML-ФАЙЛ

Первый способ, когда JavaScript-команды пишутся внутрь тега **<script>** между тегами **<body>** **</body>** (в теле страницы). **Script** также можно размещать и между тегами **<head>** **</head>** (в теле заголовка страницы).

### ЗАПОМНИ

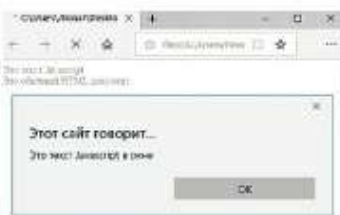
#### СКРИПТ

– это программа или программный файл, сценарий, автоматизирующие некоторую задачу, которую пользователь делал бы вручную, используя интерфейс программы.



Введем в **Notepad++** код программы на HTML, как на *рис. 4.9.1, а*. Сохраним файл как **primer18.html** в своей папке. Запустим **primer18**, должно получиться, как на *рис. 4.9.1, б*.

```
<html>
<head>
<meta charset="utf-8">
</head>
<body>
  <script type="text/javascript">
    document.write("Это текст Javascript");
    alert ("Это текст Javascript в окне");
  </script>
  <BR> Это обычный HTML документ
</body>
</html>
```



а) Вид в **Notepad++**

б) Вид в браузере

**Рис. 4.9.1.** Подключение JavaScript в HTML-файл

В этом примере использовались элементы из *табл. 4.9.1*.

### Описание функций JavaScript

**ТАБЛИЦА 4.9.1**

Функции	ОПИСАНИЕ
document.write()	Для вывода информации на экран
document.writeln()	Для перевода на новую строку, если используется тег форматирования <b>pre</b>
alert()	Выводит диалоговое окно с текстом сообщения, посетитель должен нажать <b>OK</b>
prompt()	Выводит диалоговое окно с заголовком ( <b>title</b> ), полем для ввода текста ( <b>default</b> ) и ждет. Пользователь должен либо что-то ввести и нажать <b>OK</b> , либо отменить ввод, нажав на <b>CANCEL</b> . <b>result = prompt(title, default);</b>

Функции	ОПИСАНИЕ
	Вызов <b>prompt</b> возвращает то, что ввел пользователь – строку или, если ввод отменен, специальное значение <b>null</b>
confirm()	Выводит диалоговое окно с вопросом ( <b>question</b> ) и ждет. Пользователь должен нажать <b>OK</b> либо не согласиться, нажав на <b>CANCEL</b> . При нажатии <b>OK</b> результатом будет <b>true</b> , а при <b>CANCEL (Esc)</b> – <b>false</b> . <b>result = confirm(question);</b>

### ВНЕШНИЕ СКРИПТЫ. ПОРЯДОК ИСПОЛНЕНИЯ

Второй способ – когда создается отдельный внешний файл с расширением **.js**. При прикреплении внешнего файла со скриптом тег **<script>** следует размещать в области **<head>**.

В HTML-документе помещается код:

```
<head>
<script src="myscript1.js"></script>
</head>
```

В файле с расширением **.js** (в данном конкретном примере **myscript.js**) код выглядит так:

```
document.write ("Это текст JavaScript!");
alert ("Это текст Javascript в окне ");
```

**ПРИМЕЧАНИЕ:** желательно, чтобы файл со скриптом находился в той же папке, что и HTML-файл, иначе при прикреплении в HTML-файле надо указывать полный URL-путь к файлу со скриптом.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Чтобы увидеть особенности работы метода **alert()**, выполните этот же пример (рис. 4.9.1), но поменяйте местами **alert** и **document.write**. Используйте внешний файл **myscript1.js**.
2. Посмотрите, как работает JavaScript, помещенный в теги оформления шрифта.



**ПРИМЕР.** Перенести скрипт в **BODY** после тега **H1**.

```
<h1>
<script type="text/javascript">
  alert ("Hello?");
  document.write("Hello!");
</script>
</h1>
```

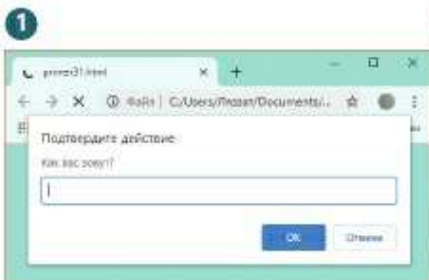


При загрузке страницы нужно запросить у пользователя ввести имя. Если **ОК**, то вывести на экран строку «Привет, введенное имя». Затем запросить «Вы действительно хотите завершить работу?». Если **ОК**, то вывести на экран строку «До свидания, введенное имя».

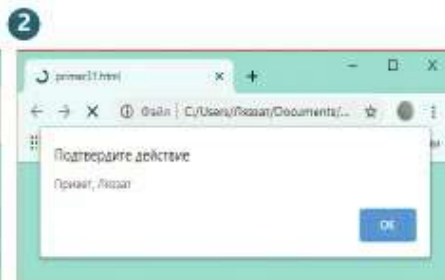
Введем в **Notepad++** код программы на HTML, как на *рис. 4.9.2, а*. Сохраним файл как **primer19.html** в своей папке. Запустим **primer19**, должно получиться, как на *рис. 4.9.2, б*.

```
<html>
<head>
<meta charset="utf-8">
</head>
<body>
  <script type="text/javascript">
    var name = prompt('Как вас зовут?', '');
    alert('Привет, ' + name);
    var result = confirm('Вы действительно хотите
завершить работу?');
    alert('До свидания, ' + name);
  </script>
</body>
</html>
```

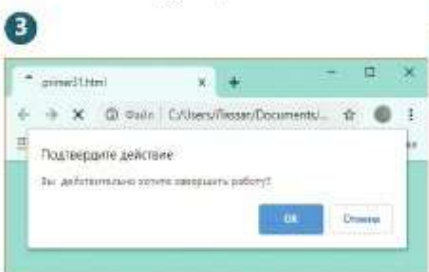
**а) Вид в Notepad++**



1 После ввода имени **Ляззат** и нажатия кнопки **OK**, появляется следующее окно



2 После нажатия кнопки **OK**, появляется следующее окно



3 После нажатия кнопки **OK**, появляется следующее окно



4 б) Вид в браузере

Рис. 4.9.2. Создание диалоговых окон

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Выполните данный выше пример, используя второй метод внедрения JavaScript в HTML, – внешний скрипт.



1. В каком месте HTML-документа находится стандартное расположение скрипта JavaScript?
2. С помощью каких методов осуществляется вывод диалоговых окон?
3. Что такое типы данных? Какие типы поддерживает JavaScript?
4. С помощью каких операторов создаются комментарии?
5. Какие операторы JavaScript вы знаете?



1. Добавьте фразу «Hello, дружище» в тело документа таким образом, чтобы содержимое, которое уже там находится, осталось на месте (код представлен ниже):

```
...  
<body><h1>Приветствие!</h1>  
</body>  
</html>
```

2. Выведите в окно браузера при помощи метода **alert()** следующие данные: ваши ФИО, возраст, хобби (каждое на новой строке: использовать специальные символы).

3. Какие значения выведет в окно браузера следующий фрагмент кода?

```
var str = "20";  
var a = 5;  
document.write(str + a + "<br/>");  
document.write(str - a + "<br/>");  
document.write(str * "2" + "<br/>");  
document.write(str / 2 + "<br/>");
```



Объявите две переменные разными способами и присвойте им значения. Выведите на экран значения переменных при помощи метода **alert()**.



Отредактируйте файл **primer17.html**, добавив туда диалоговые окна, как в **primer19.html**.

Используйте два способа встраивания JavaScript в HTML-код.

## 4.10. Внедрение мультимедиа на web-страницу

Как вы думаете, можно ли добавить звук на web-страницу?

– Как добавить фоновую музыку;  
– как добавить видео на web-страницу;  
– как управлять мультимедиафайлами, размещенными на web-страницах.

**Мультимедиа** – Мультимедиа – Multimedia  
**Аудио** – Аудио – Audio  
**Видео** – Видео – Video  
**Плагин** – Плагин – Plugin

Мультимедиа нашли свое широкое применение в науке, образовании, рекламе, искусстве, торговле, а также во многих других сферах деятельности человека.

вы узнаете

ключевые слова

это интересно

Мультимедиа – это различные аудио- и видеоролики, которые размещаются на web-страницах.

HTML5 позволяет использовать новый стандарт для размещения мультимедийных файлов в сети с программным интерфейсом без использования подключаемых модулей.

Формат медиафайла состоит из контейнера, содержащего один или несколько потоков данных, закодированных с использованием формата сжатия, называемого **кодеком**. Контейнер идентифицируется по расширению файла. Один контейнер (т.е. медиафайл) может содержать несколько потоков одного типа. В аудио- и видеопотоках находятся кодеки.

### ЗАПОМНИ

**КОДЕК** (сокращение слов *coder/decoder*

– шифратор/дешифратор) является алгоритмом сжатия данных в файле. Каждый тип контейнера имеет только определенные кодеки, которые он поддерживает.

### ДОБАВЛЕНИЕ ЗВУКА. ЭЛЕМЕНТ AUDIO

HTML5 предлагает парный тег **<AUDIO>** для вставки аудиоролика на web-страницу:

```
<AUDIO SRC="sound1.wav"></AUDIO>
```

Браузер, встретив тег **<audio>**, может действовать тремя способами:  
– загрузить и воспроизвести аудиофайл сразу;

216



- только загрузить его без воспроизведения;
- ничего не предпринимать.

Можно вывести на web-страницу элементы управления, как на плеере, чтобы с их помощью иметь возможность запускать воспроизведение аудиофайла, останавливать его, прокручивать назад или вперед, а также регулировать громкость. Рассмотрим различные атрибуты тега **<audio>** (табл. 4.10.1).

ТАБЛИЦА 4.10.1

Атрибуты тега **<AUDIO>**

Атрибут	Описание, принимаемое значение	Пример
<i>autoplay</i>	Воспроизводит аудиоролик. По умолчанию web-обозреватель не будет этого делать	<code>&lt;p&gt;Сейчас вы услышите звук!&lt;/p&gt; &lt;AUDIO SRC="sound.ogg" AUTOPLAY&gt; &lt;/AUDIO&gt;</code>
<i>controls</i>	Выведет в том месте web-страницы, где проставлен тег <b>&lt;AUDIO&gt;</b> , элементы управления воспроизведением аудиоролика. По умолчанию аудиоролик не отображается на web-странице	<code>&lt;p&gt;Нажмите кнопку воспроизведения, чтобы услышать звук.&lt;/p&gt; &lt;AUDIO SRC="sound.ogg" CONTROLS&gt; &lt;/AUDIO&gt;</code>
<i>loop</i>	Циклическое воспроизведение аудиофайла	
<i>muted</i>	Выключает звук при воспроизведении аудиофайла	
<i>src</i>	Содержит абсолютный или относительный URL-адрес аудиофайла	



Выполним пример вставки аудиоролика (рис. 4.10.1) на компьютере и сохраним файл как **primer20.html** в своей папке.

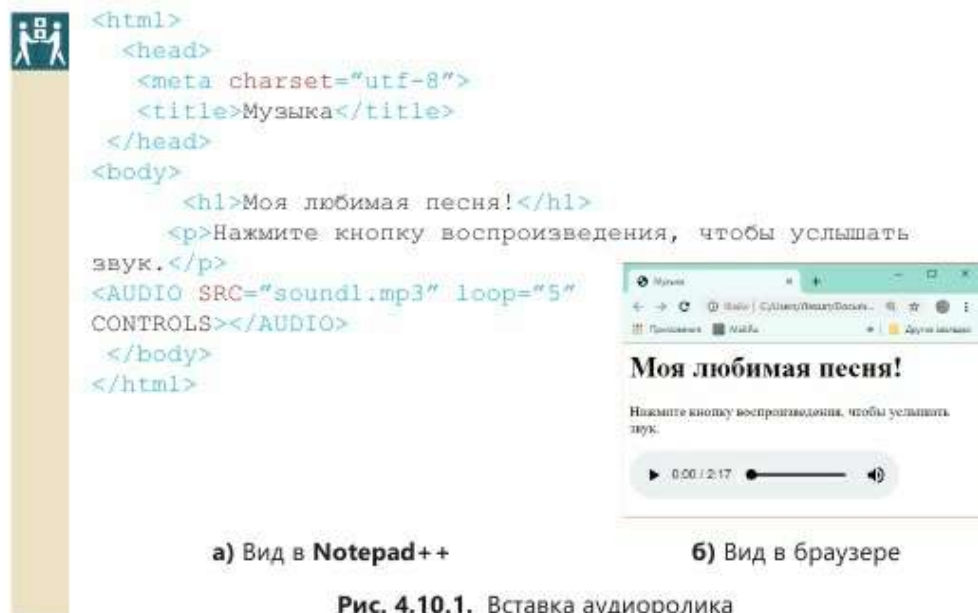


Рис. 4.10.1. Вставка аудиоролика

### АЛЬТЕРНАТИВНЫЕ МЕДИАРЕСУРСЫ

Различные браузеры поддерживают разные медиаформаты. Список основных форматов, которые используются в настоящее время, приведен в табл. 4.10.2.

Форматы аудио- и видеофайлов		
Формат	Видеокодек	Аудиокодек
.mp3		AAC
.ogg vorbis		Vorbis
.wav		Vorbis
.mp4	H.264	AAC
.ogg/.ogv	Theora	Vorbis
.webm	VP8	Vorbis



Вставим аудиоролик, который будет звуковым фоном (рис. 4.10.2) web-страницы, которая открывается на компьютере, и сохраним файл как **primer21.html** в своей папке.

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Музыка2</title>
  </head>
  <body>
    <h1>Музыкальное сопровождение</h1>
    <AUDIO SRC="sound1.mp3" AUTOPLAY></AUDIO>
  </body>
</html>
```



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.10.2. Вставка аудиоролика

## ДОБАВЛЕНИЕ ВИДЕО. ЭЛЕМЕНТ VIDEO

С помощью элемента **<video>** появилась возможность добавлять на web-страницы видеосодержимое.

HTML-код для размещения видеофайла на странице имеет следующий вид:

```
<video src="video.ogv" controls width="400" height="300"></video>
```

Атрибуты тега **<VIDEO>**

ТАБЛИЦА 4.10.3

Атрибут	Описание, принимаемое значение
<b>autoplay</b>	Автоматическое воспроизведение видеофайла сразу же после загрузки страницы
<b>controls</b>	Указывает браузеру, что нужно отобразить базовые элементы управления воспроизведением (воспроизведение, пауза, громкость)
<b>height</b>	Задаёт высоту окна для отображения видеоданных, возможные значения <b>px</b> или <b>%</b>
<b>loop</b>	Циклическое воспроизведение видеофайла
<b>src</b>	Содержит абсолютный или относительный URL-адрес видеофайла
<b>width</b>	Задаёт ширину окна для отображения видеоданных, возможные значения <b>px</b> или <b>%</b>

Видео в формате **.avi** на сайте средствами HTML не воспроизводится. Поэтому его необходимо перекодировать (табл. 4.10.2) с соответствующими видео- и аудиокодеками для вывода на сайте.

### ВСТАВКА АНИМАЦИИ

Чтобы загружать и отображать объекты, которые исходно браузер не понимает, используйте элемент **<object>**. Обычно такие объекты требуют подключения к браузеру специального модуля, который называется **плагин**, или запуска вспомогательной программы.



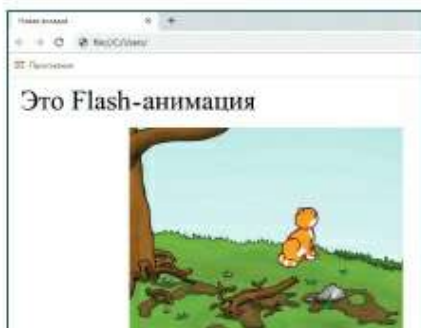
Выполним пример вставки flash-анимации. Найдем в Интернете готовую flash-анимацию и добавим ее в код программы (например, **Котенок.swf**) (рис. 4.10.3). Сохраним файл, как **primer22.html** в своей папке.

```
<html>
  <head>
    <title>Анимация</title>
    <meta charset="utf-8">
  </head>
  <body>
    <h1> Это Flash анимация</h1>
    <object width="480" height="200" align="center">
      <param name="movie"
        value="Котенок.swf">
      <param name="quality" value="high">
      <param name="bgcolor" value="#FFFFFF">
      <embed src="Котенок.swf"
        quality="high"
        bgcolor="#FFFFFF"
        width="480"
        height="200"
        align="center"
        type="application/x-shockwave-flash"

pluginspage="http://www.macromedia.com/go
/getflashplayer">
    </object>
  </body>
</html>
```

а) Вид в Notepad++





б) Вид в браузере

Рис. 4.10.3. Вставка flash-анимации

Разберем пример по использованным тегам.

Теги `<object>` `</object>` встраивают любой объект, при этом необходимо задать его URL.

Теги `<param>` задают начальные значения объекту. Параметры тега указаны в таблице 4.10.4.

ТАБЛИЦА 4.10.4

#### Параметры тега `<param>`

Параметры	Описание и значения
<code>&lt;param name="movie" value="..."&gt;</code>	Устанавливает качество отображения объекта. Значения: <b>high</b> (высокое), <b>medium</b> (среднее) и <b>low</b> (низкое)
<code>&lt;param name="quality" value="..."&gt;</code>	Устанавливает файл-источник объекта
<code>&lt;param name="bgcolor" value="..."&gt;</code>	Устанавливает фон для области отображения объекта

Тег `<embed>` используется, если старые версии браузера не поддерживают теги `<object>...</object>` (причины, почему параметры повторяются).

Параметры тега `<embed>`:

– **type** указывает MIME-тип встроенного объекта;

– **pluginspage** показывает URL-приложение, необходимое для просмотра встраиваемого объекта.



Выполним пример вставки видеоролика (рис. 4.10.4) на компьютере. Сохраним файл как **primer23.html** в своей папке.

```
<html>
<head>
<meta charset="utf-8">
</head>
<body>
  Видео, которое я создала сама</h1>
  <video src="film1.mp4" controls width="400"
    height="300">фильм...</video>
</body>
</html>
```



Видео, которое я создала сама



а) Вид в Notepad++

б) Вид в браузере

Рис. 4.10.4. Вставка видеоролика



1. Какие форматы аудио- и видеофайлов вы знаете? Чем они отличаются?
2. Какие плагины вы используете, чтобы смотреть видео, играть в компьютерные игры или слушать музыку?
3. Какой элемент надо использовать, чтобы музыка зазвучала при открытии web-страницы?
4. Можно ли воспроизвести видео на web-странице без кнопки **PLAY**? Если да, то как это сделать?
5. Какой формат аудио- и видеофайлов поддерживает браузер на вашем компьютере?
6. Какой плагин необходим для вставки flash-анимации?



1. Создайте фоновое звуковое сопровождение на web-странице.
2. Оформите **primer20.html** так, чтобы ваша любимая песня была в цветочной рамке или рамке в виде сердца и т.п.
3. Оформите **primer23.html** так, чтобы использовались отступы, обтекание созданной рамки было «вокруг текста» и т.п.



Найдите ошибку в коде.



```
<html>
  <head>
    <meta charset="utf-8">
    <title>Название документа</title>
  </head>
  <body>
    <video width="320" height="240" controls="controls"
poster="logo.png">
      <source src="movie/mp4" type=video/mp4">
      <source src="movie.ogg" type="video.ogg">
      Ваш браузер не поддерживает тег video.
    </video>
  </body>
</html>
```



Отредактируйте файл **primer17.html**, добавив туда аудиофайл – песню Абая «Ты – зрачок глаз моих» напротив списка произведений Абая, добавьте фильм «Путь Абая» напротив списка произведений М. Ауэзова. Используйте CSS для оформления.

## ТЕСТОВЫЕ ЗАДАНИЯ «WEB-ПРОЕКТИРОВАНИЕ»



Знание



Понимание



Применение



Анализ



Синтез



**1. В каком месте HTML-документа должны определяться теги <title>, <link> и <meta>?**

- A) Данные теги могут определяться в любом месте документа.
- B) В секции <body>.
- C) В секции <head>.
- D) И в секции <body>, и в секции <head>.
- E) В секции <style>.



**2. Какой тег позволяет вставлять картинки в HTML-документы?**

- A) <pic>.
- B) <img>.
- C) <picture>.
- D) <image>.
- E) <href>.



**3. Укажите CSS-свойство, позволяющее устанавливать размер шрифта.**

- A) font-weight.
- B) font-size.
- C) size.
- D) weight.
- E) height.



**4. Выберите ссылки с правильным синтаксисом.**

- A) <a href='http://www.wisdomweb.ru' value='wisdomweb.ru' />.
- B) <a name='http://www.wisdomweb.ru'>wisdomweb.ru</a>.





- C) `<a href='http://www.wisdomweb.ru'>wisdomweb.ru </a>`.  
 D) `<a src='http://www.wisdomweb.ru'>wisdomweb.ru </a>`.  
 E) `<a herf='http://www.wisdomweb.ru' title='wisdomweb.ru' />`.  
 F) `<a url='http://www.wisdomweb.ru'>wisdomweb.ru </a>`.



### 5. Укажите, какие теги используются в данном ниже коде.

Я первый **абзац**.

Я второй **абзац**.

- A) `<p>`, `<i>` и `<br />`.  
 B) `<p>`, `<b>` и `<hr />`.  
 C) `<div>`, `<b>` и `<hr />`.  
 D) `<div>`, `<i>` и `<br />`.  
 E) `<h1>`, `<h2>` и `<br />`.  
 F) `<span>`, `<b>` и `<hr />`.



### 6. Какие атрибуты использовались при создании данной таблицы?

Ячейка номер 1	Ячейка номер 2
Растянутая ячейка номер 3	

- A) Никаких.  
 B) `border='1'` и `rowspan='1'`.  
 C) `border='0'` и `rowspan='2'`.  
 D) `border='1'` и `colspan='2'`.



### 7. В какой строке содержится ошибка?

- A) `H1 { margin-left: 20px; }`  
 B) `p { margin-left: 20px; padding-left: 20px; }`  
 C) `h2 { margin-right: 20px; }`  
 D) `head { color: #rob; }`  
 E) `body { font-size: 11pt; color: #aaa; }`



**8. Установите соответствие. Какие CSS-свойства использовались для оформления границы элемента?**

1)		<b>A)</b> border-type и border-color.
2)		<b>B)</b> border-color, border-style, border-width.
3)		<b>C)</b> border-color, border-weight, border-type.
4)		<b>D)</b> border-type, border-style, border-width.
5)		<b>E)</b> border-type, border-style, border-height.



**9. Напишите основные теги HTML и CSS для создания цветного текста.**



**10. Размер окна браузера – 1000 пикселей. На страницу добавили блок шириной 40%. Затем в этот блок добавили таблицу шириной 50%. Какова будет ширина таблицы в пикселях?**

- A) 250 пикселей.
- B) Недостаточно пикселей.
- C) 500 пикселей.
- D) 200 пикселей.

## ТЕМЫ ДЛЯ ПРОЕКТНЫХ РАБОТ

1. «Казахстан-2050».
2. «Мой родной край».
3. «Самый дружный класс».
4. «Моя будущая профессия».
5. «Визитная карточка школы».
6. «Безопасность в Интернете».
7. «Польза и вред компьютера».
8. «Мой дом – моя крепость».
9. «Любимое животное».
10. «Мой бизнес-проект».

**ЦЕЛЬ:** разработать web-страницу в web-редакторе, внедрить объекты мультимедиа (графика, аудио, видео).

**ЗАДАЧИ:**

1. Определить, для чего предназначена ваша будущая web-страница (продажа чего-либо, обмен информацией о чем-либо и т.д.).
2. Продумать, что должна включать в себя ваша будущая web-страница для выполнения этой задачи.
3. Создать предварительный набросок вашей будущей web-страницы.
4. Перевести набросок в код на языке HTML.

## СТАДИИ РАБОТЫ НАД ПРОЕКТОМ



1. Использовать приобретенные знания по созданию web-страницы, по тегам HTML.
2. Выработать план действий.



1. Найти информацию по выбранной теме.
2. Найти графические картинки, рисунки для иллюстрации web-страницы.
3. Найти информацию для сравнения и анализа.



1. Подготовить web-страницу.
2. Реализовать принципы «хорошего» дизайна (удобство, простота и т.д.).



1. Подготовка и представление готовой web-страницы.
2. Дополнительные вопросы оппонентов.

## 5.1.

## Введение в базы данных

Какие информационные системы вы используете в повседневной жизни?

– Что такое реляционная база данных;  
– для чего используются системы управления базой данных;  
– что такое SQL.

**Реляционная база данных** – Қатырастық деректер қоры – Relational Database  
**Система управления базой данных (СУБД)** – Деректер қорын басқару жүйесі (ДҚБЖ) – Database Management System (DBMS)

Использование реляционных баз данных было предложено доктором Эдгаром Коддом из компании IBM в 1970 году.

вы узнаете

ключевые слова

это интересно

**ЗАПОМНИ****ИНФОРМАЦИОННЫЕ СИСТЕМЫ**

– в широком смысле – это аппаратные и программные средства, предназначенные для того, чтобы своевременно обеспечивать пользователей нужной информацией.

**ЗАПОМНИ****БАЗА ДАННЫХ (БД)**

– это специальным образом организованная совокупность данных о некоторой предметной области, хранящаяся во внешней памяти компьютера.

В нашу жизнь вошли информационные системы, с их помощью мы узнаем прогноз погоды, расписание поездов, определяем маршруты путешествий, заказываем билеты на самолеты, бронируем номера в гостиницах и т.п.

Информационная система должна обеспечивать:

- хранение данных;
- доступ к данным, т.е. возможность искать и изменять данные.

Массивы данных, с которыми работают информационные системы, имеют большой объем (нередко несколько гигабайт и даже терабайт) и размещаются во внешней памяти компьютера (например,

на сервере, в облаке). Данные хранятся в таком виде, чтобы их было легко искать и изменять. Такие наборы данных называются **базами данных**.

Эдгар Кодд предложил новую модель данных, основанную на следующих идеях:

228



1) все данные представляют собой свойства некоторых объектов;

2) объекты делятся на классы. Например, при описании данных музыкальных групп можно использовать следующие классы: **Группа**, **Альбом**, **Песня** и т.п.

**Данные о некотором объекте** – это набор свойств (атрибутов), в котором каждое свойство задается в виде пары **Название – Значение**. Например, сведения о музыкальной группе «Music 4U» можно записать так: (**Название:** Music 4U, **Вокалист:** Даниял Ануарбек, **Год создания:** 1987).

Модель данных, введенная Э. Коддом, получила название **реляционной модели данных** (англ. *relation – отношение*). Реляционные базы данных состоят из таблиц (рис. 5.1.1). Перечислим основные свойства таблицы в реляционных базах данных.

1. Каждая таблица описывает один класс объектов.
2. У каждого столбца есть уникальное имя (в пределах таблицы).
3. Порядок расположения столбцов в таблице не имеет значения.
4. Все значения одного столбца относятся к одному и тому же типу данных (число, текст, дата, ...).



Эдгар Кодд

## ЗАПОМНИ

### РЕЛЯЦИОННАЯ БАЗА ДАННЫХ

– это база данных, которую можно представить в виде набора таблиц.

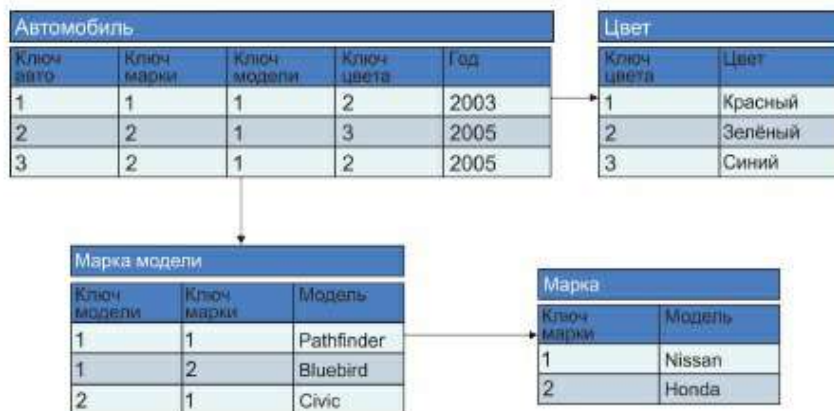


Рис. 5.1.1. Пример типичной реляционной базы данных

**ЗАПОМНИ****СИСТЕМА УПРАВЛЕНИЯ  
БАЗОЙ ДАННЫХ (СУБД)**

– это программное обеспечение (ПО), которое позволяет создавать БД, обновлять и дополнять информацию, обеспечивать гибкий доступ к информации.

5. В таблице нет двух одинаковых строк.

6. Порядок строк в таблице не определен.

Данные сами по себе бесполезны, если не уметь с ними работать. Поэтому необходимо специальное программное обеспечение, которое позволяет искать и изменять данные.

**ЧЕМ ОТЛИЧАЮТСЯ ПОНЯТИЯ «БАЗЫ ДАННЫХ» И «СУБД»?**

Хотя термины «базы данных» и «СУБД» обозначают различные понятия, они неразрывно связаны между собой: свойства базы данных определяются СУБД, которая ею управляет, и наоборот. Комплекс «БД + СУБД» называется **системой базы данных** (англ. *database system*), или **информационной системой** в узком смысле.

СУБД решает все задачи, связанные с управлением данными, в том числе:

- поиск данных;
- редактирование данных;
- выполнение несложных расчетов;
- обеспечение целостности (корректности, непротиворечивости) данных;
- восстановление данных после сбоев.

**ЗАПОМНИ****SQL** (*Structured Query Language*)

– структурированный язык запросов)  
– язык управления базами данных для реляционных баз данных.

Приведем примеры наиболее известных систем управления реляционными базами данных (табл. 5.1.1). При работе будем использовать MySQL.

Не любой набор таблиц можно считать реляционной базой данных. Как мы уже говорили, БД и СУБД неразрывно связаны, и для того, чтобы отнести систему базы данных (БД + СУБД) к определенному типу, необходимо

выяснить, какие методы управления данными используются в соответствующей СУБД.

Для управления данными в большинстве современных информационных систем используется язык SQL, в который включены команды для:

- создания новых таблиц;

ТАБЛИЦА 5.1.1

### Наиболее известные системы управления реляционными базами данных

				
<b>Oracle</b> используется преимущественно для профессиональных, больших приложений	<b>Microsoft SQL Server</b> – СУБД компании Microsoft. Доступна только для операционной системы Windows	<b>MySQL</b> – популярная свободно распространяемая СУБД с открытым исходным кодом	<b>IBM</b> имеет ряд СУБД, из которых наиболее известна <b>DB2</b>	<b>MS Access</b> – СУБД, которая позволяет создавать базы данных с пользовательским интерфейсом

- добавления новых записей;
- изменения записей;
- удаления записей;
- выборки записей из одной или нескольких таблиц в соответствии с заданным условием и некоторые другие.

Команды языка SQL позволяют управлять данными, не «привязываясь» к формату их хранения, т.е. к порядку расположения столбцов и строк в таблицах. Для выполнения операций (выборки, вставки, удаления, изменения) используются только названия столбцов и таблиц. С помощью команд SQL можно выполнить все основные операции, введенные Коддом, поэтому СУБД (и соответствующие системы баз данных), которые используют язык SQL, называют реляционными.



1. Что такое информационная система? Из каких компонентов она состоит?
2. Что такое база данных? Какими свойствами она должна обладать? Является ли базой данных бумажная картотека в библиотеке? Ответ обоснуйте.
3. Что такое реляционная база данных? Почему она так называется?
4. В каком виде хранится информация в реляционной базе данных?
5. Что такое СУБД?
6. Приведите примеры СУБД.





Преобразуйте данную ниже информацию в таблицу «Страны Центральной Азии».



**Республика Казахстан.** Площадь территории республики – 2 724 902 км<sup>2</sup>, занимает 9-е место в мире по территории. Климат в республике в основном резко-континентальный. Население – 18 157 078 человек (на 1 января 2018 г.). Столица – Нур-Султан. Государственный язык – казахский. Язык межнационального общения, используемый в стране наряду с государственным, – русский.



**Кыргызская Республика.** Площадь территории – 198 500 км<sup>2</sup>, занимает 85-е место в мире по территории. Климат резко-континентальный. Население – 6 256 700 человек (на 1 января 2018 г.). Столица – Бишкек. Государственный язык – кыргызский. Официальный язык, используемый в стране наряду с государственным, – русский.



**Республика Узбекистан.** Площадь территории – 447 400 км<sup>2</sup>, занимает 56-е место. Климат резко-континентальный, жаркий и засушливый. Население Узбекистана на 1 января 2019 года составило 33 254 100 человек. Столица – Ташкент, с населением более 2 миллионов человек.



**Туркменистан.** Площадь территории республики – 491 210 км<sup>2</sup>, занимает 53-е место в мире. Климат континентального типа, характеризуется засухами. Население – 5 758 075 человек (по данным 2017 года). Столица – Ашхабад. Государственный язык – туркменский.



**Республика Таджикистан.** Площадь территории республики – 141 400 км<sup>2</sup>. Климат субтропический. По данным переписи населения





2010 года численность населения Республики Таджикистан составила 7 417 400 человек. **Столица** – город Душанбе. Государственным и официальным языком Таджикистана признан таджикский язык.



1. Чем отличаются система управления базами данных и собственно базы данных? Заполните диаграмму Венна.
2. Проведите исследование на тему «Реляционная база данных». Какие виды баз данных, кроме реляционной, существуют? Приведите примеры.

## 5.2. Основные понятия базы данных. Проектирование базы данных

– Основные понятия базы данных;  
– о проектировании структуры БД.

**Поле** – Әріс – Field  
**Запись** – Жазба – Record  
**Индекс** – Индекс – Index  
**Первичный ключ** –  
Алғашқы кілт –  
Primary Key

Термин «база данных» (англ. *database*) появился в начале 1960-х годов. В широкое употребление в современном понимании термин вошел лишь в 1970-е годы.

вы узнаете

ключевые слова

это интересно

Из предыдущего урока вы знаете, что данные, хранящиеся в современных базах данных, чаще всего удобно представлять в виде таблиц. Например, так называемый «список адресов» может выглядеть, как показано на рис. 5.2.1.

Name	Address
Шыныбек Даниал	Пр. аль-Фараби, 5
Рыздыбаев Муса	Пр. Достык, 1
Нағашыбай Али	Ул. Кунаева, 11

Рис. 5.2.1. Таблица в реляционной базе данных

**ЗАПОМНИ**

Столбцы таблицы называются **полями**, а строки – **записями**.

Реляционная база данных представляет собой множество взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного вида. Каждая строка таблицы содержит данные об одном объекте (например, автомобиле, компьютере, клиенте), а столбцы таблицы содержат различные характеристики этих объектов – атрибуты (например, номер двигателя, марка процессора, телефоны фирм или клиентов).

**ЗАПОМНИ**

Любое поле должно иметь уникальное (неповторяющееся) имя. Например, нельзя назвать два поля **Адрес**, но можно одно назвать **Адрес**, а второе – **Временный адрес**.

Таблица на *рис. 5.2.1* относится к типу **Объект – Свойства**, т.е. запись – это описание некоторого объекта (в данном случае – человека), а поля содержат свойства этого объекта. В этой таблице два поля (**Name, Address**) и три записи.

Представим себе, что в таблице с адресами, о которой говорилось выше, записаны данные большого количества людей (в реальных базах данных могут быть миллионы записей).

На практике данные нужно искать по нескольким полям в каждой таблице и нет возможности каждый раз сортировать записи (это очень долго для больших баз). Как же в такой ситуации обеспечить быстрый поиск?

В программировании в данном случае можно увеличить скорость работы алгоритма за счет расхода дополнительной памяти.

**ЗАПОМНИ****ИНДЕКС**

– это вспомогательная таблица, которая служит для ускорения поиска в основной таблице.

Фамилия	№
Шыныбек	3
Рыздыбаев	2
Нағашыбай	1

**Рис. 5.2.2.** Индекс по полю **Фамилия**

Для ускорения поиска во многие книги включают **индекс** – список ключевых слов с указанием страниц, где они встречаются. В базах данных специально для поиска создаются дополнительные таблицы, которые также называются индексами.

Теперь, если нам нужны люди с фамилией **Шыныбек**, мы можем искать номера **Шыныбек** в индексной таблице, используя быстрый **двоичный** поиск (там фамилии стоят по алфавиту!). Затем, когда номера нужных записей определены, данные **Шыныбек** можно взять из основной таблицы. Заметим, что искать что-то в основной таблице уже не нужно. Таким образом, можно использовать двоичный поиск по разным полям.

Как вы думаете, что такое уникальное поле?

При чтении и изменении данных в таблице очень важно убедиться, что мы обращаемся именно к нужной записи. Это означает, что для надежной работы каждая запись должна содержать какое-то уникальное значение, отличающее ее от всех остальных.

Это значит, что ключ обладает свойством **уникальности**: в таблице не может быть двух записей, у которых одинаковое значение ключа. Например, ключом может быть номер удостоверения личности, номер мобильного телефона, адрес электронной почты и т.п. (рис. 5.2.3).

## ЗАПОМНИ

### ПЕРВИЧНЫЙ КЛЮЧ, или КЛЮЧ

– это одно или несколько полей, совокупность значений которых однозначно определяет каждую запись в таблице. Наличие ключа в таблице исключает возможность появления в ней двух одинаковых записей.

Ключевое поле

№	Фамилия	Имя	Класс	Дата рождения
1	Асанов	Дамир	10	11.06.2002
2	Мырзахметова	Арайлым	11	02.10.2001
3	Фетисов	Александр	10	24.02.2002

Рис. 5.2.3. Ключевое поле



## 10 класс | информатика

Использование ключа дает следующие преимущества:

- уникальность записей не позволяет вводить и хранить в таблице записи, имеющие одинаковое значение в ключевом поле;
- связи. Используя ключ, устанавливаются связи между таблицами;
- скорость. Создается индекс по ключевому полю, ускоряющий поиск нужных записей и выполнение запросов;
- упорядочение. Автоматически сортирует и отображает записи таблицы в порядке возрастания или убывания значений в ключевом поле.

#### ПОЭКСПЕРИМЕНТИРУЙТЕ

Постройте вручную индексы по полю **Адрес** (рис. 5.2.1).

Создание БД начинается с ее проектирования. Процесс проектирования БД включает следующие основные этапы.

#### 1. Определение назначения БД.

Необходимо определить список задач, решаемых с ее помощью, и какие данные для этого нужны.

#### 2. Определение структуры таблиц:

1) каждая из таблиц должна содержать информацию о наборе однотипных объектов, например сведения об учениках или итоги сдачи экзаменов;

2) каждому из таких наборов данных должна соответствовать отдельная таблица. Например, сведения об учениках и о полученных ими оценках на экзамене должны храниться в разных таблицах. Тогда при удалении сведений об оценках ученика информация останется в БД;

3) информация в таблице не должна дублироваться. Не должно быть повторов и между таблицами. Это исключает возможность несовпадения информации в разных таблицах и делает работу с БД более эффективной.

3. **Определение полей.** Каждая таблица содержит информацию о наборе объектов одного типа. При разработке полей для таблицы необходимо помнить следующее:



- 1) в таблице должна присутствовать вся необходимая информация о данном наборе объектов;
- 2) каждое поле должно содержать сведения о том или ином свойстве именно этого, а не другого набора объектов.

#### 4. Определение ключевых полей.

Для того чтобы БД могла связать данные из разных таблиц, каждая таблица должна содержать первичный ключ, или просто ключ.

#### 5. Определение связей между таблицами.

После распределения данных по таблицам и определения ключевых полей необходимо выбрать схему для связи данных в разных таблицах. Для этого нужно определить связи между таблицами. Эти связи используют при создании многотабличных запросов, форм и отчетов.

#### 6. Выбор программных инструментов для создания БД.



Спроектируем базу данных казахских музыкальных инструментов. Для этого необходимо:

- 1) Определить задачу: «Для чего нужна база данных казахских музыкальных инструментов?».
- 2) Спроектировать структуру таблицы (например, «Сведения о музыкальных инструментах»).
- 3) Определить названия полей («Название музыкальных инструментов», «Типы музыкальных инструментов», «Фото музыкальных инструментов» и т.д.).
- 4) Определим первичные ключи.



1. Что означают термины «поле» и «запись»?

2. Что такое индекс? Для чего он используется? Можно ли создать базу данных без индекса?



3. Объясните принцип поиска с помощью индекса.

4. Можно ли для одной и той же таблицы построить несколько индексов?



5. Что такое ключ таблицы?
6. Чем отличаются понятия «ключ» и «первичный ключ»?
7. Назовите этапы проектирования базы данных.



1. Спроектируйте базу данных учащихся вашего класса, которая отвечает на следующие вопросы:
  - 1) Сколько мальчиков в классе?
  - 2) Укажите фамилии учеников, которые родились в марте.
  - 3) Укажите имена учеников, которые записаны в кружки по рисованию.
2. Выберите названия, которые могут быть полями: **Фамилия, Возраст, Пение, Хобби, Адрес, Алматы, Футбол.**
3. В таблице четыре поля: **Дата, Номер заказа, Название товара** и **Количество**. Какие индексы можно построить?
4. В таблице четыре поля: **Дата, Номер заказа, Название товара** и **Количество**. Что можно выбрать в качестве первичного ключа?



Постройте индекс по полю **Название блюда** для таблицы 5.2.1.

Восточные блюда				
№	Дата	Заказ	Название блюда	Количество
1	10.04.2018	25	Бешбармак	9
2	11.04.2018	26	Куырдак	15
3	11.04.2018	27	Сырне	10
4	12.04.2018	28	Плов	20
5	13.04.2018	29	Лагман	15

## 5.3. Разработка однотабличной базы данных в MySQL

Как подготовить структуру базы данных?

– Как создать базу данных в MySQL;  
– как определять типы данных.

**Создание базы данных** – Деректер қорын құру – Create Database  
**Создание таблицы** – Кесте құру – Create Table

Построение космического корабля включает в себя сборку нескольких миллионов деталей. Столкнувшись с задачей координации заказов миллиона деталей, компания Rockwell в сотрудничестве с IBM в 1968 г. разработала систему управления информацией (IMS), которая заложила основу концепции СУБД.

вы узнаете

ключевые слова

это интересно

### ПОДГОТОВКА КОМПЬЮТЕРА ДЛЯ РАБОТЫ С БАЗОЙ ДАННЫХ

В этом разделе мы научимся разрабатывать базу данных и устанавливать связь web-страницы с базой данных. Поэтому нам потребуется устанавливать на компьютере не только визуальный интерфейс для работы с СУБД MySQL, но и Apache и PHP. Данные инструменты собраны в одном установочном файле под названием **Open Server** ([www.ospanel.io](http://www.ospanel.io)).



ТАБЛИЦА 5.3.1

#### Средства, входящие в Open Server для разработки базы данных и установления связи web-страницы с базой данных



Очень популярный визуальный интерфейс для работы с MySQL. Позволяет значительно упростить работу с базами данных в MySQL. Многие из базовых SQL-функций работы с данными в PhpMyAdmin сведены к интуитивно понятным интерфейсам и действиям, напоминающим переход по ссылкам в Интернете



	<p>Apache – это HTTP-web-сервер – специальное приложение сервера, предназначенное для передачи данных в сети Интернет. При поступлении в Сети к такому серверу запроса от клиента, отвечает именно Apache-сервер. Он поддерживает следующие ОС: Linux, BSD, Mac OS, Microsoft Windows. Преимущества web-сервера Apache: поддержка языков программирования PHP, Python, Ruby, Perl, ASP. В нашем случае локальный компьютер будет служить и сервером, и клиентом</p>
	<p>PHP (<i>Hypertext Preprocessor</i> – препроцессор гипертекста) – язык программирования, специально разработанный для написания web-приложений (сценариев), исполняющихся на web-сервере. Преимущество языка PHP заключается в возможности создания HTML-документов с внедренными командами PHP</p>

Процесс установки и первоначальная настройка на локальном компьютере, работающем под операционной системой Windows 7, смотрите в рубрике «Дополнительная информация».

После успешной установки **Open Server** откройте **PhpMyAdmin**. Главное окно **PhpMyAdmin** выглядит, как на *рис. 5.3.1*.

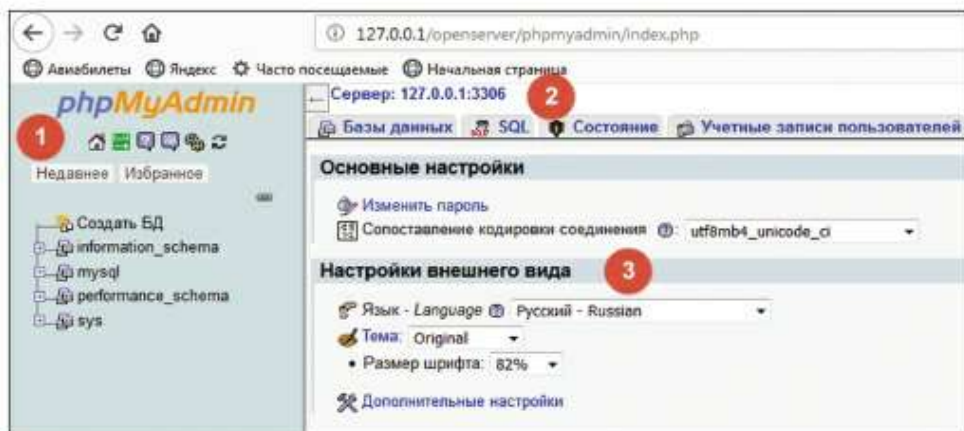


Рис. 5.3.1. Главное окно **PhpMyAdmin**



1. На **боковой панели** отображается вертикальный список доступных в данный момент баз данных, а также их таблиц. Можно развернуть каждую из них, чтобы посмотреть таблицы. Также тут расположены быстрые кнопки **Обновить**, **Домой** и т.д.

2. **Главное меню** – это полоска, которая находится под именем сервера. С помощью нее можно перемещаться по различным возможностям программы, таким как базы данных, SQL, пользователи, экспорт и импорт и так далее.

3. **Рабочая область**. Здесь отображается рабочая информация. В данный момент в главном окне находится информация о сервере, в других случаях может находиться содержимое базы данных или поле ввода текста.



Создадим базу данных **Метеостанция**. Для этого нажмем первую вкладку меню **Базы данных**. Здесь можно увидеть список доступных вам баз данных, а также поле для создания новой базы над ними. Введем в это поле имя для новой базы **meteo\_station**, а затем выберем кодировку **utf8-general-ci** (рис. 5.3.2). Нажмем кнопку **Создать**.

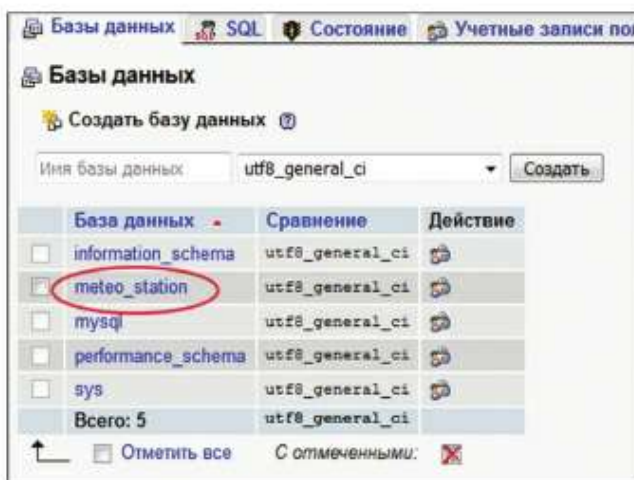


Рис. 5.3.2. Создание базы данных

Мы создали базу данных с помощью графического режима. Теперь попробуем создать БД в текстовом режиме с помощью SQL-команд. SQL-команды вводятся во вкладке **SQL** (phpMyAdmin).

## СОЗДАНИЕ ОДНОТАБЛИЧНОЙ БАЗЫ ДАННЫХ В MYSQL

**CREATE DATABASE** – это SQL-команда для создания базы данных.

```
CREATE DATABASE name_database;
```

Предположим, что нужно создать базу данных с названием **musical\_instruments**. Для этого выполним следующую команду SQL (рис. 5.3.3).

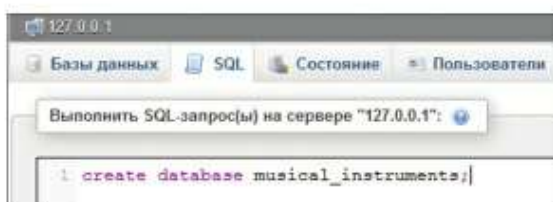


Рис. 5.3.3. Ввод SQL-команд

## СОЗДАНИЕ ТАБЛИЦ MYSQL

Таблицы могут быть созданы с использованием инструкции **CREATE TABLE** и фактически имеют следующий синтаксис:

```
CREATE TABLE `TableName` (`fieldname` dataType [дополнительные параметры]);
```

– **CREATE TABLE** является ответственной за создание таблицы в базе данных;

– **TableName** – имя создаваемой таблицы;

– **fieldName** – это имя поля;

– **dataType** – это тип данных, определяющий характер данных, которые должны храниться в поле;

– **[дополнительные параметры]** – дополнительная информация о поле, такая как **AUTO\_INCREMENT**, **NOT NULL** и т.д.

Здесь **AUTO\_INCREMENT** – автоматическая нумерация, **NOT NULL** – непустое значение.

MySQL имеет три основные категории данных:

1. **Числовой;**
2. **Текст;**
3. **Дата/время.**

## Числовые типы данных

INT ()	–2 147 483 648 до 2 147 483 647 нормальный 0 до 4 294 967 295 НЕОГРАНИЧЕН
FLOAT	Небольшое приблизительное число с плавающей запятой

## Типы текстовых данных

CHAR ()	Фиксированный раздел длиной от 0 до 255 символов
VARCHAR ()	Переменный раздел длиной от 0 до 255 символов

## Дата/время

DATE	Тип данных для хранения даты
TIME	Тип данных для хранения времени
YEAR	Тип данных для хранения года

В MySQL существуют **некоторые другие типы данных**.

BOOL	Синоним <b>TINYINT (1)</b> используется для хранения логических значений
------	--



Создадим в уже имеющейся базе данных **musical\_instruments** таблицу **instruments** с полями **id** (код), **name** (название), **type** (тип), **data** (дата создания), **num\_string** (количество струн), **photo** (название файла с фото).

```
create table instruments (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name varchar(20),
  type varchar(20),
  data date,
  num_string int(10),
  foto varchar(20)
);
```

## ЧТО ТАКОЕ INSERT INTO?

Основная задача систем баз данных – хранить данные в таблицах. Данные обычно предоставляются прикладными программами, которые рабо-

тают поверх базы данных. С этой целью SQL имеет команду **INSERT**, которая используется для хранения данных в таблице.

#### Основной синтаксис

```
INSERT INTO `table_name`
VALUES (значение_1, значение_2, ...);
```

– **INSERT INTO table\_name** – это команда, которая сообщает серверу MySQL добавить новую строку в таблицу с именем **table\_name**.

– **VALUES (value\_1, value\_2, ...)** указывает значения, добавляемые в новую строку.

При определении значений данных, которые необходимо вставить в новую таблицу, при рассмотрении разных типов данных следует учитывать следующее:

– **Строковые типы данных** – все строковые значения должны быть заключены в одинарные кавычки;

– **Числовые типы данных** – все числовые значения должны предоставляться напрямую без их включения в одинарные или двойные кавычки;

– **Типы данных даты.** Нужно заключить значения даты в одинарные кавычки в формате `ГГГГ-ММ-ДД`.







Предположим, что у нас есть следующий список новых музыкальных инструментов, которые необходимо добавить в базу данных (табл. 5.3.1).

Таблица INSTRUMENTS					
id	name	type	data	num_string	photo
1	Домбра	Щипковый	25.12.2018	2	
2	Кобыз	Смычковый	25.12.2017	2	
3	Жетыген	Щипковый	15.04.2018	7	





id	name	type	data	num_string	photo
4	Сыбызгы	Духовой	07.10.2018	0	
5	Дабыл	Ударный	03.11.2018	0	
6	Шертер	Щипковый	11.07.2018	2	
7	Сазсырнай	Духовой	22.09.2017	0	

Команда **INSERT** позволяет вводить данные одни за другими. Начнем с домбры. Мы будем обрабатывать количество струн как числовой тип данных. Число можно не заключать в одинарные кавычки.

```
INSERT INTO instruments VALUES (1, 'Домбыра', 'Щипковый',
'2018-12-25', 2, 'dombra.jpg');
INSERT INTO instruments VALUES (2, 'Кобыз', 'Смычковый',
'2017-12-25', 2, 'kobyz.jpg');
```

### ПОЭКСПЕРИМЕНТИРУЙТЕ

1. Используя образец, заполните все остальные записи таблицы **instruments**.
2. Используйте инструкцию **SELECT** для просмотра всех строк в таблице **instruments** **SELECT \* FROM instruments**.



1. Какие команды используются для создания базы данных?
2. Какая команда используется для создания таблицы?
3. Какие типы данных поддерживаются в MySQL?
4. Зачем каждому полю присваивают свой тип данных?
5. Какая команда используется для добавления новых данных в таблицу?



1. Создайте базу данных **Полезные ископаемые Казахстана**, которая включает следующие данные: **Виды полезных ископаемых** (например, горючие, рудные, нерудные и т.д.), **Название**, **Месторождение**, **Область**. Введите 10 записей.

2. Изучите структуру таблиц.

3. Определите типы данных.

№	Данные	Типы данных
1	Верблюд	
2	3	
3	3000 тг	
4		
5	25.05.2018	
6	www.egov.kz	
7	Первые профессиональные художники Казахстана (А. Исмаилов, К. Ходжиков, Х. Ходжиков) проявили свой талант в 1920–1930-х годах. Эпоха революционных изменений, становление нового мира, история народа – главные темы их творений. В развитии живописи Казахстана в этот период особое значение играет Абылхан Кастеев (1904–1973). Именно он заложил основы национальной живописи	
8	Да/Нет	



1. Охотник на птиц решил хранить сведения о своей добыче в базе данных. Он хочет сохранить следующую информацию о каждой охоте: **дату**, **место**, **погоду**, **вес** пойманной птицы. Помогите охотнику грамотно построить таблицу базы данных.

Так же он решил запоминать, **сколько и каких птиц** поймал на каждой охоте. Как нужно изменить базу данных?

2. Подготовьте презентацию «Типы данных, хранящиеся в БД».

## ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ

## Установка Open Server (на примере ОС: Windows 7)

1. Скачать **Open Server** с официального сайта <https://ospanel.io>. Далее на странице **Скачать** выберите версию **Ultimate**, введите капчу и нажмите на кнопку скачивания (рис. 5.3.4).

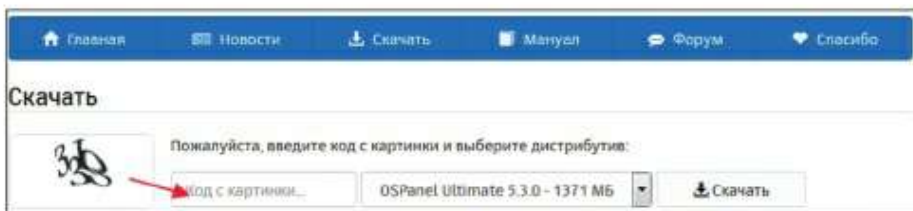


Рис. 5.3.4. Страница **Скачать** официального сайта **Open Server**

2. Щелкните по скачанному архиву. Запустится инсталлятор. На следующем шаге введите путь для распаковки **D:\OpenServer**.

3. После завершения процесса установки можно перейти в заданную папку и запустить программу. В зависимости от разрядности операционной системы выберите либо **Open Server x64.exe** (для 64-разрядной ОС), либо **Open Server x86.exe** (для 32-разрядной ОС).

4. Выберите язык интерфейса (здесь выбран **Русский**). Если программа запускается впервые, вам могут предложить установить патчи для Microsoft Visual C++. Для базовой работы с программой это можно не делать.

5. После завершения установки щелкните по ярлыку **Open Server** на панели задач и нажмите **Запустить** (рис. 5.3.5).

6. Откройте в любом браузере страницу <http://localhost>.

7. Если выйдет страница, как на рис. 5.3.6, то установка прошла правильно.

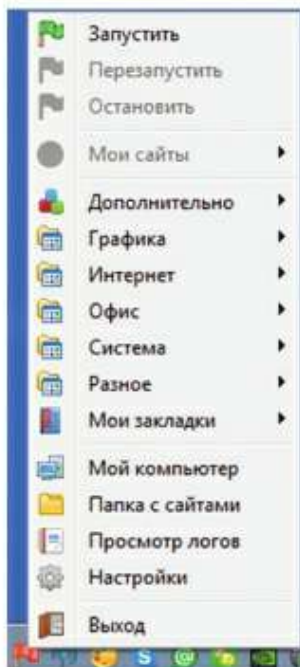


Рис. 5.3.5. Запуск **Open Server**

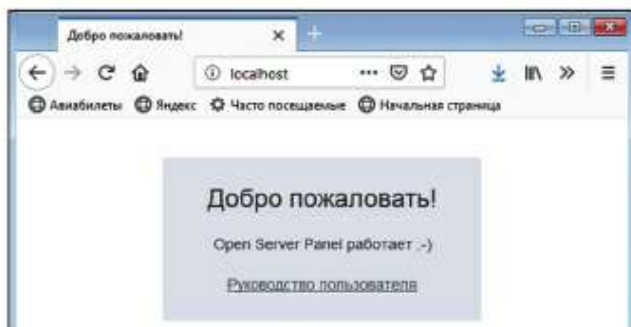


Рис. 5.3.6. Начальная страница Open Server

8. Для работы с базой данных нужно настроить MySQL. Для выбора необходимой версии MySQL в **Open Server** после запуска программы нажмите на значок в панели задач  и выберите меню **Настройки**.

9. В появившемся окне во вкладке **Модули** выберите версии Apache, PHP и MySQL, как показано на *рис. 5.3.7*. После сохранения на экране появится окно о перезагрузке **Open Server**, которая нужна для того, чтобы изменения вступили в силу.

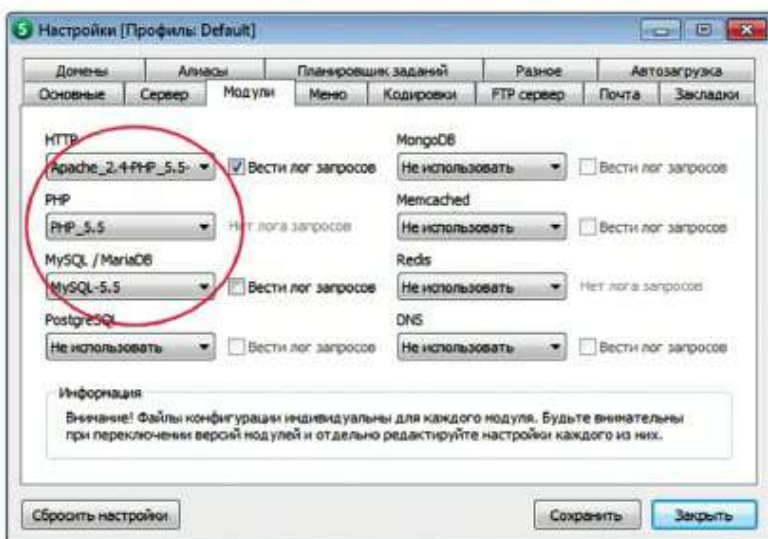


Рис. 5.3.7. Выбор версий MySQL



10. Далее выберите  – **Дополнительно** – **PhpMyAdmin**. В стартовом экране нужно ввести имя пользователя MySQL и его пароль для подключения к серверу (рис. 5.3.8). По умолчанию:

Пользователь: **root**

Пароль: **(пустой пароль)**.

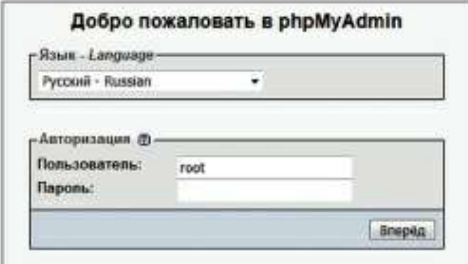


Рис. 5.8.5. Окно авторизации PhpMyAdmin

## 5.4. Разработка многотабличной базы данных

Почему не собрать всё в одной таблице?

Как создавать многотабличные базы данных.

**Связи** – Байланыстар – Relationships

**Целостность базы данных** – Мәліметтер қорының тұтастығы – Database Integrity

На сегодняшний день корпорация Oracle является абсолютным лидером на рынке систем управления базами данных. Она почти в два раза опережает такие крупные компании, как IBM и Microsoft. Ей принадлежит около 45% рынка.

вы узнаете

ключевые слова

это интересно

Мы рассмотрели простейшую структуру, в которой все данные сведены в одну таблицу, и поэтому искать информацию достаточно просто. Однако у такой модели есть и недостатки:

– дублирование данных: например, в базе данных **musical\_instruments** будет много раз храниться тип инструмента;

## 10 класс | информатика

- при изменении каких-то данных (например, **name\_instr** (название инструмента)), возможно, придется менять несколько записей;
- нет защиты от ошибок ввода (опечаток).

**Однотабличная база данных** – это аналог картотеки, в которой все карточки имеют одинаковую структуру. В то же время на практике в одной базе нужно хранить данные, относящиеся к объектам разных типов, которые связаны между собой. Поэтому возникает вопрос: «Какую модель лучше использовать для описания и хранения этих данных?»

Посмотрим, как можно организовать базу данных интернет-магазина по продаже казахских музыкальных инструментов. Дополним базу данных **musical\_instruments** следующими данными: **material** (материал изготовления), **producer** (изготовитель), **price** (цена), **sales** (скидки), **num\_instr** (количество инструментов в магазине) – и назовем новую базу данных **kaz\_mus\_intruments**. В однотабличной базе таблица **musical\_instruments** выглядит, как в таблице 5.4.1.

ТАБЛИЦА 5.4.1

## MUSICAL\_INSTRUMENTS

id_instr	name_instr	type	material	producer	num_instr	price	sales	data_begin	data_end	photo
1	Домбра	Щипковый	Сосна	Турдугулов Ж.	5	200 000				
2	Кобыз	Смычковый	Береза	Абрамкин В.	4	150 000				
3	Жетыген	Щипковый	Клен	ТОО «Лира»	6	180 000				
4	Сыбызгы	Духовой	Бамбук	Завод «Юпитер»	15	20 000	30%	01.03.18	31.03.18	
5	Дабыл	Ударный	Береза	Завод «Юпитер»	9	60 000				
6	Шертер	Щипковый	Орех	ТОО «Лира»	12	100 000	40%	15.03.18	30.03.18	
7	Саз-сырнай	Духовой	Глина	«Sapaly Saz»	35	15 000	20%	15.04.18	10.05.18	
8	Домбра	Щипковый	Бамбук	«Sapaly Saz»	21	150 000	28%			
9	Кобыз	Смычковый	Сосна	Абрамкин В.	3	110 000				
10	Шертер	Щипковый	Береза	«Sapaly Saz»	17	90 000				

Сразу видно, что в таблице есть дублирование – название изготовителя (символьная строка) повторяется для каждого инструмента этого изготовителя. Причина в том, что в данных таблицы на самом деле есть сведения не только об инструментах, но и об изготовителях – объектах совершенно другого класса. Поэтому для хранения всей информации об изготовителях нужно сделать отдельную таблицу (табл. 5.4.2).

ТАБЛИЦА 5.4.2

PRODUCERS

id_producers	name	address
1	Турдугулов Ж.	г. Алматы ул. Абая, 56
2	Абрамкин В.	г. Шымкент ул. Шакарима, 21
3	«Sapaly Saz»	г. Алматы ул. Рыскулова, 141
4	ТОО «Лира»	г. Павлодар ул. Баянауыл, 11
5	Завод «Юпитер»	г. Москва ул. Калужская, 15

Здесь первичным ключом будет **id\_producers**. В таблицу можно добавить и другие данные об изготовителе, например год создания фирмы, телефон и т.п. В таблице **instruments** теперь будут храниться не названия изготовителей, а их коды (табл. 5.4.3).

ТАБЛИЦА 5.4.3

MUSICAL\_INSTRUMENTS с кодами изготовителей

id_instr	name_instr	type	material	id_producers	num_instr	price	sales	data_begin	data_end	photo
1	Домбра	Щипковый	Сосна	1	5	200 000				
2	Кобыз	Смычковый	Береза	2	4	150 000				
3	Жетыген	Щипковый	Клен	4	6	180 000				
4	Сыбызгы	Духовой	Бамбук	5	15	20 000	30%	01.03.18	31.03.18	
5	Дабыл	Ударный	Береза	5	9	60 000				
6	Шертер	Щипковый	Орех	4	12	100 000	40%	15.03.18	30.03.18	
7	Саз-сырнай	Духовой	Глина	3	35	15 000	20%	15.04.18	10.05.18	
8	Домбра	Щипковый	Бамбук	3	21	150 000	28%			
9	Кобыз	Смычковый	Сосна	2	3	110 000				
10	Шертер	Щипковый	Береза	3	17	90 000				



## 10 класс | информатика

Таким образом, база данных состоит из двух таблиц (*табл. 5.4.2* и *табл. 5.4.3*) и хранит сведения о двух классах объектов. Эти таблицы нужно связать. Если между таблицами установлены связи, возможен контроль за данными, хранящимися в базе.

Реляционная база данных представляет собой совокупность связанных таблиц. Связи между таблицами устанавливаются через ключевые поля. После того как в БД созданы основные таблицы, следует указать, как они связаны друг с другом. Эти связи СУБД будет использовать в запросах, формах и отчетах при отборе информации из нескольких таблиц. Связь между двумя таблицами можно показать таким образом, как это показано на *рис. 5.4.1*.

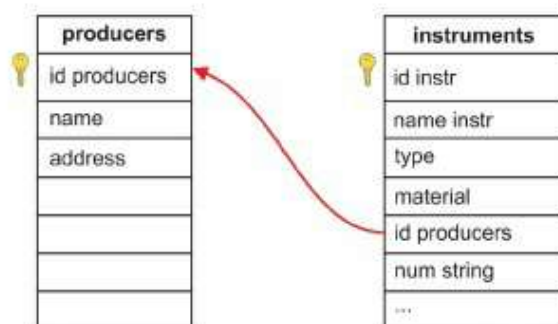


Рис. 5.4.1. Связь между таблицами

### ЗАПОМНИ

Одна из важнейших задач СУБД – поддерживать целостность базы данных, т.е. обеспечить правильность связей между таблицами.

Действия СУБД при связанных таблицах:

- запретить удаление записи до тех пор, пока в базе есть связанные с ней подчиненные записи;
- выполнить каскадное удаление, т.е. вместе с удаляемой записью удалить также все связанные с ней подчиненные записи в других таблицах;
- игнорировать проблему, т.е. разрешить внести изменения.



1. Создадим индексные таблицы для полей **name\_instr**, **type** и **material** таблицы 5.4.3.
2. Преобразуем табл. 5.4.3 в соответствии с индексными таблицами.
3. Определим типы полей для всех пяти таблиц.
4. Создадим базу данных **kaz\_mus\_instruments** в **phpMyAdmin**.

```
Create database kaz_mus_instruments;
```



## NAME\_INSTR

ТАБЛИЦА 5.4.4

id_name_ins	name_instr
1	Домбра
2	Кобыз
3	Жетыген
4	Сыбызгы
5	Дабыл
6	Шертер
7	Сазсырнай

## TYPE

ТАБЛИЦА 5.4.5

id_type	name_type
1	Щипковый
2	Смычковый
3	Духовой
4	Ударный

## MATERIAL

ТАБЛИЦА 5.4.6

id_material	name_material
1	Сосна
2	Береза
3	Клен
4	Бамбук
5	Орех
6	Глина

ТАБЛИЦА 5.6.7

Таблица INSTRUMENTS

id_instr	id_name_ins	id_type	id_material	id_producers	num_instr	price	sales	data_begin	data_end	photo
1	1	1	1	1	5	200 000				
2	2	2	2	2	4	150 000				
3	3	1	3	4	6	180 000				
4	4	3	4	5	15	20 000	30%	01.03.18	31.03.18	
5	5	4	2	5	9	60 000				
6	6	1	5	4	12	100 000	40%	15.03.18	30.03.18	
7	7	3	6	3	35	15 000	20%	15.04.18	10.05.18	
8	1	1	4	3	21	150 000	28%			
9	2	2	1	2	3	110 000				
10	6	1	2	3	17	90 000				



5. В БД `kaz_mus_instruments` создадим таблицы `instruments`, `producers`, `names`, `types`, `materials` в `phpMyAdmin`.

```
Create table name_instr (
  id_name_ins INT AUTO_INCREMENT PRIMARY KEY,
  name_instr varchar(20));
```

Здесь **PRIMARY KEY** – ключевое поле.

6. Введем все данные в таблицы в текстовом режиме. Можно использовать кнопки, расположенные внизу вкладки SQL.

**7. Введем команду INSERT:**

```
INSERT INTO `name_instr`
VALUES (NULL, 'домбра');
```



**1.** Почему собирать все данные в одной таблице во многих случаях нерационально?



**2.** По какому принципу данные разбиваются на несколько таблиц?

**3.** Какие таблицы связываются в многотабличных базах данных?

**4.** Что такое целостность базы данных? Как она обеспечивается?



**1.** Санжар решил хранить сведения о футбольных матчах своей любимой команды. Он хочет сохранять следующую информацию о каждом матче: **дату, место, с какой командой играли и счет**. Помогите Санжару построить многотабличную базу данных.

**2.** Арайлым хочет сохранить в базе данных информацию о своих поездках: **куда, когда, на чем ездил и срок пребывания**. Помогите Арайлым построить многотабличную базу данных.



**1.** Кондитерская фирма ведет базу данных заказчиков, состоящую из двух связанных таблиц.

Заказчики		
Код	ФИО	Код города
1	Баймаханова О.	2
2	Сейтова Г.	5
3	Мергенбаев Б.	2
4	Ким Г.	1
5	Воеводин А.	5

Города	
Код	Название
1	Алматы
2	Кызылорда
3	Павлодар
4	Тараз
5	Актау

Сколько заказчиков располагается в Кызылорде?

**2.** Во фрагменте базы данных представлены сведения о родственных отношениях. На основании приведенных данных определите фамилию и инициалы:



а) племянника Тен Н.О. (примечание: племянник – сын сестры или брата);

б) дедушки Ерғазы Ж.

Персоны			Дети	
Код	ФИО	Пол	Код члена семьи	Код ребенка
75	Нурғалиев А.Т.	м	94	69
69	Нурғалиев К.А.	м	75	12
48	Ерғазы Ж.	ж	81	48
81	Королев С.В.	ж	75	69
33	Королев А.С.	м	12	48
12	Ерғазы М.	м	81	33
94	Тен Р.Н.	ж	12	33
9	Тен Н.О.	м	94	12
...	...	...	75	9
...	...	...	...	...

## 5.5. SQL-запросы

Как производились сортировка и фильтрация в MS Excel?

- Как создавать запросы;
- как удалить и обновить записи в базе данных;
- как создавать сложные запросы.

**Запрос** – Сұраныс – Query  
**Выбрать** – Таңдау – Select  
**Удалить** – Жою – Delete  
**Обновить** – Жаңарту – Update  
**Объединять** – Біріктіру – JOIN

вы узнаете

ключевые слова

Понимание принципов работы SQL помогает создавать более точные запросы и упрощает исправление запросов, которые возвращают неправильные результаты.

### ЗАПОМНИ

#### ЗАПРОС (QUERY)

– это средство выбора необходимой информации из базы данных. Вопрос, сформулированный по отношению к базе данных, и есть запрос.

## 10 класс | информатика

Инструкция **SELECT** служит для описания набора данных на языке SQL. Она содержит полное описание набора данных, которые необходимо получить из базы данных, включая следующее:

- таблицы, в которых содержатся данные;
- связи между данными из разных источников;
- поля или вычисления, на основе которых отбираются данные;
- условия отбора, которым должны соответствовать данные, включаемые в результат запроса;
- необходимость и способ сортировки.

Инструкция SQL состоит из нескольких частей, называемых предложениями. Каждое предложение в инструкции SQL имеет свое назначение. Некоторые предложения являются обязательными. В таблице 5.5.1 указаны предложения SQL, используемые чаще всего.

Инструкция языка SQL		
Предложение SQL	Описание	Обязательное
SELECT	Определяет поля, которые содержат нужные данные	Да
FROM	Определяет таблицы, которые содержат поля, указанные в предложении <b>SELECT</b>	Да
WHERE	Определяет условия отбора полей, которым должны соответствовать все записи, включаемые в результаты	Нет
ORDER BY	Определяет порядок сортировки результатов	Нет

Пример синтаксиса **SELECT** показан на рис. 5.5.1.



Рис. 5.5.1. Пример синтаксиса **SELECT**



Это рекомендуемый синтаксис написания запроса, на самом деле ключевые слова можно вводить в любом регистре. В списке выбора указывается, из каких полей нужно выбрать данные. В списке таблиц определяется, в каких таблицах содержатся эти поля. В предложении **WHERE** указываются интересующие вас строки. Они могут содержать в себе как константы, так и выражения.

Давайте разберем запрос, чтобы понять:

- **SELECT** – какие поля выводим на экран;
- **FROM** – из каких таблиц берем данные;
- **WHERE** – показываем только те записи, которые удовлетворяют условию.

Рассмотрим использование оператора **SELECT** на примере таблицы **instruments**. Напишем простой SQL-запрос для вывода всех записей таблицы **instruments** (БД **musical\_instruments**):

```
SELECT * FROM Instruments;
```

Здесь \* означает все поля таблицы. После выполнения данной выборки получаем результат, как на *рис. 5.5.2*.

			id	name	type	data	num_string	photo
<input type="checkbox"/>			1	домбра	щипковый	2018-12-26	2	dombra.png
<input type="checkbox"/>			2	кобыз	смычковый	2018-10-25	2	kobyz.png
<input type="checkbox"/>			3	жетыген	щипковый	2018-04-15	7	jetygen.png
<input type="checkbox"/>			4	сыбызгы	духовой	2018-10-25	0	sybyzgy.png
<input type="checkbox"/>			5	дабыл	ударный	2018-11-03	0	dabyl.png
<input type="checkbox"/>			6	шертер	щипковый	2018-07-11	2	sherter.png
<input type="checkbox"/>			7	сазсырмай	духовой	2017-09-22	0	sazsyrmai.png

Рис. 5.5.2. Выборка данных из таблицы **instruments**



Из таблицы **instruments** нужно выбрать щипковые инструменты. Нужно выводить только их названия и дату изготовления, т.е. поля **name** и **data**. Затем этот же список выведите по возрастанию по полю **data**.

1. SQL-запрос будет следующим:

```
SELECT name, data FROM Instruments WHERE type="щип-
ковый";
```



2. Результат выборки показан на *рис. 5.5.3*.
3. Следующим условием запроса было вывести список по возрастанию по полю **data**. Предложение **ORDER BY** определяет сортировку, а значение **ASC** показывает вид сортировки по возрастанию (**DESC** – по убыванию):

```
SELECT name, data FROM Instruments
WHERE type="щипковый" ORDER BY data ASC;
```

4. Результат выборки показан на *рис. 5.5.4*.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Условия в **WHERE** могут быть сложными, если использовать логические операторы **OR**, **AND**. Попробуйте создать следующий SQL-запрос: из таблицы **instruments** выведите список домбры и жетыгена.

name	data
домбра	2018-12-25
жетыген	2018-04-15
шергер	2018-07-11

**Рис. 5.5.3.** Щипковые инструменты

name	data	▲ 1
жетыген	2018-04-15	
шергер	2018-07-11	
домбра	2018-12-25	

**Рис. 5.5.4.** Щипковые инструменты по возрастанию по полю **data**

## КОМАНДА DELETE

Команда SQL **DELETE** используется для удаления записей, которые больше не требуются из таблиц базы данных. Она удаляет всю строку из таблицы. Команда **DELETE** удобна для удаления временных или устаревших данных из базы данных. Также эта команда может удалять более одной строки из таблицы в одном запросе.

Как только строка была удалена, ее невозможно восстановить. Основной синтаксис команды **DELETE** показан ниже.

```
DELETE FROM table_name WHERE [Условие];
```

– **DELETE FROM table\_name** сообщает серверу MySQL удалять записи из таблицы.

– **WHERE [Условие]** является необязательным и используется для размещения фильтра, который ограничивает количество записей, затронутых запросом **DELETE**.

**ПРИМЕЧАНИЕ:** если предложение **WHERE** не используется в запросе **DELETE**, все строки в данной таблице будут удалены.



Из таблицы **instruments** в базе данных **musical\_instruments** удалим сазсырнай.

1. Вспомним названия полей этой таблицы. Для этого сделаем просмотр всех записей в таблице, выполнив команду:

```
SELECT * FROM instruments;
```

2. SQL-запрос для удаления:

```
DELETE FROM instruments WHERE name = "сазсырнай";
```

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Напишите одну команду **DELETE**, которая удалит лишние записи, не затрагивая другие записи из таблицы **instruments**. Сделайте просмотр всей таблицы, затем добавьте одну запись.

## КОМАНДА UPDATE

Команда **Update** используется для изменения строк в таблице, а также используется для одновременного обновления одного поля или нескольких полей, для обновления таблицы со значениями из другой таблицы.

Основной синтаксис команды SQL **Update** показан ниже.

```
UPDATE `table_name` SET `column_name` = `new_value` WHERE  
`[условие];`
```

– **UPDATE table\_name** – это команда, которая сообщает MySQL об обновлении данных в таблице.

– **SET column\_name = `new\_value`** – это имена и значения полей, на которые влияет запрос на обновление. Обратите внимание, что при настройке значений обновления типы данных строк должны быть в одинарных кавычках. Числовые значения необязательно должны быть в кавычках. Тип данных даты должен быть в одинарных кавычках и в формате **`ГГГГ-ММ-ДД`**.

– **WHERE [Условие]** является необязательным и может использоваться для размещения фильтра, ограничивающего количество записей, на которые влияет запрос **UPDATE**.





Обновим данные в таблице **instruments** (БД **musical\_instruments**). Изменим дату изготовления сыбызгы с 07.10.2018 на 25.10.2018.

**1.** Для начала узнаем **id** сыбызгы, потому что SQL-команды легче писать с **id**, чем с текстовыми полями.

```
SELECT * FROM instruments WHERE name = "сыбызгы";
```

Выполнение приведенного выше сценария дает нам следующий результат, **id = 4**:

id	name	type	data	num_string	photo
4	сыбызгы	духовой	2018-10-25	0	sybyzgy.png

**2.** Для обновления даты используйте скрипт, показанный ниже:

```
UPDATE instruments SET data = "2018-10-25" WHERE id = 4;
```

**3.** Просмотрим еще раз.

```
SELECT * FROM instruments WHERE id = 4;
```

Выполнение приведенного выше сценария дает нам следующие результаты:

id	name	type	data	num_string	photo
4	сыбызгы	духовой	2018-10-25	0	sybyzgy.png

### ПОЭКСПЕРИМЕНТИРУЙТЕ

В таблице **instruments** обновите данные других инструментов.

### КОМАНДА LIKE

Оператор **SQL LIKE** устанавливает соответствие символьной строки с шаблоном.

Оператор **SQL LIKE** имеет следующий синтаксис:

```
expression LIKE pattern
```

где, **expression** – любое символьное выражение, такое как столбец или поле; **pattern** – шаблон, по которому будет происходить проверка выражения **expression**. Шаблон может включать в себя следующие специальные символы (табл. 5.5.2).



## Специальные символы

ТАБЛИЦА 5.5.2

Символ	Описание
%	Строка любой длины
_	Любой одиночный символ
[]	Диапазон или последовательность символов
[^]	Исключающий диапазон или последовательность символов



В таблице **instruments** (БД **musical\_instruments**):

1. С помощью оператора **LIKE** выведем записи инструментов, имеющих в своем названии слово «щипковый»:

```
SELECT * FROM instruments WHERE type LIKE 'щип%'
```

В этом примере в качестве шаблона оператора **LIKE** послужил **'щип%'**. Исходя из условия задачи, после **щип** может быть любая длина символов.

2. С помощью оператора **LIKE** можно вывести записи инструментов, если вы не помните точно, как ввели название **жетыген** или **жетиген**, используя символ **\_** (нижняя черточка):

```
SELECT * FROM instruments WHERE name LIKE 'жет_ген'
```

## ЗАПРОСЫ В ЗАПРОСЕ

Для многотабличной базы данных характерны сложные запросы, т.е. запросы в запросе (вложенные запросы). Например, в базе данных **kaz\_mus\_instruments** нужно получить информацию о стоимости домбры. В данной базе названия инструментов и цены на них находятся в разных таблицах, т.е. выборка данных будет из двух таблиц: **instruments**, **name\_instr**. В таких случаях используется функция **IN()**. Синтаксис запроса будет следующим:

```
SELECT * FROM table1 WHERE id IN (SELECT id FROM table2);
```

Теперь по этому синтаксису напишем SQL-команду для нашего примера:

## 10 класс | информатика

```
SELECT id_name_ins, price
FROM instruments
WHERE id_name_ins
IN (SELECT id_name_ins FROM name_instr WHERE name_instr=
"домбра");
```

Во многих случаях можно переписать запрос, чтобы не использовать вложенный запрос. Например, синтаксис предыдущего запроса можно переписать следующим образом:

```
SELECT table1.* FROM table1,table2
WHERE table1.id=table2.id;
```

Соответственно SQL-запрос будет следующим:

```
SELECT instruments.id_name_ins, instruments.price
FROM instruments, name_instr
WHERE instruments.id_name_ins=name_instr.id_name_ins
AND name_instr="домбра";
```

В обоих случаях результат запроса будет:

id_name_ins	price
1	200000
1	150000

Как мы знаем, что под первым номером – домбра, а чтобы результат выдал название инструмента, нужна SQL-команда **JOIN**.

### SQL-КОМАНДА JOIN

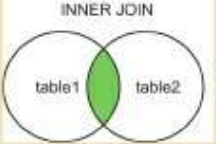

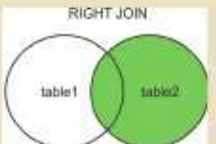
**JOIN** значит «объединять». В базе данных MySQL можно объединять при выборке поля из разных таблиц. Объединения позволяют извлекать данные из нескольких таблиц без создания временных таблиц и за один запрос.

В зависимости от требований к результату MySQL позволяет производить три разных типа объединения:

- 1) INNER JOIN (CROSS JOIN)** – внутреннее (перекрестное) объединение;
- 2) LEFT JOIN** – левостороннее внешнее объединение;
- 3) RIGHT JOIN** – правостороннее внешнее объединение.

Синтаксис SQL-команды JOIN (табл. 5.5.3).

ТАБЛИЦА 5.5.3

<pre>SELECT column_name(s) FROM table1 INNER JOIN table2 ON table1.column_name = table2. column_name;</pre>	 <p>INNER JOIN</p>
<pre>SELECT column_name(s) FROM table1 LEFT JOIN table2 ON table1.column_name = table2. column_name;</pre>	 <p>LEFT JOIN</p>
<pre>SELECT column_name(s) FROM table1 RIGHT JOIN table2 ON table1.column_name = table2. column_name;</pre>	 <p>RIGHT JOIN</p>

Теперь, используя синтаксис **JOIN**, перепишем предыдущий запрос, чтобы выходило название (домбра), а не код инструмента. Чтобы выбрать один из трех видов **JOIN**, сначала нужно определиться, какая таблица будет **table1**, а какая – **table2**. Пусть будет **table1** – **name\_instr**, **table2** – **instruments**. Так как поле **price** находится в **table2**, а нужно, чтобы это поле выходило с правой стороны, то выберем синтаксис **RIGHT JOIN**:

```
SELECT name_instr.name_instr, instruments.price
FROM name_instr
RIGHT JOIN instruments
ON instruments.id_name_ins=name_instr.id_name_ins
WHERE name_instr="домбра";
```

Результат запроса:

name_instr	price
домбра	200000
домбра	150000

## ПОЭКСПЕРИМЕНТИРУЙТЕ

Попробуйте написать предыдущий запрос, используя **LEFT JOIN**.



1. Что такое запрос? Для чего используются запросы?
2. Назовите основные предложения SQL. Опишите их.
3. Как вводить SQL-запрос в phpMyAdmin?
4. Что произойдет, если условие **WHERE** не задано?
5. Как вы думаете, может ли команда **DELETE** удалить только один столбец из записи?
6. Условие **WHERE** в команде **DELETE** чем-нибудь отличается от **WHERE** в команде **SELECT**?
7. Может ли команда содержать более одной секции **SET**?
8. Для чего используется команда **SQL LIKE**?
9. В каких случаях используются запросы в запросе?
10. Назовите виды **JOIN**. Опишите их.



1. Как вы думаете, каков будет результат выполнения следующих SQL-запросов?
  - 1) **SELECT** surname FROM student WHERE student\_code=15;
  - 2) **SELECT** Команда FROM Футбол WHERE Победы>7;
2. Дана таблица **Характеристики процессора** (табл. 5.5.4). Какова будет правильная последовательность записей после сортировки по убыванию по полю **Процессор**? Напишите SQL-запрос.

ТАБЛИЦА 5.5.4

## Характеристики процессора

№	Процессор	Жесткий диск	ОЗУ
1	Intel core i7	1 Тб	8 Гб
2	Intel core i5	200 Гб	4 Гб
3	Intel Dual-core	120	2 Гб
4	AMD Athlon II	320 Гб	4 Гб

3. Составьте базу данных **Напитки**. Для всех напитков, стоимостью 250 тенге, поднять цену до 350, а для напитков с ценой 350 – до 450. Удалите из базы данных напитки, у которых истек срок годности.



4. Имеется таблица **Universities**.

ТАБЛИЦА 5.5.5

ID	University Name	Students	Faculties	Professores	Location	Site
1	Al farabi Kazakh National University	12400	12	1229	Almaty	kaznu.kz
2	Abai Kazakh National Pedagogical University	21300	24	13126	Almaty	kaznpu.kz
3	Novosibirsk State University	7200	13	1527	Novosibirsk	nsu.ru
4	Moscow State University	35100	39	14358	Moscow	msu.ru
5	Higher School of Economics	20335	12	1615	Moscow	hse.ru
6	Nazarbaev University	5000	19	5640	Astana	nu.edu.kz
7	National Research Nuclear University	8600	10	936	Moscow	mephi.ru

С помощью оператора SQL LIKE выведите записи университетов:

- 1) имеющих в своем названии слово **National**;
- 2) доменное имя сайта которых содержит пять символов (за исключением домена **.kz**);
- 3) первая буква доменного имени сайта которых содержит буквы из диапазона [k-o];
- 4) вторая буква названия города которых не входит в диапазон [e-o].
5. Откройте базу данных **kaz\_mus\_instruments**. Постройте следующие запросы:
  - 1) выбрать ударные инструменты со скидкой;
  - 2) выбрать все инструменты с фотографиями;
  - 3) выбрать музыкальные инструменты, цена которых меньше 100 000; инструменты должны быть расставлены по убыванию по цене;
  - 4) вывести список инструментов с их изготовителями и адресами.



**1.** Создайте базу данных учащихся, которая включает следующие личные данные: **ФИО, Дата рождения, Гороскоп, Пол, Интересы, Адрес, Телефон.** Введите в таблицу 15 записей (данные должны быть разнообразными).

Используя SQL-запросы, найдите:

- лица женского пола (Ж) возраста более 16 лет;
- лица с определенным именем (например, Мухит);
- лица, родившиеся после определенной даты (22.04.2005);
- лица, по гороскопу относящиеся к Рыбам и родившиеся в 2005 г. (с 23.02.2005 до 22.03.2005);
- лица с определенным хобби (например, танцы), которым от 10 до 15 лет.

**2.** Спортсмены хотят сохранить в базе данных информацию о спортивных состязаниях и о том, где проходили спортивные турниры, кто из них какое место занял и в каком году. Помогите им грамотно построить базу данных. С каждым турниром нужно обновлять базу данных. Создайте 3 запроса с помощью команд JOIN, LIKE.

## 5.6. Установление связи web-страницы с базой данных

Где хранилась информация при создании web-страницы?

Как установить связь web-страницы с базой данных.

**Связь web-страницы с БД** –  
Web-парақшаның ДК-мен байланысы –  
Connect web page to data base

**вы узнаете**

**ключевые слова**

Темы следующих уроков – создание форм и отчетов. Мы создадим их как web-страницы. Так как данные хранятся в базе данных, то сначала нужно установить связь web-страницы с базой данных. Для этого необходимо знать, кроме HTML, один из языков web-программирования. Рассмотрим PHP-примеры.

**266**

**PHP-сценарии** – это простые текстовые файлы. Значит, при их создании можно пользоваться любыми известными средствами работы с текстовыми файлами, например **Notepad++**. Но программа, написанная на PHP, не запустится без локального сервера. В нашем случае локальным сервером является компьютер, а web-сервером – **Apache**.

У вас на компьютере установлен **Open Server (Apache + PHP + MySQL)**, значит, можно работать со скриптовым языком PHP для создания web-страницы с базой данных.

Чтобы связать **PHP** и **MySQL**, т.е. получить данные из базы данных **MySQL** в **PHP**-скрипт, нужно знать:

- 1) хост – адрес сервера баз данных **MySQL** (localhost);
- 2) имя базы данных;
- 3) имя пользователя;
- 4) пароль для доступа к БД;
- 5) SQL-команды.

## ЗАПОМНИ

### WEB-СТРАНИЦЫ С БАЗОЙ ДАННЫХ

– это диалоговая web-страница, которая поддерживает динамическую связь с базой данных и позволяет просматривать, редактировать и вводить данные в базу.



Создадим домен в пакете **Open Server**.

1. Корень сервера находится в папке **D: OpenServer/domains/localhost**.

2. В корневой папке создадим папку с новым именем домена (например, **test2.kz**).

3. Перезапустим **Open Server** (или запустим, если он не был запущен).

4. Создадим PHP-скрипт в **Notepad++** (рис. 5.6.1) и сохраним файл под названием **main.php** в папке **D:/OpenServer/domains/localhost/test2.kz/main.php** на локальном сервере.

Следующий код из файла **main.php** подключается к базе данных с именем **kaz\_mus\_instruments** и выполняет выборку изготовителей инструментов с адресами из таблицы **producers**. Код PHP сохраняет все данные, загруженные из базы, в массиве.

```

1 <html>
2 <head>
3   <title>!DOCTYPE</title>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5 </head>
6 <body>
7 <? $conn = mysql_connect("localhost","root","");
8
9 if (!$conn)
10 {
11     die('Did not connect: ' . mysql_error());
12 }
13 mysql_select_db("kaz_mus_instruments", $conn);
14 mysql_query("SET NAMES utf8", $conn);
15
16 $result = mysql_query("SELECT * FROM producers");
17 while($row = mysql_fetch_array($result))
18 {
19     echo $row['id_producers'] . " " . $row['name'] . " " . $row['address'];
20     echo "<br />";
21 }
22 mysql_close($conn);
23 ?>
24 </body>
25 </html>

```

Рис. 5.6.1. PHP-код для установления связи web-страницы с базой данных

## ПОДРОБНЕЕ О КАЖДОЙ СТРОКЕ

Всегда нужно отделять HTML-теги от PHP-кода символами `<? ... ?>`. Все переменные в PHP начинаются с символа `$`.

1) `<?>`

Первая строка сообщает web-серверу о том, что дальше следует код PHP.

2) `$conn = mysql_connect( "localhost", "root", "" )`

Чтобы подключиться к базе данных, необходимо сообщить web-серверу ее местонахождение, имя пользователя и пароль (при установке **Open Server** по умолчанию имя пользователя – **root**, а пароль не ставится). По этим данным создается строка подключения, которой присваивается имя

268



**\$conn.** Функция PHP **mysql\_connect()** получает эту информацию и пытается использовать ее для подключения к РСУБД.

```
3) if ( !$conn)
    {
    die ( 'Did not connect: ' . mysql_error());
    }
```

Если подключение не состоялось, PHP отправляет сообщение с описанием причины, а обработка кода PHP на этом прекращается.

```
4) mysql_select_db("kaz_mus_instruments", $conn);
```

```
5) mysql_query("SET NAMES utf8", $conn);
```

Эта строка используется, чтобы отправить данные на сервер с символами, которые не могут быть представлены в чистом ASCII.

Подключение к РСУБД прошло успешно. Теперь мы должны указать PHP, какая база данных нас интересует.

```
6) $result = mysql_query("SELECT * FROM producers");
```

База данных выбрана, подключение создано, но еще нет запроса для выполнения. Создаем такой запрос и отправляем его РСУБД функцией **mysql\_query()**. Все возвращаемые записи сохраняются в массиве **\$result**.

```
7) while($row = mysql_fetch_array($result))
    {
```

Теперь синтаксис PHP используется для вывода содержимого **\$result** на web-странице. Цикл **while** перебирает записи, пока не будет достигнут конец данных.

```
8) echo $row['id_producers'] . " ". $row['name'] . " ".
    $row['address'];
    echo "<br />";
    }
```

## 10 класс | информатика

Две команды PHP **echo** выводят на web-странице поля **id\_producers**, **name** и **address** каждой записи. Выводимые строки разделяются тегами HTML **<br>**.

9) `mysql_close($conn);`

После того как все данные будут выведены, подключение к РСУБД закрывается (по аналогии с завершением сеанса работы за терминалом).

?>

Сценарий PHP на этом завершен.

Файл **main.php** готов, теперь открываем его через браузер **localhost/test2.kz/main.php** (рис. 5.6.2). Если страница будет выглядеть, как показано на указанном рисунке, то на вашем компьютере правильно установлены все службы **Open Server**.

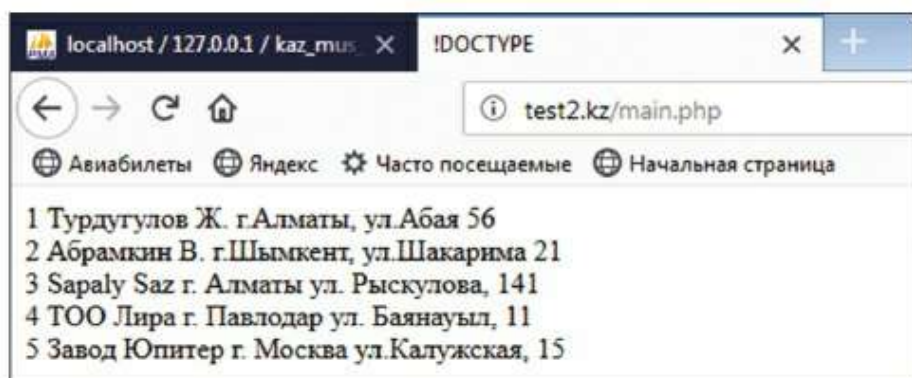


Рис. 5.6.2. Результат файла **main.php**



1. Почему удобно использовать **Open Server** при создании web-проектов?
2. Назовите команды для подключения к базе данных.
3. Где сохраняются PHP-файлы?
4. Как отделяются HTML-теги от PHP-кода?
5. Как пишутся переменные в PHP?

270



Установите соответствие. Какому фрагменту кода А–С соответствует результат 1–3?

A)

```

6 <body>
7 <? $conn = mysql_connect("localhost","root","");
8
9 if (!$conn)
10 {
11     die('Did not connect: ' . mysql_error());
12 }
13 mysql_select_db("kaz_mus_instruments", $conn);
14
15 $result = mysql_query("SELECT * FROM producers");
16 while($row = mysql_fetch_array($result))
17 {
18     echo $row['id_producers'] . " " . $row['name'] . " " . $row['address'];
19     /* echo "<br />"; */
20 }
21 mysql_close($conn);
22 ?>
23 </body>

```

1)



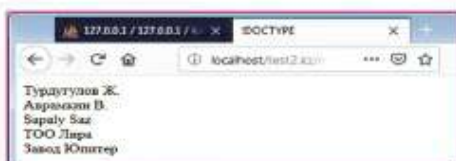
B)

```

6 <body>
7 <? $conn = mysql_connect("localhost","root","");
8
9 if (!$conn)
10 {
11     die('Did not connect: ' . mysql_error());
12 }
13 mysql_select_db("kaz_mus_instr", $conn);
14
15 $result = mysql_query("SELECT * FROM producers");
16 while($row = mysql_fetch_array($result))
17 {
18     echo $row['id_producers'] . " " . $row['name'] . " " . $row['address'];
19     echo "<br />";
20 }
21 mysql_close($conn);
22 ?>
23 </body>

```

2)



C)

```

6 <body>
7 <? $conn = mysql_connect("localhost","root","");
8
9 if (!$conn)
10 {
11     die('did not connect: ' . mysql_error());
12 }
13 mysql_select_db("kaz_mus_instruments", $conn);
14
15 $result = mysql_query("SELECT name FROM producers");
16 while($row = mysql_fetch_array($result))
17 {
18     echo $row['id_producers'] . " " . $row['name'] . " " . $row['address'];
19     echo "<br />";
20 }
21 mysql_close($conn);
22 ?>
23 </body>

```

3)



Сделайте изменения в файле **main.php**, чтобы выводился список инструментов, т.е. данные из таблицы **instruments**.

# 5.7. Формы

Почему для базы данных недостаточно таблицы?

- Как создавать формы;
- как связать формы с базой данных.

**Форма** – Форма – Form

вы узнаете

ключевые слова

## ЗАПОМНИ

### ФОРМА

– это средство, упрощающее ввод, редактирование и отображение информации.

Форма представляет собой набор функциональных элементов, которые используются для отображения информации, ввода и изменения данных, выполнения определенных действий. Примерами функциональных элементов являются кнопки, текстовые поля, списки и т.д.

Нас интересуют HTML-формы, так как вы уже знаете HTML и умеете связать web-страницу с базой данных.

Весь код формы заключается в элемент **<form>**. Элемент **<input>** – это самый важный элемент формы. Элемент **<input>** может быть показан несколькими способами, это зависит от атрибута **type**. В таблице 5.7.1 показаны некоторые примеры.

ТАБЛИЦА 5.7.1

Типы/Описание	Пример кода	Результат в браузере
<code>&lt;input type="text"&gt;</code> Определяет однострочное текстовое поле ввода	<pre> &lt;form&gt; Логин:&lt;br&gt; &lt;input type="text" name="login"&gt;&lt;br&gt; Пароль:&lt;br&gt; &lt;input type="text" name="password"&gt; &lt;/form&gt; </pre>	Логин: <hr/> Пароль: <hr/>



Типы/Описание	Пример кода	Результат в браузере
<input type="radio"> Определяет переключатель (для выбора одного из вариантов)	<pre>&lt;form&gt; &lt;input type="radio" name="gender" value="male" checked&gt; Мужчина&lt;br&gt; &lt;input type="radio" name="gender" value="female"&gt; Женщина&lt;br&gt; &lt;/form&gt;</pre>	
<input type="submit"> Определяет кнопку <b>Отправить</b> (для отправки формы)	<pre>&lt;form method="post" action="action_page.php"&gt; &lt;input type="submit" value="OK"&gt; &lt;/form&gt;</pre>	
<select> ... </select> Создает выпадающий список	<pre>&lt;select name="country"&gt; &lt;option&gt;Выберите страну проживания&lt;/option&gt; &lt;option value="Казахстан"&gt; Казахстан&lt;/option&gt; &lt;option value="Россия"&gt;Россия&lt;/option&gt; &lt;option value="Беларусь"&gt; Беларусь&lt;/option&gt; &lt;/select&gt;</pre>	

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Имя

Пол  мужской  
 женский

E-mail

Страна

Попробуйте создать HTML-форму и CSS, используя **Notepad ++** как на *рис. 5.7.1*.

**Рис. 5.7.1.** Пример HTML-формы



1. Создадим форму, как на *рис. 5.7.2*, которая добавляет новые инструменты в базу данных **kaz\_mus\_instruments** в таблицу **name\_instr**.

### Музыкальные инструменты

Название инструмента:

Рис. 5.7.2. Форма Музыкальные инструменты

2. Назовем файл создания формы **form.php** (*рис. 5.7.3*). Сохраним в папке **test2.kz**.

```

1  $name_instr = $_POST['name_instr'];
2  /* проверка поля $name_instr на пустоту */
3  if ($name_instr) {
4      /* если поле $name_instr не пустое, то соединение к БД */
5      $conn = mysql_connect('localhost', 'root', '');
6      if (!$conn)
7          die('Did not connect: ' . mysql_error());
8      mysql_select_db('kaz_mus_instruments', $conn);
9      $SQL = "INSERT INTO name_instr VALUES (NULL, '$name_instr')";
10     mysql_query($SQL);
11     /* если поле $name_instr пустое, то выводим следующее сообщение */
12     } else { echo "Пожалуйста введите значение!"; }
13 }
14
15 <html>
16 <head>
17 <title>ДОСТУП</title>
18 <meta http-equiv="Content-Type" content="form.php" charset=utf-8" />
19 </head>
20 <body>
21 <h1>Музыкальные инструменты</h1>
22 <br>
23 <form method="post" action="">
24 <input type="text" name="name_instr" value="" />
25 <input type="submit" value="Save" />
26 </form>
27
28 $conn = mysql_connect('localhost', 'root', '');
29 if (!$conn)
30     die('Did not connect: ' . mysql_error());
31 mysql_select_db('kaz_mus_instruments', $conn);
32 $SQL="SELECT * FROM name_instr";
33 $result = mysql_query($SQL);
34 /* выводим значения полей таблицы name_instr записанные в массив переменных $row */
35 while($row = mysql_fetch_array($result))
36 {
37     /* вывод на экран */
38     echo $row['id_name_instr'] . " " . $row['name_instr'];
39     echo "<br />";
40 }
41 mysql_close($conn);
42 }
43 </body>
44 </html>

```

Рис. 5.7.3. PHP-код создания формы Музыкальные инструменты



3. Результат **form.php** показан на *рис. 5.7.4.*

## Музыкальные инструменты

Название инструмента:

- 1 домбра
- 2 кобыз
- 3 жетыген
- 4 сыбызгы
- 5 дабыл
- 6 шертер
- 7 сазсырнай

**Рис. 5.7.4.** Форма Музыкальные инструменты

Подробнее опишем метод **POST**. Мы создали HTML-страницу с простой формой. Запомните, что метод **POST** может использоваться только в форме.

Первый параметр формы – **method**, он определяет метод, который мы будем использовать для передачи. Все названия переменных и значения будут передаваться как запрос браузера к web-серверу.

Второй параметр формы – **action**. Это путь и имя файла скрипта, которому мы передаем данные. В нашем случае это **form.php**. Если не указать значение параметра **action**, то вся информация будет передаваться главному скрипту, то есть индексной странице **form.php** сайта.

Обязательно нужно проверять на пустоту и допустимые значения. Ниже приведен фрагмент кода, где проверяется значение поля: если переменная **\$name\_ins** не пустая, то значение поля записывается в таблицу базы данных, иначе выведет сообщение «**Нельзя вводить пустые значения**».

```
if ($name_ins) {
    $conn = mysql_connect("localhost", "root", "");
```

```

if (!$conn)
{
    die('Did not connect: ' . mysql_error());
}
mysql_select_db("kaz_mus_instruments", $conn);
$$SQL="INSERT INTO name_instr VALUES (NULL,'$name_ins')";
mysql_query($$SQL);
} else { echo "Нельзя вводить пустые значения"; }

```

Далее нужно уточнить, почему переменная называется именно **name\_ins**. В 31-й строке кода `<input type="text" name="name_ins" value="">` параметр **name** и задает имя переменной, которую мы получим с помощью этого поля. Передача переменных между формами и страницами – это именно та интерактивность, ради которой используется PHP.



1. Что такое форма? Зачем используются формы?
2. Что служит источником данных для формы?
3. Какие функциональные элементы вы знаете? Опишите их.
4. Как создаются HTML-формы?
5. Опишите принцип работы метода **POST**.



Создайте базу данных **Исторические столицы Казахстана**. Создайте таблицу с полями: **Код города**, **Название города**, **Год основания города**, **В какие годы город был столицей**.

- 1) Создайте HTML-форму **Столицы** (рис. 5.7.5).
- 2) Введите через данную форму пять записей в таблицу базы данных.



Столицы			
Код	<input type="text" value="1"/>	Год основания	<input type="text" value="1820"/>
Название города	<input type="text" value="Кызылорда"/>	В какие годы	<input type="text" value="1925-1927"/>

Рис. 5.7.5. Форма **Столицы**



Подготовьте презентацию на тему «Как лучше хранить изображения в базе данных».





## 5.8.

## Отчеты

Что такое отчеты?

Как создавать отчеты.

Отчеты – Есентеп – Reports

вы узнаете

ключевые слова

**Отчеты** позволяют извлечь из базы нужные сведения и представить их в виде, удобном для восприятия, а также предоставляют широкие возможности для обобщения и анализа данных.

Предположим, что нужно создать отчет в соответствии с макетом, как на *рис. 5.8.1* (используйте базу данных `kaz_mus_instruments`).

**ЗАПОМНИ****ОТЧЕТ**

– это форматированное представление данных, которое выводится на экран, в печать или в файл.

Музыкальные инструменты по изготовителям (Текущая дата)			
Изготовитель _____			
Номер	Название инструмента	Цена	Скидка
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____
			На сумму: _____

**Рис. 5.8.1.** Макет отчета **Музыкальные инструменты по изготовителям**

Для начала напишем PHP-код web-страницы вывода отчета без оформления. Используем сложные запросы с помощью SQL-команды **JOIN**

## 10 класс | информатика

(рис. 5.8.2). Обратите внимание на синтаксис сложного запроса, как нужно написать на несколько строк (вспомните SQL-команду **JOIN** из урока 5.5).

```

$$SQL1= "SELECT instruments.price, instruments.sales,
name_instr.name_instr FROM instruments";
$$SQL1 .= "LEFT JOIN name_instr ON instruments.id_name_
ins=name_instr.id_name_ins";
$$SQL1 .= "WHERE instruments.id_producers=$row[id_
producers]";

```

```

1 <html>
2 <head>
3 <title>!DOCTYPE</title>
4 <meta http-equiv="Content-Type" content="reports.php; charset=utf-8">
5 </head>
6 <body>
7
8 <H3>Музыкальные инструменты по изготовителям</H3>
9 <br>
10 <? $conn = mysql_connect("localhost", "root", "");
11 if (!$conn)
12 {
13     die("Did not connect: " . mysql_error());
14 }
15 mysql_select_db("kaz_mus_instruments", $conn);
16 $$SQL="SELECT id_producers, name FROM producers";
17 $result = mysql_query($$SQL);
18 while($row = mysql_fetch_array($result))
19 {
20     /* выводим переменные: $i - для вывода нумерации $sum - для подсчета общей суммы */
21     $i++; $sum=0;
22     echo "Изготовитель:"; echo $row['name'];
23     echo "<br/>";
24     /* так как название инструмента и цена находится в разных таблицах,
25     чтобы вывести на экран по запросу отчета, используем LEFT JOIN, т.е.
26     нужно объединить поля из разных таблиц */
27
28     $$SQL1= "SELECT instruments.price, instruments.sales, name_instr.name_instr FROM instruments";
29     $$SQL1 .= "LEFT JOIN name_instr ON instruments.id_name_ins=name_instr.id_name_ins";
30     $$SQL1 .= "WHERE instruments.id_producers=$row[id_producers]";
31
32     $res = mysql_query($$SQL1);
33     while($r = mysql_fetch_array($res))
34     {
35         echo $i=$i+1; echo " ".$r[name_instr]." ". $r[price]." ". $r[sales];
36         echo "<br/>";
37         $sum=$sum+$r[price];
38     }
39     echo "<br/>";
40     echo "Итого суммарно:"; echo " ", $sum;
41     echo "<br/>"; echo "<br/>";
42 }
43 mysql_close($conn);
44 ?>
45 </body>
46 </html>

```

**Рис. 5.8.2.** Фрагмент кода отчета  
**Музыкальные инструменты по изготовителям**

Результат данного кода показан на рис. 5.8.3.

**278**

### Музыкальные инструменты по изготовителям

Изготовитель: Турдугулов Ж.  
1 домбра 200 000  
На сумму: 200 000

Изготовитель: Абрамкин В.  
1 кобыз 150 000  
2 кобыз 110 000  
На сумму: 260 000

Изготовитель: «Sapaly Saz»  
1 сазсырнай 15000 20%  
2 домбра 150 000 28%  
3 шертер 90 000  
На сумму: 255 000

Изготовитель: ТОО «Ли́ра»  
1 жетыген 180 000  
2 шертер 100 000 40%  
На сумму: 280 000

Изготовитель: Завод «Юпитер»  
1 сыбызгы 20 000 30%  
2 дабыл 60 000  
На сумму: 80 000

**Рис. 5.8.3.** Результат отчета  
**Музыкальные инструменты  
по изготовителям**

Теперь нужно, используя HTML-теги, оформить отчет. Сначала добавим таблицу, чтобы отчет выглядел, как на *рис. 5.8.4.*

Изготовитель: Турдугулов Ж.			
№	Название инструмента	Цена	Скидка
1	домбра	20 000	
На сумму: 200 000			
Изготовитель: Абрамкин В.			
№	Название инструмента	Цена	Скидка
1	кобыз	150 000	
2	кобыз	110 000	
На сумму: 260 000			

**Рис. 5.8.4.** Результат на браузере **reports.php** после добавления таблицы

Фрагмент кода после добавления таблицы показан на *рис. 5.8.5*.

```
<? echo "Изготовитель: "; echo $row['name'];?> </p>
<table>
<tr>
<td>?</td><td>Название инструмента</td><td>Цена</td><td>Скидка</td>
</tr>

<? $sSQL= "SELECT instruments,price, instruments.sales, name_instr,name_instr FROM instruments";
$sSQL .= " LEFT JOIN name_instr ON instruments.id =name_instr.id name_instr";
$sSQL .= " WHERE instruments.id_producers=$row[id_producers]";

$res = mysql_query($sSQL);
while($r = mysql_fetch_array($res))
{
?> <td><? echo $r[id];?></td><td> <? echo $r[name_instr]; ?></td><td> <? echo $r[price]; ?></td>
<td> <? echo $r[sales]; ?></td>
</tr>
<? $sum=$sum+$r[price];
} ?>
<tr>
|
<td></td><td></td><td><td><? echo "Всего: "; ?></td><td><? echo " ". $sum;?></td>
</tr>
</table>
<? } ?>
```

**Рис. 5.8.5.** Фрагмент кода **reports.php** после добавления таблицы.

### ПОЭКСПЕРИМЕНТИРУЙТЕ

Попробуйте далее оформить web-страницу (отчет **Музыкальные инструменты по изготовителям**).



1. Что такое отчет?
2. Каково основное назначение отчетов?



1. Создайте отчет вывода на печать списка инструментов по типам. Оформите его.
2. Создайте отчет вывода на печать списка инструментов, у которых есть скидки на введенную пользователем дату. Оформите его.



- Подготовьте реферат на тему «Как еще можно создавать отчеты из базы данных?».



# 5.9. Big Data

Что такое Big Data?

Как оценивать положительные и отрицательные стороны использования Big Data.

**Большие данные** –  
Үлкен деректер –  
Big Data

Термин «большие данные» ввел редактор журнала Nature Клиффорд Линч еще в 2008 году в спецвыпуске, посвященном взрывному росту мировых объемов информации.

вы узнаете

ключевые слова

это интересно

## ЗАПОМНИ

### BIG DATA

– это различные инструменты, подходы и методы обработки как структурированных, так и неструктурированных данных для того, чтобы их использовать для конкретных задач и целей.

## ЗАПОМНИ

### НЕСТРУКТУРИРОВАННЫЕ ДАННЫЕ

– это информация, которая не имеет заранее определенной структуры или не организована в определенном порядке.

Организации создают огромные объемы данных, но большая их часть представлена в формате, который плохо соответствует традиционному структурированному формату БД (web-журналы, видеозаписи, текстовые документы, машинный код, геопространственные данные и т.д.). Всё это хранится во множестве разнообразных хранилищ, иногда даже за пределами организации. В результате корпорации могут иметь доступ к огромному объему своих данных и не иметь необходимых инструментов, чтобы установить взаимосвязи между этими данными и сделать на их основе значимые выводы. Традиционные методы анализа информации не могут угнаться за огромными объемами постоянно обновляемых данных. Это в итоге привело к технологиям **больших данных** (табл. 5.9.1).

Большие данные относятся к большому количеству информации, которая передается по различным каналам – обычно онлайн – каждую секунду. Эти данные слишком велики, сложны и динамичны, чтобы для них можно

ТАБЛИЦА 5.9.1

## Традиционная база данных и база больших данных

Характеристика	Традиционная база данных	База больших данных
Объем информации	От гигабайт до терабайт	От петабайт до эксабайт
Способ хранения	Централизованный	Децентрализованный
Структурированность данных	Структурирована	Полуструктурирована или неструктурирована
Модель хранения и обработки данных	Вертикальная модель	Горизонтальная модель
Взаимосвязь данных	Сильная	Слабая

было использовать традиционные инструменты. Благодаря технологическому прогрессу эти данные теперь можно интерпретировать и анализировать, что дает большие преимущества для здравоохранения, правительства, розничного производства, электронного обучения и других отраслей, управляемых данными.

Обычно большие данные поступают из трех источников:

- 1) Интернет (соцсети, форумы, блоги, СМИ и другие сайты);
- 2) корпоративные архивы документов;
- 3) показания датчиков, приборов, камер и других устройств.

Следует отметить, что накопленные объемы данных не всегда можно отнести к числу больших. Существует пять признаков, характеризующих сферу больших данных (рис. 5.9.1).

Уже все крупные банки и лидеры телеком-рынка используют технологии Big Data. То же касается медиа-, интернет-компаний, госсектора. Удивительно, что лидирующие позиции в части использования Big Data занимают промышленные предприятия и добывающие секторы, например металлургия и нефтедобывающий сектор. Если телекоммуникационная индустрия и банковский сектор работают с данными, которые генерируют люди, то реальный сектор получает сведения с датчиков, установленных на оборудовании. Данных с таких датчиков можно собрать в тысячи раз больше, чем данных о людях.



Рис. 5.9.1. Признаки больших данных

Перегруженность особенно коснулась интернет-компаний. Google обрабатывает более петабайта данных в день. В Facebook загружают более 10 миллионов новых фотографий ежечасно. Люди нажимают кнопку **Нравится** или пишут комментарии почти три миллиарда раз в день, оставляя за собой цифровой след, с помощью которого компания изучает предпочтения пользователей. 800 миллионов ежемесячных пользователей службы YouTube компании Google каждую секунду загружают видео длительностью более часа.

Классический пример использования Big Data – торговая площадка Amazon или китайская ее версия AliExpress. Анализируя просмотры, покупки, интересы, программа отслеживает закономерность и рекомендует то, что потенциально может заинтересовать пользователя. Уровень продаж благодаря такому инструменту значительно возрастает.

«Сбербанк» внедрил систему, которая анализирует фотографии для идентификации клиентов банка и предотвращает мошенничество. Подобная система была внедрена еще в 2014 году. В ее основе – сравнение фотографий из базы, которые попадают туда с web-камер на стойках благодаря



«компьютерному зрению». Основа системы – биометрическая платформа. Преимущества анализа больших данных значительны, но существуют потенциальные недостатки (табл. 5.9.2).

### ПРЕИМУЩЕСТВА И НЕДОСТАТКИ БОЛЬШИХ ДАННЫХ

ТАБЛИЦА 5.9.2

Преимущества	Недостатки
<p><b>Эффективность принятия решений.</b> Аналитика может дать лицам, принимающим бизнес-решения, информацию, которая им необходима, чтобы помочь их компаниям конкурировать и развиваться</p>	<p><b>Потребность в специалистах ИТ.</b> Наем или обучение специалиста в области ИТ может значительно увеличить расходы, а процесс приобретения навыков работы с большими данными может занять значительное время</p>
<p><b>Повышение производительности.</b> Современные инструменты для работы с большими данными позволяют аналитикам быстрее анализировать больше данных, что повышает их личную производительность</p>	<p><b>Качество данных.</b> Прежде чем использовать большие данные, надо убедиться, что используемая информация является точной, актуальной и в надлежащем формате для анализа</p>
<p><b>Улучшение обслуживания клиентов.</b> Социальные сети и другие системы управления взаимоотношениями с клиентами предоставляют предприятиям обширную информацию о своих клиентах. Эти данные удобно использовать для лучшего обслуживания клиентов</p>	<p><b>Конфиденциальность.</b> Большая часть информации, хранящейся в хранилищах больших данных компаний, является конфиденциальной или личной, и это означает, что фирме может потребоваться обеспечить соответствие отраслевым стандартам или государственным требованиям при обработке и хранении данных</p>



Преимущества	Недостатки
<p><b>Обнаружение мошенничества.</b> Системы анализа больших данных способны распознавать закономерности и аномалии. Например, банкам и компаниям, выпускающим кредитные карты, предоставляется возможность обнаружить украденные кредитные карты или мошеннические покупки</p>	<p><b>Риски кибербезопасности.</b> Хранение больших данных, особенно конфиденциальных, может сделать компанию более привлекательной целью для злоумышленников</p>
<p><b>Увеличение доходов</b> часто является результатом использования организацией больших данных</p>	<p><b>Быстрые изменения.</b> Эта технология быстро меняется. Организации сталкиваются с реальной возможностью того, что они будут инвестировать в определенную технологию, чтобы через несколько месяцев появилось что-то лучшее</p>
<p><b>Повышение гибкости.</b> Многие организации используют большие данные, чтобы лучше согласовать свои усилия в области ИТ и бизнеса, они используют аналитику для поддержки более быстрых и частных изменений своих бизнес-стратегий</p>	<p><b>Потребность в оборудовании.</b> Предприятия несут значительные расходы, связанные с оборудованием, техническим обслуживанием и сопутствующими услугами (пространство для хранения данных, пропускная способность сети, вычислительные ресурсы и т.д.)</p>
<p><b>Большие инновации.</b> Многие руководители вкладывают средства в аналитику, прежде всего, как средство для инноваций. Если появляются идеи, которых нет у их конкурентов, они смогут выйти на рынок с новыми продуктами и услугами</p>	<p><b>Сложность интеграции систем.</b> Большинство предприятий, которые существовали в течение очень многих лет, собирали данные в различных приложениях и системах в своих средах. Интеграция всех этих разрозненных источников данных увеличивает время и затраты на работу с большими данными</p>



1. Что такое Big Data?

2. Объясните причину появления технологии Big Data.

3. Какие данные можно отнести к большим данным?

4. Перечислите признаки, характеризующие большие данные.



---

Составьте диаграмму Венна на тему «Плюсы и минусы Big Data».

---



Выберите из списка одно из направлений:

- банковская сфера;
- операторы связи;
- интернет-компании;
- образование;
- сфера услуг;
- торговля.

Выполните исследовательскую работу и подготовьте сообщение на тему «Как можно использовать Big Data для улучшения работы компаний?».

## ТЕСТОВЫЕ ЗАДАНИЯ

## «ИНФОРМАЦИОННЫЕ СИСТЕМЫ»



Знание



Понимание



Применение



Анализ



Синтез

**1. База данных – это:**

- А) таблица, позволяющая хранить и обрабатывать данные и формулы;
- В) организованная совокупность данных;
- С) файл специального формата, содержащий информацию, которая имеет заданную структуру;
- Д) прикладная программа для обработки информации пользователя;
- Е) программные средства, осуществляющие поиск информации.

**2. Есть база данных, которая описывает интересы учеников. Перечислите возможные имена полей.**

- А) ФИО.
- В) Теннис.
- С) Возраст.
- Д) Хобби.
- Е) Футбол.
- Ф) Плавание.

**3. Укажите поля, которые могут быть использованы в качестве ключевого.**

- А) Фамилия.
- В) Индивидуальный код школьника.
- С) Марка компьютера.
- Д) Порядковый номер сотрудника.
- Е) Дата рождения.
- Ф) ИИН сотрудника.

**4. В таблице представлен фрагмент базы данных о погоде. Количество записей в данном фрагменте, удовлетворяющих условию:  $\text{Осадки} = \text{«дождь»}$  и  $\text{Температура воздуха} < 10$ , равно:**

287



Дата	Температура воздуха, °С	Влажность воздуха, %	Осадки
18.10.17	+12	91	дождь
19.10.17	+13	78	нет
20.10.17	+8	62	нет
21.10.17	+5	90	дождь
22.10.17	+9	91	нет
23.10.17	+10	75	дождь
24.10.17	+13	61	дождь
25.10.17	+8	91	нет
26.10.17	+15	66	нет

А) 1; В) 2; С) 3; D) 4; E) 5.



**5. Определите соответствие типа данных 1–5 для каждого поля таблицы А–Е.**

1	Текстовый	А)	ФИО учащегося
2	Числовой	В)	Дата рождения
3	OLE	С)	Занятия в секции (Да/Нет)
4	Логический	D)	Достижения (в виде файла)
5	Дата/Время	E)	Возраст



**6. База данных содержит сведения о товарах: *Название, Группа товаров, Изготовитель, Дата продажи*. Как выглядит запрос для вывода списка кондитерских изделий фабрики «Рахат» с 01.12.2018 г. включительно?**

А) Группа товаров = «Кондитерские изделия» или Изготовитель= «Рахат» и Дата продажи >01.12.2018.





**B)** Группа товаров = «Кондитерские изделия» и Изготовитель= «Рахат» и Дата продажи <01.12.2018.

**C)** Группа товаров = «Кондитерские изделия» или Изготовитель= «Рахат» или Дата продажи >=01.12.2018.

**D)** Группа товаров = «Кондитерские изделия» и Изготовитель= «Рахат» и Дата продажи >=01.12.2018.



**7. SQL-команда для добавления новой записи в базу данных:**

**A)** CREATE;

**B)** JOIN;

**C)** INSERT;

**D)** UPDATE;

**E)** DELETE.



**8. Напишите синтаксис создания таблицы под названием table\_name в базе данных.**



**9. Что будет выведено в результате запроса (из БД musical\_instruments)?**

```
SELECT Count(*)  
From instruments  
WHERE (type="щипковый" AND num_string=2)
```



**10. Напишите запрос, который бы выбирал инструменты, названия которых начинаются с буквы «д», в алфавитном порядке.**

**ТЕМЫ ДЛЯ ПРОЕКТНЫХ РАБОТ**

1. «Семейный бюджет».
2. «Библиотека».
3. «Фитнес-центр».
4. «Кадровое агентство».
5. «Пиццерия».
6. «Мастерская по ремонту мобильных телефонов».
7. «Телефонный справочник организации».
8. «База данных автомобилей».
9. «Интернет-магазин».
10. «Курсы иностранных языков».
11. «Туристическое агентство».
12. «Салон красоты».

**Цель:** разработка и наполнение базы данных (БД).

**Задачи:**

1. Проанализировать существующие БД.
2. Разработать структуру БД.
3. Наполнить данными БД.
4. Создать запрос, форму и отчет.

**СТАДИИ РАБОТЫ НАД ПРОЕКТОМ****1. Проектирование базы данных:**

- определение назначения БД;
- определение структуры таблиц;
- определение полей;
- определение первичного ключа;
- определение связей между таблицами.

**2. Создание базы данных:**

- создание однотабличной базы данных;
- преобразование в многотабличную базу данных;
- наполнение БД нужными данными по исследуемой теме.

**3. Работа с запросами:**

- с помощью двух SQL-запросов изменить структуру двух таблиц (например, добавить или удалить какое-либо поле);
- создать SQL-запросы, которые будут использоваться для манипулирования данными;
- сформулировать два подзапроса (однотабличный и многотабличный), создать их с помощью SQL.

**4. Создание форм для ввода данных.** Для созданной базы данных разработайте необходимые формы для ввода данных при помощи инструментов SQL.

**5. Создание отчета.** Для своей базы данных при помощи инструментов SQL разработайте необходимые отчеты для вывода данных.

**Этапы работы над проектом****ПОДГОТОВИТЕЛЬНЫЙ**

1. Определение тематического поля и темы проекта.
2. Поиск и анализ проблемы.
3. Постановка цели проекта.

**АНАЛИТИЧЕСКИЙ**

1. Анализ имеющейся информации.
2. Поиск информационных лагун.
3. Сбор и изучение информации.
4. Поиск оптимального способа достижения цели проекта (анализ альтернативных решений), построение алгоритма деятельности.
5. Составление плана реализации проекта: пошаговое планирование работ.
6. Составление плана.

**ПРАКТИЧЕСКИЙ**

1. Выполнение запланированных технологических операций.
2. Текущий контроль качества.
3. Внесение (при необходимости) изменений в конструкцию и технологию.

**ПРЕЗЕНТАЦИОННЫЙ**

1. Подготовка презентационных материалов.
2. Презентация проекта.
3. Изучение возможностей использования результатов проекта (выставка, продажа, включение в «банк проектов», публикация и т.д.)

**КОНТРОЛЬНЫЙ**

1. Анализ результатов выполнения проекта.
2. Оценка качества выполнения проекта.

В проектной деятельности за раздел должны быть охвачены все пройденные темы за раздел.



## ГЛОССАРИЙ

**Big Data** – это различные инструменты, подходы и методы обработки как структурированных, так и неструктурированных данных для того, чтобы их использовать для конкретных задач и целей.

**CSS** (*Cascading Style Sheets*) – технология управления внешним видом элементов web-страницы.

**Create Database** – это SQL-команда для создания базы данных.

**IP-адрес** – это уникальный адрес в сети, необходимый для нахождения, передачи и получения информации от одного компьютера (узла) к другому.

**HTML** (*HyperText Markup Language* – язык разметки гипертекста) – стандартизированный язык разметки документов во Всемирной паутине. Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

**MAC-адрес (или аппаратный адрес)** – это цифровой код длиной 6 байт, устанавливаемый производителем сетевого адаптера и однозначно идентифицирующий данный адаптер.

**SQL** (*Structured Query Language* – структурированный язык запросов) – язык управления базами данных для реляционных баз данных.

**Web-страница с базой данных** – это диалоговая web-страница, которая поддерживает динамическую связь с базой данных и позволяет просматривать, редактировать и вводить данные в базу.

**Алфавит системы счисления** – это используемый набор цифр.

**Аргумент (или параметр)** – это значение, которое передается подпрограмме из основной программы при ее вызове.

**Арифметико-логическое устройство (АЛУ)** – основная часть процессора, предназначенная для выполнения арифметических и логических операций.

**Архитектура сети** – это реализованная структура сети передачи данных, определяющая ее топологию, состав устройств и правила их взаимодействия в сети.

**Атрибуты** – свойства, применяемые для предоставления дополнительной информации об элементе.

## ГЛОССАРИЙ

**Аутентификация** – процесс проверки подлинности или принадлежности пользователю предъявленного им идентификатора. Другими словами, аутентификация заключается в проверке: является ли подключающийся пользователь тем, за кого он себя выдает.

**База данных (БД)** – это специальным образом организованная совокупность данных о некоторой предметной области, хранящаяся во внешней памяти компьютера.

**Вызов подпрограммы** – обращение к ней по имени с целью ее использования.

**Граф** (от греч. *grapho* – пишу) – это абстрактное представление множества объектов и связей между ними. Граф называется деревом, если он связный и не имеет замкнутых путей – циклов.

**Декодирование** – преобразование данных из двоичного кода в форму, понятную человеку.

**Дизъюнкция** – логическая операция, ставящая в соответствие двум высказываниям новое, являющееся ложным тогда и только тогда, когда оба исходных высказывания ложны.

**Запрос** (*query*) – это средство выбора необходимой информации из базы данных. Вопрос, сформированный по отношению к базе данных, и есть запрос.

**Защита информации** – порядок и правила применения принципов и средств защиты информации.

**Идентификация** – это процесс предъявления идентификатора.

**Идентификатор** (*ID, data name, identifier – опознаватель*) – это уникальный признак объекта, позволяющий отличать его от других объектов.

**Индекс** – это вспомогательная таблица, которая служит для ускорения поиска в основной таблице.

**Информационная безопасность в сфере информатизации** – это состояние защищенности электронных информационных ресурсов, информационных систем и информационной инфраструктуры от внешних и внутренних угроз. *Закон «Об информатизации» (11 апреля 2019 г.)*

**Информационные системы** – в широком смысле – это аппаратные и программные средства, предназначенные для того, чтобы своевременно обеспечивать пользователей нужной информацией.

**Итерируемые объекты** – это объекты, которые предоставляют возможность поочередного прохода по своим элементам.

**Код** – это набор условных обозначений (или сигналов) для записи (или передачи) некоторых заранее определенных понятий.

**Кодек** (сокращение слов *coder/decoder* – шифратор/дешифратор) – это алгоритм сжатия данных в файле. Каждый тип контейнера имеет только определенные кодеки, которые он поддерживает.

**Кодирование информации** – это процесс формирования определенного представления информации, т.е. это переход от одной формы представления информации к другой, более удобной для хранения, передачи или обработки.

**Кодовая таблица (или таблица кодировки)** – это таблица, содержащая упорядоченный некоторым образом перечень кодируемых символов, в соответствии с которым происходит преобразование символа в его двоичный код и обратно.

**Колонтитул** (*фр. colonne – столбец и лат. titulus – надпись, заголовок*) – строка, расположенная на краю полосы набора и содержащая заголовки, имя автора, название произведения, части, главы, параграфа и т.д.

**Компьютерные вирусы** – разновидность вредоносных программ, которые могут создавать свои копии и внедрять их в файлы, системные области компьютера и компьютерных сетей и тем самым самостоятельно распространяться.

**Конкатенация** – это операция, при которой оператор '+' используется для объединения (сцепления) строк.

**Контент** – информационное содержание сайта (тексты, графическая, звуковая информация и др.), а также книги, газеты, сборника статей, материалов и др.

**Конъюнкция** – логическая операция, ставящая в соответствие двум высказываниям новое, являющееся истинным тогда и только тогда, когда оба исходных высказывания истинны.



## ГЛОССАРИЙ

**Логический элемент (вентиль)** – простейшая структурная единица компьютера, выполняющая определенную логическую операцию над двоичными переменными.

**Логические операции** – мыслительные действия, результатом которых является изменение содержания или объема понятий, а также образование новых понятий.

**Логическая схема** – это электронное устройство, которое реализует любую логическую функцию, описывающую работу устройств компьютера.

**Маршрутизация** – это процесс выбора маршрута следования пакета. Устройство, совершающее этот выбор, называется маршрутизатором.

**Мощность** – количество символов в алфавите.

**Неструктурированные данные** – это информация, которая не имеет заранее определенной структуры или не организована в определенном порядке.

**Октет** – восемь двоичных разрядов.

**Основание системы** – нижний индекс, используемый для различных систем счисления.

**Отрицание** – логическая операция, которая каждому высказыванию ставит в соответствие новое высказывание, значение которого противоположно исходному.

**Отчет** – это форматированное представление данных, которое выводится на экран, в печать или в файл.

**Пакет (кадр)** в телекоммуникации и компьютерных сетях – это фрагмент (блок данных) определенного формата для передачи данных по каналу связи.

**Первичный ключ, или ключ** – это одно или несколько полей, совокупность значений которых однозначно определяет каждую запись в таблице. Наличие ключа в таблице исключает возможность появления в ней двух одинаковых записей.



**Подпрограмма** – это именованная группа операторов, выполняющих некоторую задачу. Подпрограмму запускают, вызвав по имени необходимое количество раз из любых мест программы.

**Разряд** – позиция цифры в числе. Разряды в записи целых чисел нумеруются от нуля справа налево.

**Рекурсия** (от лат. *recursio* – возвращение) – это способ описания объектов или вычислительных процессов через самих себя.

**Рекурсивная функция** – это функция, которая вызывает сама себя непосредственно или с помощью других функций.

**Реляционная база данных** – это база данных, которую можно представить в виде набора таблиц.

**Сетевой протокол** – набор правил и действий (очередности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включенными в сеть устройствами.

**Символьная строка** – это последовательность символов, расположенных в памяти в соседних ячейках. Для работы с символами во многих языках программирования есть переменные специального типа: символы и символьные строки.

**Система управления базой данных (СУБД)** – это программное обеспечение (ПО), которое позволяет создавать БД, обновлять и дополнять информацию, обеспечивать гибкий доступ к информации.

**Скрипт** – это программа или программный файл, сценарий, автоматизирующие некоторую задачу, которую пользователь делал бы вручную, используя интерфейс программы.

**Список** – это взаимосвязанный набор отдельных фраз или предложений, который начинается с маркера или цифры.

**Стек** (англ. *stack* – кipa, стопка) – специально организованная область ОЗУ, используемая для временного сохранения данных или адресов.

**Тег** (от англ. *tag*) – команда HTML, указывающая, какую операцию необходимо выполнить, например вывести на экран текст, нарисовать линию или использовать графику.

## ГЛОССАРИЙ

**Форма** – это средство, упрощающее ввод, редактирование и отображение информации.

**Форматирование текста** – это выбор средств для его изменения, таких как редактирование шрифта или использование разнообразных эффектов.

**Функции** – название всех подпрограмм в Python. Они могут быть двух типов: возвращающими или не возвращающими значение.

**Шаг рекурсии (рекурсивный вызов)** – обращение функции к себе самой, но с другим значением параметра, таким, который приближает к базовому случаю.

**Шифрование данных (криптография)** – это преобразование информации в другую форму таким образом, что только пользователь, имеющий секретный ключ, называемый ключом дешифрования, может получить к нему доступ.

**Электронная цифровая подпись (ЭЦП)** – это личная подпись, представляющая собой набор электронных цифровых символов, подтверждающая достоверность электронного документа, его принадлежность и неизменность содержания.

## Список использованной литературы

1. Концепция кибербезопасности («Киберцит Казахстана»). Июнь, 2017 г.
2. Закон Республики Казахстан «Об информатизации» от 24 ноября 2015 года № 418-V ЗРК.
3. *Бейли Л.* Изучаем SQL. – СПб.: Питер, 2012. – 573 с.
4. *Босова Л. Л.* Информатика. 10 класс. Базовый уровень. – М.: 2017. – 288 с.
5. *Босова Л. Л.* Информатика. 11 класс. Базовый уровень. – М.: 2017. – 256 с.
6. *Дейт К. Дж.* SQL и реляционная теория. Как грамотно писать код на SQL. – СПб.: Символ-Плюс. – 2010. – 474 с.
7. *Олифер В., Олифер Н.* Компьютерные сети. Принципы, технологии, протоколы. 5-е издание. – СПб.: Питер, 2016. – 992 с.
8. *Поляков К. Ю.* Программирование. Python. C++. Часть 4. Учебное пособие. – М.: Бином. Лаборатория знаний, 2019. – 192 с.
9. *Поляков К. Ю., Еремина Е.А.* Информатика. 10 класс. Учебник. В 2-х частях. Базовый и углубленный уровни. ФГОС. – М.: Бином. Лаборатория знаний, 2019. – 704 с.
10. *Сергеев А. Н.* Основы локальных компьютерных сетей. Учебное пособие. – СПб.: Лань, 2016. – 184 с.

### Интернет-ресурсы

1. <http://netemul.sourceforge.net/>
2. <https://notepad-plus-plus.org/download/v7.5.6.html>
3. <https://ospanel.io>
4. <http://ilab.xmedtest.net/?q=node/4650>

## СОДЕРЖАНИЕ

### Раздел 1.

#### КОМПЬЮТЕРНЫЕ СЕТИ И ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

Введение .....	3
1.1. Сетевые устройства: узлы, коммутаторы .....	4
1.2. IP-адресация. Система доменного именования .....	10
Дополнительная информация к §§1–2 .....	17
1.3. Информационная безопасность .....	20
1.4. Методы защиты информации .....	27
1.5. Методы идентификации личности .....	31
Тестовые задания по разделу 1 .....	37

### Раздел 2.

#### ПРЕДСТАВЛЕНИЕ ДАННЫХ

2.1. Системы счисления .....	40
2.2. Перевод чисел из одной системы счисления в другую .....	45
2.3. Логические операции. Построение таблиц истинности .....	49
2.4. Логические элементы компьютера .....	56
2.5. Логические основы компьютера .....	61
2.6. Принципы кодирования текстовой информации .....	64
Тестовые задания по разделу 2 .....	70
Проектная деятельность за I четверть .....	73

### Раздел 3.

#### АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ

3.1. Пользовательские процедуры .....	75
3.2. Параметры .....	80
3.3. Пользовательские функции .....	87
3.4. Рекурсия .....	93



3.5. Символьные строки .....	99
3.6. Функции и методы строковой обработки.....	106
3.7. Файлы .....	113
3.8. Работа с файлами .....	120
3.9. Методы сортировки .....	126
3.10. Алгоритмы на графах .....	134
Тестовые задания по разделу 3 .....	142
Проектная деятельность за II четверть .....	146

## **Раздел 4.**

### **WEB-ПРОЕКТИРОВАНИЕ**

4.1. Введение в HTML.....	149
4.2. Дополнительные возможности форматирования текста в HTML .....	157
4.3. Вставка таблиц и создание списков в HTML.....	165
4.4. Вставка изображения на web-страницу .....	172
4.5. Вставка гиперссылки на web-страницу .....	178
4.6. Использование каскадной таблицы стилей (CSS) .....	184
4.7. CSS-селекторы и свойства .....	193
4.8. Разметка web-страницы.....	202
4.9. Использование скриптов на web-странице.....	209
4.10. Внедрение мультимедиа на web-страницу.....	216
Тестовые задания по разделу 4 .....	224
Проектная деятельность за III четверть .....	227

## **Раздел 5.**

### **ИНФОРМАЦИОННЫЕ СИСТЕМЫ**

5.1. Введение в базы данных.....	228
5.2. Основные понятия базы данных. Проектирование базы данных.....	233
5.3. Разработка однотабличной базы данных в MySQL .....	239

5.4. Разработка многотабличной базы данных .....	249
5.5. SQL-запросы .....	255
5.6. Установление связи web-страницы с базой данных.....	266
5.7. Формы.....	272
5.8. Отчеты.....	277
5.9. Big Data .....	281
Тестовые задания по разделу 5 .....	287
Проектная деятельность за IV четверть .....	290
Приложение.....	292
Глоссарий .....	293
Список использованной литературы.....	299

Учебное издание

**Исабаева Даража Нагашыбаевна  
Рахимжанова Ляззат Болтабаевна  
Киселёва Елена Анатольевна  
Курмангалиева Нургуль Айтбаевна  
Аубекова Мадина Аскаркызы**

## **ИНФОРМАТИКА**

Учебник для 10 класса естественно-математического направления  
общеобразовательной школы

Зав. редакцией *Н. Жиенгалиев*  
Редактор *Л. Холина*  
Художественный редактор *З. Аульбекова*  
Технический редактор *О. Рысалиева*  
Компьютерная верстка *Н. Развинавичене*

ИБ № 040

Сдано в набор 25.02.2019. Подписано в печать 23.07.2019. Формат 70 × 90 1/16.  
Бумага офсетная. Гарнитура «Segoe UI». Печать офсетная. Усл. печ. л. 22,23.  
Уч.-изд. л. 15,22. Тираж 7000 экз. Заказ № 4418.

ТОО «Корпорация «Атамұра», 050000, г. Алматы, пр. Абылай хана, 75.  
Полиграфкомбинат ТОО «Корпорация «Атамұра», Республика Казахстан, 050002,  
г. Алматы, ул. М. Макатаева, 41.

