Қазақстан Республикасы Білім және ғылым министрлігі ұсынған

Г.И.Салғараева
Ж.Б.Базаева
А.С.Маханова

ИНФОРМАТИКА

Жалпы білім беретін мектептің жаратылыстану-математика бағытының 10-сыныбына арналған оқулық

10

басы
арман-пв

ӘОЖ 373.167.1
КБЖ 32.973 я 72
С 18

Ғылыми кеңесшісі:
Ж.У.Кобдикова *– педагогика ғылымдарының докторы*

Салғараева Г.И., ж.б.
С 18 **Информатика:** Жалпы білім беретін мектептің жаратылыстану-математика бағытының 10-сыныбына арналған оқулық. / Г.И.Салғараева, Ж.Б.Базаева, А.С.Маханова – Нұр-Сұлтан: «Арман-ПВ» баспасы, 2019. – 240 бет.

ISBN 978-601-318-228-5

Оқулық жалпы орта білім беру деңгейінің жаңартылған мазмұндағы үлгілік оқу бағдарламасына сәйкес оқушылардың жас ерекшеліктері ескеріле отырып жазылды. Оқулық тілі жеңіл, түрлі мақсаттағы тапсырмалармен қамтылған.

ӘОЖ 373.167.1
КБЖ 32.973 я 72

ISBN 978-601-318-228-5

ШАРТТЫ БЕЛГІЛЕР

Жаңа тақырыпты меңгеру тапсырмалары – функционалдық сауаттылықты қалыптастыру тапсырмалары

1. Сұрақтарға жауап берейік
2. Ойланайық, талқылайық
3. Талдап, салыстырайық
4. Дәптерде орындайық
5. Компьютерде орындайық
6. Ой бөлісейік

Естеріңе түсіріңдер:

Өткен тақырыптардан бүгінгі сабаққа негіз болатын тапсырмалар

Меңгерілетін білім:

Тақырыптағы игерілетін мәліметтер; күтілетін нәтижелер

Қызықты ақпарат

Материалды жеңіл меңгеруге жетелейтін ақпараттар

Терминдер

Ғылыми ұғымдар

Назар аудар

Электронды қосымша жүктелген CD қолжетімсіз болған жағдайда, қосымшаны *arman-pv.kz* сайтынан тауып, өз компьютеріңе жүктеп алуыңа болады

Алғы сөз

Қымбатты жас шәкірттер! Жаңа оқу жылының басталуымен құттықтаймыз!

Қолдарыңдағы оқулық «Компьютерлік желілер және ақпараттық қауіпсіздік», «Деректерді ұсыну», «Алгоритмдеу және программалау», «Web-жобалау» және «Ақпараттық жүйелер» бөлімдерінен тұрады.

«Компьютерлік желілер және ақпараттық қауіпсіздік» бөлімінде желілік компоненттердің, IP-мекенжайдың, домендік атаулар жүйесінің, жеке виртуалды желінің мақсаттары түсіндіріледі.

«Деректерді ұсыну» бөліміне бір санау жүйесінен екінші санау жүйесіне сандарды аудару, логикалық операциялар (дизъюнкция, конъюнкция, инверсия), ақиқат кестесін құру, мәтіндік ақпараттарды кодтау принциптері кіріп отыр.

«Алгоритмдеу және программалау» бөлімі функциялармен процедуралар, жолдармен, сұрыптау алгоритмдерімен, графтардағы алгоритмдермен жұмыс жасау туралы деректерді қамтиды.

«Web-жобалау» бөлімі HTML-дің, CSS-тің негізгі тегтері, web-беттерді жасауда скриптерді қолдану және web-бетке мультимедиа нысандарын кірістіру үшін HTML тегтерін қолдану сияқты тақырыптардан тұрады.

«Ақпараттық жүйелер» бөлімі деректер қорының негізгі түсініктері мен деректер қорын құру, Big Data-ны пайдалануда оң және теріс әсерлерін бағалау, деректерді енгізуге арналған форма жасау (SQL) тәрізді өздерің үшін ең қызықты деректер мен тапсырмалар жүйесінен тұрады.

«Қызықты ақпарат» айдарындағы деректерді де оқып, зерделеріңе тоқи жүрсеңдер, білімдерің де, ой-өрістерің де күннен күнге арта түседі.

«Сұрақтарға жауап берейік», «Ойланайық, талқылайық», «Талдап, салыстырайық», «Дәптерде орындайық», «Компьютерде орындайық», «Ой бөлісейік» тапсырмалар тобын орындау барысында жаңа тақырыпты оңай меңгересіңдер. Глоссарий қандай да бір ұғымдар мен анықтамаларды есте сақтауға көмектеседі.

Оқулыққа қосымша электронды оқу құралы (CD диск) берілген. Дискіде берілген интерактивті тапсырмаларды орындап, сыныпта алған білімдеріңді үйде бекіте аласыңдар. Сендерге осы пәнді қызыға оқып, алған білімдеріңді практикалық тұрғыдан күнделікті өмірде табысты қолдана алуларыңа тілектеспіз!

1-БӨЛІМ

КОМПЬЮТЕРЛІК ЖЕЛІЛЕР ЖӘНЕ АҚПАРАТТЫҚ ҚАУІПСІЗДІК

Күтілетін нәтижелер:

- желілік компоненттердің мақсатын сипаттау (тораптар, маршрутизаторлар, коммутаторлар);
- IP-мекенжайының мақсатын және көрсетілуін түсіндіру;
- домендік атаулардың жүйесінің (DNS) мақсатын түсіндіру;
- жеке виртуалды желінің мақсатын түсіндіру;
- «ақпараттық қауіпсіздік», «құпиялылық», «тұтастық» және «қолжетімдік» терминдерінің маңызын түсіндіру;
- деректерді шифрлау қажеттілігін бағалау;
- пайдаланушы деректерін қорғау шараларын қолдануды түсіндіру: пароль, тіркеулік жазба, аутентификация, биометриялық аутентификация.

§ 1. Компьютерлік желілердің жұмыс жасау принциптері. Желілік компоненттер

Естеріңе түсіріңдер:

- *компьютерлік желі дегеніміз не?*
- *қандай желі түрлерімен таныссыңдар?*

Меңгерілетін білім:

- *«компьютерлік желі» түсінігі;*
- *компьютерлік желі түрлері;*
- *аппараттық компоненттер.*

Терминдер:

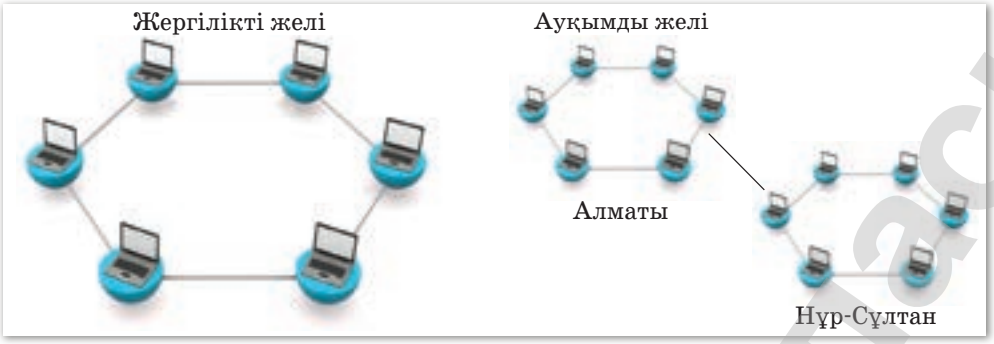
- маршрутизатор;
- коммутатор;
- концентратор.

Қазіргі уақытта Қазақстанда телекоммуникация саласының тез дамуы жаңа жоғары технологиялық қызметтер – деректерді тарату, ұялы байланыс және Интернет желісіне қатынауды ұсыну бойынша қызметтер сегментінің туындауына негізделген. Жалпы **желі** деп дербес компьютерлердің және принтер, модем және есептеу құрылғыларының бір-бірімен байланысқан жиынын айтады. Желілер әрбір қолданушыға өзге қолданушымен дерек алмасып, құрылғыларды ортақ пайдалануға, қашықта орналасқан компьютерлердегі деректер қорымен қатынас құруға және тұтынушылармен тұрақты байланыс жасауға мүмкіндік береді.

Компьютерлік желі – бір-бірімен дерек алмаса алатын кем дегенде екі компьютердің байланыс құралдары көмегімен қарым-қатынас жасауына арналған ақпарат өңдеудің тармақталған жүйесі. Компьютерлік желілер, оларды қамтитын аумаққа байланысты *жергілікті* (LAN – Local Area Network) және *ауқымды* (WAN – World Area Network) желі болып бөлінеді (*1-сурет*).

Егер бір бөлмеде, ғимаратта немесе жақын орналасқан ғимараттар кешенінде пайдаланушылар қандай да бір мәселені бірігіп шешуге, деректер алмасуға немесе ортақ деректерді пайдалануға тиісті бірнеше компьютер бар болса, онда осы компьютерлерді **жергілікті желіге** біріктіру орынды болады.

Ауқымды желі – жүздеген және мыңдаған километрлік аумақты қамтитын халықаралық, мемлекетаралық, республикалық немесе салалық компьютер желілері. Кез келген компьютерді олардың орналасқан географиялық мекенжайына қарамастан, бір-бірімен байланыстыруға мүмкіндік береді.



1-сурет. *Жергілікті және ауқымды желі*

Желідегі компьютер екі қызмет атқара алуы мүмкін:

1. Егер компьютер ақпарат алу үшін және сервиспен басқа желідегі компьютермен байланысса, онда бұл компьютер **жұмыс станциясы** деп аталады.
2. Егер компьютер басқа желідегі компьютерге ақпарат және сервис берсе, онда ол **сервер** деп аталады.

Компьютерлік желі негізгі аппараттық және программалық компоненттерден құрастырылып, компоненттердің өзара келісімді әрекеттесуі арқылы қызмет етеді (*2-сурет*).

2-сурет. *Желідегі желілік компоненттер және жұмыс станциялары*

Негізгі аппараттық компоненттер:

- абоненттік жүйелер – компьютерлер (жұмыс станциялары немесе серверлер), принтерлер, сканерлер, т.с.с.

- желілік компоненттер – концентратор (хаб), коммутатор және маршрутизатор.
- байланыс арналары – сымдар, қосқыш ұяшықтар, сымсыз технологиялар арқылы деректер жіберу және қабылдау құрылғылары.

Негізгі желілік программалық компоненттер:

- желілік операциялық жүйелер – Windows NT, Windows NT Server, Windows for Workgroups, LANtastic, NetWare, Unix, Linux және т.б.
- желілік программалық жабдықтар – желі клиенті, желілік адаптерлер, хаттамалар, қашықтықтан ену қызметі.

Компьютерлік желідегі желілік компоненттер желінің негізгі құрауыштары болып табылады. Олардың әрқайсысы маңызды және желілік компьютерлер арасындағы байланысты оңтайландыруда әртүрлі қызмет атқарады. Сырт келбеті бойынша бұл құрылғылар ұқсас болып келеді (*3-сурет*).

3-сурет. *Концентратор, коммутатор, маршрутизатор*

Концентратор – коммутатор мен маршрутизатормен салыстырғанда желідегі ең қарапайым және арзан құрылғы. Барлық деректер концентратордың бір портына келіп түсіп, қалған порттарға таратылады. Осылайша, бір концентраторға қосылған барлық компьютер желіде бір-бірін көріп отырады.

Коммутатордың (switch немесе қосқыш) қызметі концентратор қызметіне ұқсас. Ерекшелігі – желіде таратылатын әр дерек пакетінде адресат пен дереккөздің MAC-мекенжайлары болуында. Коммутатор, порттарына жалғанған әр компьютер мекенжайын «еске сақтап алып», реттеуші қызмет атқарып – деректерді тек адресат компьютеріне ғана жібереді. Бұл желінің

өнімділігін арттырады, себебі қажет емес пакеттер жоқ және желінің өткізгіштік қабілеті ұлғаяды.

Маршрутизатор (router) – деректер пакетін тарату үшін екі не одан да көп желілерді байланыстыратын «ақылды» құрылғы. Маршрутизатор коммутатормен салыстырғанда күрделі әрі қымбат желілік құрылғы болып табылады. Ол желілік трафикті бақылау, желідегі өзгерістерді жылдам анықтау және қажетті пакеттерді өткізіп, қажетсіз пакеттерді блоктау қызметтерін атқарады.

Коммутатор мен маршрутизатор айырмашылығын көрсететін мысал – корпорацияның пошталық сервері. Қызметкер хатты жібергенде, хат адресатқа компанияның поштаны ішкі жеткізу жүйесі немесе жергілікті пошта бөлімшесі (егер хат алушы компаниядан тыс болса) арқылы жеткізілуі мүмкін. Мұнда коммутатор – компанияның ішкі сервері, ал маршрутизатор жергілікті пошта бөлімшесі ретінде көрінеді.

1 Сұрақтарға жауап берейік

1. Компьютерлік желі дегенді қалай түсінесіңдер?
2. Компьютерлік желінің қандай түрлері бар?
3. Желілік компоненттер қандай жағдайда қолданылады?
4. Компьютерлік желіні құру үшін қандай желілік компоненттер қажет?
5. Компьютерлік желіні құру барысында нені үйренесіңдер?

2 Ойланайық, талқылайық

1. Компьютерлік желі не үшін қажет?
2. Не себепті ауқымды желінің мүмкіндігі жоғары?
3. Компьютерлік желілерді белгілері бойынша бөлудің принциптерін анықтаңдар.

3 Талдап, салыстырайық

1. Желілік компоненттердің бір-бірінен қандай ұқсастығы және айырмашылығы бар?
2. Негізгі аппараттық компоненттер қызметтерін салыстырып, ұқсастықтарын анықтаңдар.
3. Жергілікті желінің ауқымды желіден айырмашылығы қандай? Қайсысының мүмкіндігі жоғары?

4 Дәптерде орындайық

1. Берілген кестені дәптерлеріңе толтырыңдар.

Атауы	Қызметі
Концентратор	
Коммутатор	
Маршрутизатор	

2. Интернет желісін пайдаланып, байланыс арналарына жататын желілік сымның анықтамасын және түрлерін дәптерге жазыңдар.

5 Компьютерде орындайық

Мектептегі компьютердің қандай желіге қосылғанын зерттеңдер. Сервердің қай кабинетте тұрғанын анықтаңдар.

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.

Қандай желіні көп пайдаланасыңдар? Неліктен?

§ 2. Компьютерлік желілердің жұмыс жасау принциптері. IP–мекенжай

Естеріңе түсіріңдер:

- *компьютерлік желі дегеніміз не?*
- *желілік компоненттерге нелер жатады?*

Меңгерілетін білім:

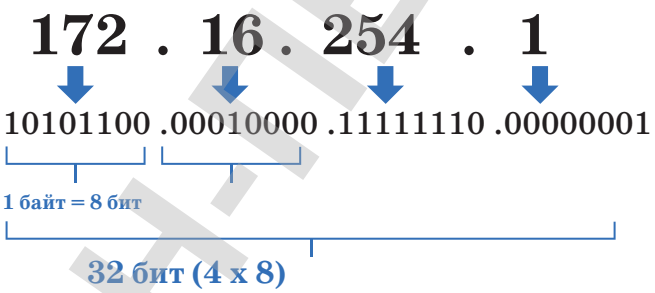
- *«IP-мекенжай» түсінігі;*
- *IP-мекенжай қажеттілігі.*

Терминдер:

- IP-мекенжай;
- провайдер;
- маска.

Қарапайым хат жіберу үшін адамдар хатты алушының мекенжайын – мемлекет, қала, көше, үй және пәтер нөмірін толық жазуы қажет. Компьютерлік желідегі IP-мекенжай да желідегі құрылғы мекенжайы сияқты көрінеді.

IP-мекенжай (ағылш. Internet Protocol Address) – бір түйіннен екінші түйінге ақпаратты жіберу, алу, табу үшін қажетті бірегей мекенжай. **Түйін** – желіге қосылу мүмкіндігі бар кез келген құрылғы (ұялы телефон, компьютер, принтер, концентратор, коммутатор, маршрутизатор және т.б.) (*4-сурет*).



4-сурет. *IP-мекенжай мысалы*

IPv4 мекенжай

1 байттан тұратын әр разрядты **октет** деп атайды. **IPv4** протоколы арқылы IP-мекенжайды 4 миллиардтан астам компьютерге беруге болады. Әр октет максималды 255, ал минималды 0 саннан тұрады. **IPv4** протоколымен анықталған 4 байт көмегімен Интернетте барлық компьютерлерге мекенжай беру қиын

болып бара жатыр. Сондықтан оның орнына 16 байттан тұратын **IPv6** протоколы қолданылады. Бұл жағдайда сандарды әрбір екі байт сайын қос нүктемен ажыратып жазады, мысалы:

2001:0DB8:AA10:0001:0000:0000:0000:00FB.

IP-мекенжай 2 түрге бөлінеді:

1. Ішкі IP-мекенжай (дербес, жергілікті, «сұр»);
2. Сыртқы IP-мекенжай (көпшілік, ауқымды, «ақ»).

Ішкі IP-мекенжайлар тек жергілікті желіде қолданылады және олар үшін мынадай диапазондағы IP-мекенжайлар резервтелген:

- 10.0.0.0 – 10.255.255.255
- 172.16.0.0 – 172.31.255.255
- 192.168.0.0 – 192.168.255.255

IP-мекенжайларды провайдерлер тағайындайды.

Провайдер – ұйым мен жеке тұлғаларға Интернет қызметтерін ұсынатын компания.

Сыртқы IP-мекенжайлар Интернет желісінде қолданылады. Ауқымды желідегі құрылғылар ғана көре алатын бүкіл Интернет желісіндегі бірегей мекенжай. Мысалы, сыныптағы желіге қосылған компьютерлердің IP-мекенжайлары – ішкі, ал оларды сервермен байланыстырып тұрған желілік компоненттің IP-мекенжайы – сыртқы, яғни желідегі компьютерлерден келіп түскен сұранысты серверге (ауқымды желіге) жіберу үшін тағайындалады.

Сонымен қатар IP-мекенжайлар статикалық және динамикалық болып бөлінеді. **Статикалық** (тұрақты) мекенжай – компьютер әрбір желіге қосылған сайын оған қызмет ететін провайдер компаниясы бір ғана мекенжай тағайындап беріп отыратын IP-мекенжай. Ол желідегі басқа құрылғыға беріле алмайды және уақыт өте келе өзгертілмейді. Ал **динамикалық** (тұрақсыз) мекенжай – белгілі бір шектеулі уақытқа ғана провайдермен берілетін мекенжай. Біраз уақыттан кейін бірінші берілген мекенжай екіншісіне, екіншісі – үшіншісіне т.с.с. ауыстырылып отырады. whoer.net сервисі қолданушыларға желідегі құрылғыларының IP-мекенжайларының қай түрге жататынын анықтауға мүмкіндік береді. Бұл сайтқа бірнеше рет Интернетті ажыратып тастап кіріп көреміз. Егер неше рет кірсек те, өзгермейтін бір IP-мекенжайды көрсететін болса, онда біздің

IP-мекенжайымыз – статикалық. Ал егер әр кезде өзгеріп тұрса, онда провайдер бізге динамикалық IP-мекенжай тағайындағанын дегенді білдіреді.

Желідегі компьютерлерде IP-мекенжайдан бөлек **маска** да болады. Ол ішкі желі (подсеть) шекараларын (басқаша айтқанда диапазонын) анықтау үшін: әр компьютерде кімнің онымен бірге бір (ішкі) желіде, кімнің осы желіден тыс орналасқандығын білуі үшін қажет. Мұндағы негізгі мақсат – бір желі ішінде орналасқан компьютерлер пакеттермен «тікелей» алмасу. Ал егер пакеттерді басқа желіге жіберу керек болса, онда пакеттер үнсіз келісім бойынша шлюзге жіберіледі. Осы жағдай қалай іске асатынын қарастырайық.

Ішкі желі маскасы да 32 бит. Бірақ онда нөлдер мен бірліктер алмасып, кезектесіп кездесуі мүмкін емес. Әрқашан бірінші тек бірнеше бірлік, содан соң ғана бірнеше нөл болады, яғни мынадай түрдегі маска болуы мүмкін емес:

120.22.123.12 = 01111000.00010110.01111011.00001100.

Бірақ мынадай маска болуы мүмкін:

255.255.248.0 = 11111111.11111111.11111000.00000000.

Ішкі желінің шекараларын анықтау үшін, компьютер IP-мекенжай және маска арасында биттік көбейту (логикалық ЖӘНЕ) орындайды. Көбейтіндінің нәтижесінде масканың нөлдері тұрған позициялар нөлге айналдырылған мекенжай алынады. Мысалы, 192.168.11.10/21:

11000000.10101000.00001011.00001010
11111111.11111111.11111000.00000000
———————————————————————————————

11000000.10101000.00001000.00000000 = 192.168.8.0

Көбейтінді нәтижесінде алынған мекенжай 192.168.8.0 *ішкі желі мекенжайы* деп аталады.

Хартли формуласы бойынша, 32 биттік Интернет-мекенжайлардың жалпы N – санын есептеуге болады. *Мысалы*: Интернет-мекенжайдағы ақпараттың саны $i = 32$ бит болғанда, N әртүрлі Интернет-мекенжайлардың жалпы саны:

$$N = 2^i = 2^{32} = 4\ 294\ 967\ 296$$ болады.

Қорытынды: Ұзындығы 32 биттік Интернет-мекенжай, Интернетке 4 миллиардтан астам компьютерді қосуға мүмкіндік береді.

1 Сұрақтарға жауап берейік

1. IP-мекенжай қалай құрастырылады?
2. IP-мекенжайдың компьютерлік желіде алатын орны қандай?
3. IP-мекенжайды тағайындау қалай жүзеге асырылады?
4. Провайдердің қызметі қандай?
5. Маска деген не?

2 Ойланайық, талқылайық

1. IP-мекенжай не үшін қажет?
2. Неліктен IP-мекенжайлар ішкі және сыртқы деп бөлінеді?
3. Егер Интернетке қосылған сайын компьютеріміздің IP-мекенжайы өзгеріп отырса, оның IP-мекенжайы қандай түрге жатады?
4. Неліктен ішкі желінің шекараларын анықтау үшін, компьютер IP-мекенжай және маска арасында биттік көбейту (логикалық ЖӘНЕ) орындайды?
5. Егер ұзындығы 32 биттік IP-мекенжай Интернетке 4 миллиардтан астам компьютерді қосуға мүмкіндік берсе, ұзындығы 128 биттік IP-мекенжай Интернетке неше компьютерді қосуға мүмкіндік береді?

3 Талдап, салыстырайық

1. IPv4 және IPv6 протоколдарының бір-бірінен айырмашылықтары қандай?
2. Ішкі және сыртқы IP-мекенжайлар ерекшеліктерін атаңдар.
3. Статикалық мекенжайдың динамикалық мекенжайдан айырмашылығы неде?
4. IP-мекенжай мен желі маскасының арасындағы байланыс қандай?

4 Дәптерде орындайық

IP-мекенжай түрлері бойынша кестені дәптерлеріңе толтырыңдар.

Атауы	Қызметі
Ішкі IP-мекенжай	
Сыртқы IP-мекенжай	
Статикалық IP-мекенжай	
Динамикалық IP-мекенжай	

5 Компьютерде орындайық

1. *whoer.net* сервисі көмегімен сыныптағы компьютерлердің IP-мекенжайын және оның қандай түрге жататынын анықтаңдар.
2. Іздеу жүйелерінің көмегімен желідегі құрылғылардың IP-мекенжайларын анықтауға мүмкіндік беретін тағы қандай сервистері бар екенін анықтаңдар.

6 Ой бөлісейік

Өздерің пайдаланатын құрылғылар қандай IP-мекенжаймен ақпарат алмасады. Сендерге қандай провайдер қызмет көрсетеді? Статикалық мекенжай тиімді ме, әлде динамикалық мекенжай тиімді ме?

§ 3. Компьютерлік желілердің жұмыс жасау принциптері. Домен. Жеке виртуалды желі

Естеріңе түсіріңдер:

- *IP-мекенжай деген не?*
- *IP-мекенжай неліктен қажет?*

Меңгерілетін білім:

- *«домен» түсінігі;*
- *жеке виртуалды желі.*

Терминдер:

- домен;
- протокол;
- виртуалды желі.

Адамға сандық мекенжайды ойда сақтау оңай емес, сондықтан Интернеттің пайдаланушыларына ыңғайлы болу үшін домендік атау жүйесі енгізілді. Бұл жүйеде компьютердің сандық Интернет-мекенжайына сәйкес бірегей домендік атау қойылады. Арнайы серверлік программаның көмегімен сандық және домендік мекенжайлар арасында байланыс орнатылады.

Домен – нүктемен бөлінген символдық аттар. Домендік атау жүйесі иерархиялық құрылымды болады: жоғарғы деңгейдегі домендер, екінші деңгейдегі домендер, үшінші деңгейдегі домендер. Жоғарғы деңгейдегі домендер екі типті болады: **географиялық** және **әкімшілік**. Дүниежүзінің әр еліне екі әріпті кодпен белгіленген өзінің географиялық домені бөлінеді. *Мысалы:* kz – Қазақстан, ru – Ресей, uk – Ұлыбритания, fr – Франция. Қазақстанның ең алғашқы *.kz* жоғары дәрежелі ұлттық домені 1994 жылы 19 қыркүйекте тіркелген. Әдетте, домендер компьютер орналасқан елді анықтайды, демек, ұлттық желінің бір бөлігі болып табылады. *Мысалы,* www.zakon.kz.

Әкімшілік домендер үш немесе одан да көп әріптермен белгіленеді. Әрбір компания оны өздерінің екінші деңгейдегі домендерін тіркеу үшін қолданады. Мысалы, Microsoft компаниясының сайты жоғарғы деңгейдегі әкімшілік *.com* доменінде, ал Мәскеу ашық білім беру институты екінші деңгейлі metodist доменін жоғарғы деңгейлі географиялық *.ru* доменінде тіркеген.

Қызықты ақпарат

Интернеттік байланыс сапасын анықтайтын Британдық Cable сайтының 200 мемлекеттің 163 млн астам желілер талданған 2018 жылғы рейтингінде секундына 4,45 Мбит жылдамдықпен Қазақстан 95-орынды иеленді. Ал рейтингті бастап тұрған мемлекеттер – Сингапур, Швеция және Дания.

Интернет-сервердің домендік аты (оңнан солға қарай) – жоғарғы деңгейдегі доменнің атынан, екінші деңгейдегі доменнің атынан және компьютердің өзінің атынан тұрады. Мысалы, Microsoft компаниясының негізгі серверінің аты – www.microsoft.com, ал институт серверінің аты – iit.metodist.ru.

Желіге қосылған әр компьютердің Интернет-мекенжайы болады, әйтсе де оның домендік аты болмауы да мүмкін. Әдетте, Интернетке телефон желісі арқылы қосылған компьютерлердің домендік аты болмайды.

Протокол – компьютерлік желіде ақпаратты ұсыну, түрлендіру және жіберу стандарты.

Бейнелеп айтқанда, протокол – анықталған желілік тіл. Әртүрлі ғаламдық желілер автономды жұмыс жасаған кезде олар «әртүрлі тілде сөйлесті». Оларды біріктіру үшін жалпы тілін ойлап табу қажет болды. Сондай жалпы желілік тіл болып TCP/IP протоколы қалыптасты. TCP/IP термині деректерді жіберудің екі портоколының атынан құралады:

- TCP (Transmission Control Protocol) – тасу протоколы;
- IP (Internet Protocol – маршруттау протоколы.

TCP/IP протоколының негізінде желінің сервис негізін құрайтын Интернеттің басқа қолданбалы протоколдары жүзеге асырылды. Бұл протоколды желінің программалық және аппараттық құралдары қолдайды. Ол мына жұмыстарды стандарттайды:

- жіберілетін деректерді пакеттерге (бөліктерге) бөлу;
- пакеттерді анықталған маршруттары бойынша тағайындалған пунктіне жеткізу;
- пакеттерді деректердің бастапқы түріне келтіріп жинау.

Бұл ретте пакетті қабылдау-жіберу дұрыстығы, қажетті орында жіберілген барлық пакеттердің жиналу дұрыстығы тексеріледі.

Осылайша, бір мекемеде орналасқан компьютерлерді бір желіге қосып, ауқымды желіге шығаруға мүмкіндігіміз бар. Ал бір-бірінен қашықтықта орналасқан компьютерлерді бір желіге қосу немесе бір компьютерден екінші компьютер жұмысын басқару, қауіпсіздікті күшейту және т.б. қажеттіліктерді қанағаттандыру керек болса, жеке виртуалды желі құру қажет.

Жеке виртуалды желі (VPN – Virtual Private Network) – жоғары жылдамдықты Интернет болған жағдайда желі «үстінен» қауіпсіз (сырттан қолжеткізу мүмкіндігі жоқ) логикалық желі құруға мүмкіндік беретін технология (*5-сурет*).

5-сурет. *Жеке виртуалды желі*

Жеке виртуалды желілердің қолданылу мақсаттары:

- **Интернетке қолжетімдік.** Әдетте, қалалық желілер провайдерлері және ірі ұйымдар кеңінен қолданады. Негізгі артықшылығы – жергілікті желіге және Интернетке екі түрлі желі арқылы қатынау орындалатындықтан, оларға екі түрлі қауіпсіздік деңгейлері тағайындалады.
- **Корпоративті желіге сырттан қолжеткізу.** Бұл жеке виртуалды желіні ойлап табудың басты себебі болған. Алыс қашықтықта орналасқан ұйымның бөлімшелеріндегі компьютерлерін бір корпоративті желіге қосуға және мекемеден тыс жерде (үйде, іссапарда, демалыста) отырып, өзінің жұмыс компьютеріне қолжеткізуге мүмкіндік береді (*6-сурет*).

6-сурет. *Корпоративті желіге сырттан қолжеткізу*

- **Корпоративті желі сегменттерін біріктіру.** Мекеменің желісі қауіпсіздік және сенімділік деңгейлері әртүрлі бірнеше сегменттен тұрады. Мұндай жағдайда сегменттер арасында өзара байланысты ұйымдастыру үшін жеке виртуалды желіні пайдалануға болады. Мысалы, қоймалар желісін жеке белгілі бір сату бөлімшелерінің желісіне қолжетімдікті ұйымдастыруға болады. Бұл жеке логикалық желі болғандықтан, корпоративті елінің басқа жеке желілер жұмысына әсер етпей, барлық қажетті қауіпсіздік талаптарын беруге болады.

1 **Сұрақтарға жауап берейік**

1. Доменнің қанша түрі бар?
2. Протокол деген не?
3. Virtual Private Network күнделікте өмірде қайда қолданылады?
4. Үйде отырып сыныптағы компьютердің жұмыс үстелінде орналасқан файлды көшіріп алу үшін қандай технологияның мүмкіндіктерін пайдалану керек?

2 **Ойланайық, талқылайық**

1. Домен не үшін қажет?
2. Неліктен домендер географиялық және әкімшілік деп бөлінеді?
3. Не себепті жеке виртуалды желілердің өзектілігі артуда?
4. Не үшін жеке виртуалды желіні желідегі қауіпсіздікті күшейту үшін қолданады?

3 **Талдап, салыстырайық**

Географиялық және әкімшілік домендердің бір-бірінен айырмашылықтары қандай?

4 **Дәптерде орындайық**

1. Дәптерлеріңе Қазақстан Республикасында қолданылып жүрген домендерді жазыңдар.
2. Дәптерге қысқаша жеке виртуалды желінің қолданылу мақсатын жазыңдар.

5 **Компьютерде орындайық**

edu.kz, *gov.kz*, *mil.kz* домендерімен жұмыс жасап көріңдер.

6 **Ой бөлісейік**

Бүгінгі сабақта алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.

§ 4. Ақпараттық қауіпсіздік

Естеріңе түсіріңдер:

- *домен дегеніміз не және оның түрлері қандай?*
- *домен аттар жүйесінің қызметі қандай?*
- *жеке виртуалды желінің мақсаты қандай?*

Меңгерілетін білім:

- *«ақпараттық қауіпсіздік» түсінігі;*
- *құпиялылық, тұтастық және қолжетімдік шараларын ұйымдастыру.*

Терминдер:

- ақпараттық қауіпсіздік;
- құпиялылық;
- қолжетімдік;
- тұтастық.

Бүгінгі күні әрекетіміздің басым бөлігі компьютерлік технологияға тәуелді болғандықтан, ақпараттық қауіпсіздік ережелерін қатаң сақтауға міндеттіміз.

Ең алдымен, мемлекеттік және әскери құпиялар, заңнамалық және дәрігерлік құпияларды сақтау маңызды болып саналады. Сонымен қатар жеке басымызға қатысты ақпараттарды да құпия сақтауымыз керек. Олар: жеке басымызды куәландыратын құжаттағы ақпараттар, сайттағы логин мен құпиясөздер және тағы басқа да деректер.

Ақпараттық қауіпсіздік – ақпараттың бұрмалануы немесе жоғалуы салдарынан қолданушыға орны толмас зиян келтіретін кез келген әрекеттерден ақпаратты қорғау.

Ақпараттық қауіпсіздік – ақпараттың *қолжетімдігін, тұтастығын* және *құпиялылығын* қамтамасыз ету үрдісі.

Виртуалды кеңістікте орындайтын мақсаттарымыз бен міндеттерімізге байланысты осы аталған ақпараттық қауіпсіздіктің негізгі үш түсінігінің әрқайсысы үшін түрлі қорғау шаралары қажет болады.

Мысалы, біз виртуалды кеңістікті веб-парақшаларды қарау үшін ғана қолданатын болсақ, онда ақпараттық қауіпсіздікті қамтамасыз етуде бірінші кезекте антивирустық программаларды қолдану, Интернет желісінде жұмыс істеудің қарапайым қауіпсіздік ережелерін сақтау қажет болады.

Ақпаратты қорғау – ақпараттық қауіпсіздікті қамтамасыз етуге бағытталған шаралар кешені. Ақпаратты қорғауда деректерді енгізу, сақтау, өңдеу және тасымалдау үшін қолданылатын ақпарат пен ақпарат қорларының тұтастығын, қолжетімдік оңтайлылығын және құпиялылығын қамтамасыз ете аламыз.

Ақпаратты қорғау нәтижесінде мына сызбадағылар қамтамасыз етілуі керек (*1-сызба*):

***1-сызба.** Ақпаратты қорғау нәтижесі*

Ақпараттың қолжетімдігі компьютер істен шыққанда немесе Интернет арқылы түрлі қауіпті программалардың шабуылы нәтижесінде веб-сайт қолданушысының сұранысына жауап бермеген жағдайда бұзылады.

Ақпараттың тұтастығының бұзылуы ақпараттың ұрлануы немесе бұрмалануы кезінде орын алады. *Мысалы*, электронды пошта хаттарының және басқа да сандық құжаттар мазмұнының өзгеруі.

Ақпараттың құпиялылығы адамдарға құпия ақпарат жария болғанда бұзылады, бұл ақпараттың таралуына себеп болуы мүмкін.

Ақпарат қолданушысы деректердің қолжетімдігінің, құпиялылығының және тұтастығының қауіптерін ескеру керек.

Ақпарат қолжетімдігінің негізгі қауіптері

Оған ақпараттық жүйенің ішкі бас тартуы мен сүйемелдеуші инфрақұрылымның істен шығуын жатқызуға болады.Ақпараттық жүйенің ішкі бас тартуларына мыналар жатады:

- тасымалдау ережелерінің бұзылуы (кездейсоқ немесе ойластырылған);
- жүйенің істен шығуы (сұраныстардың шектен тыс көп болуы, өңделетін ақпарат көлемінің көптігі және т.б.);
- зиянды программалық жасақтамалар;
- программалық және аппараттық жасақтаманың істен шығуы;
- ақпараттардың бұзылуы.

Сүйемелдеуші инфрақұрылымның істен шығуына қатысты қауіптерге:

- байланыс жүйесі, электр сымдары немесе ауа айналымы жұмыстарындағы ақаулар;
- жүйенің қызмет көрсету жұмысының тоқтап қалуы жатады.

Ақпарат тұтастығының негізгі қауіптері

Бұл жерде ақпарат тұтастығының қауіптерін **статикалық тұтастық** пен **динамикалық тұтастық** деп екіге бөлуге болады. Ақпараттың статикалық тұтастығының қаупіне қате ақпараттар енгізу, деректерді өзгерту жатады.

Ақпараттық динамикалық тұтастығының қаупіне деректердің қайталануы, қосымша хабарламалар енгізу, деректерді ұрлау жатады.

Ақпарат құпиялылығының негізгі қауіптері

Ақпарат құпиялылығы пәндік және қызметтік болып бөлінеді. Қызметтік ақпараттар (мысалы, қолданушы құпиясөзі) белгілі бір пәндік аумаққа жатпайды. Өз қызметін асыра пайдаланудан қорғану қиын, бұл – негізгі қауіптердің бірі. Мысалы, жүйелік администратор кез келген файлды, ақпаратты оқып, кез келген қолданушы поштасына кіре алады және т.с.с. Ақпараттық қауіпсіздікті қамтамасыз ету үшін нені, кімнен, қалай және қандай әдістер мен шараларды қолданып қорғау керектігін білу қажет.

Компьютерлік желілердегі ақпарат қауіпсіздігі дербес компьютердегімен салыстырғанда төмен болады, себебі:

- желіде көптеген қолданушылар жұмыс істейді, олардың құрамы үнемі өзгеріп отырады;
- желіге заңсыз қосылу мүмкіндігі бар;
- желілік программалық жасақтамада осалдық байқалады;
- желі арқылы зиянды программалардың шабуыл жасау мүмкіндіктері жоғары.

Қазақстанда ақпаратты қорғауға байланысты мәселелерді ақпаратты қорғау саласындағы Қазақстан Республикасының заңнамасы реттейді.

Кез келген қорғау жүйесінің ең әлсіз тұсы – адам. Кейбір қолданушылар өз құпиясөздерін көзге түсетін, көрінетін жерлерге жазып қояды, тіпті басқаларға да таратуы мүмкін. Мұндай жағдайда кез келген ақпаратқа заңсыз қол жеткізу

мүмкіндігі артады. Сондықтан қолданушыларды ақпараттық қауіпсіздікке үйрету өте маңызды болып табылады.

1 Сұрақтарға жауап берейік

1. Ақпараттық қауіпсіздікті сақтау деген не?
2. Ақпаратты қорғау қажеттілігі туындаған жағдайда не істеу керек?
3. Ақпараттың қауіпсіздік шаралары қандай жағдайда қолданылады?
4. Ақпараттың тұтастығын қамтамасыз ету үшін қандай әрекеттер орындау қажет?
5. Ақпараттың қолжетімдігінің тиімділігін қалай қамтамасыз етуге болады?
6. Ақпараттың құпиялылығын сақтау үшін қандай әрекеттер орындалады?

2 Ойланайық, талқылайық

1. Ақпараттың құпиялылығын, қолжетімдігін, тұтастығын сақтаудың маңызы неде?
2. Неліктен компьютерлік желілердегі ақпарат қауіпсіздігі дербес компьютердегімен салыстырғанда төмен болады?

3 Талдап, салыстырайық

Ақпараттың тұтастығы, ақпараттың құпиялылығы, ақпараттың қолжетімдігі түсініктерінің өзіне тән ерекшеліктерін талдандар.

Ақпараттың тұтастығы	Ақпараттың құпиялылығы	Ақпараттың қолжетімдігі

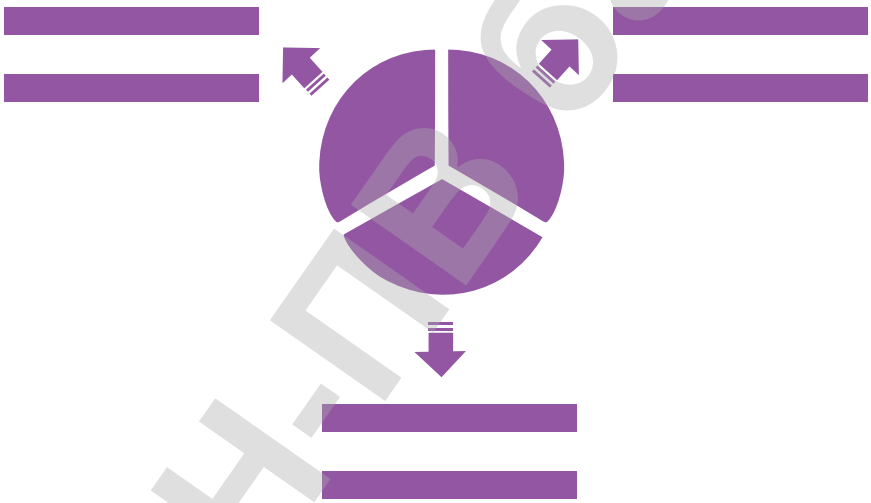
4 Дәптерде орындайық

Кестеге ақпараттың қолжетімдігінің, құпиялылығының және тұтастығының қауіптерін жазып, толтырыңдар.

Қауіптер	Түрлері	Мысал
Ақпарат құпиялылығының қауіптері		
Ақпарат тұтастығының қауіптері		
Ақпарат қолжетімдігінің қауіптері		

5 Компьютерде орындайық

Мәтіндік редакторды пайдаланып (SmartArt), ақпараттық қауіпсіздіктің негізгі түсініктері мен оларға төнетін қауіп түрлерінің классификациясын жасаңдар.



6 Ой бөлісейік

Ақпараттық қауіпсіздіктің негізгі ұғымдарын түсіндіріңдер және сұрақтарға жауап беріңдер.

1. Қалай ойлайсыңдар, ақпараттық қауіпсіздікті қалай қамтамасыз етуге болады?
2. Өз компьютеріңде ақпараттық қауіпсіздік қалай қамтамасыз етілген?

§ 5–6. Ақпаратты қорғау әдістері

Естеріңе түсіріңдер:

- *ақпараттық қауіпсіздік деген не?*
- *ақпараттың қолжетімдігі деген не?*
- *ақпараттың құпиялылығы деген не?*
- *ақпараттың тұтастығы деген не?*

Меңгерілетін білім:

- *қауіпсіздік шаралары;*
- *деректерді резервтік көшіру;*
- *деректерді шифрлау.*

Ақпараттық қауіпсіздікті қамтамасыз етуде ең бастысы – қауіпсіздік шараларын сақтау. Қауіпсіздікті қамтамасыз етуде ақпаратты қорғаудың техникалық, программалық, ұйымдастырушылық шараларын қолдану маңызды.

Ақпаратты қорғаудың **техникалық шараларына** бейнебақылау мен дабыл жүйелерін, ақпараттың таралуы мүмкін барлық жолдарды бұғаттап, алдын алатын басқа да құралдарды жатқызуға болады.

Ақпаратты қорғаудың **программалық шаралары** – белгілі бір деректердің қолжетімдігіне пароль орнату, мәтінді шифрлау, файлдарды уақытша жою мен қауіпті программалардан қорғауды қамтамасыз ету.

Ақпаратты қорғаудың **ұйымдастырушылық шараларына** байланыс арналарын оларға қолжетімдік барынша қиын болатындай етіп тарату мен ұйымның қауіпсіздік саясаты жатады.

Қауіпті программалардың ақпаратқа тигізетін басты зияны – ақпаратты жойып жіберу немесе құпия ақпараттың кілтін жою. Осындай зардапты болдырмас үшін маңызды ақпараттардың **резервтік көшірмесін** сақтап қою керек. Егер мұндай шараларды жасамайтын болсақ, онда өз файлдарымызды қайта қалпына келтіре алмай, жоғалтып аламыз.

Қандай файлдардың резервтік көшірмесін жасау маңызды екендігі жайында қарастырайық. Әрине, ең алдымен өзіміз үшін маңызды жеке файлдардың көшірмесін жасау қажет. Себебі, егер келеңсіз жағдай орын алатындай болса, онда әрқашан операциялық жүйені қайта орнатып, қажетті программаларды жүктей аламыз, ал жекелеген файлдарды қалпына келтіру мүмкін емес. Сол үшін де компьютердегі құжаттар, суреттер, бейнежазбалар мен кез келген жеке ақпараттардың резервтік көшірмесін жасау программасын жиі қосып отыру керек. Сонымен қатар операциялық жүйе, программалар, жүйелік баптаулардың резервтік көшірмесін жасауға болады.

Ақпараттардың резервтік көшірмесін сыртқы жинақтауыштардан бастап, қашықтағы серверлерге дейін сақтауға болады. Әрбір әдістің өз артықшылығы мен кемшілігі бар.

Сыртқы жинақтауыштарға резервтік көшірме жасау

Егер бізде сыртқы USB-диск бар болса, онда резервтік көшірмені бірден сол құрылғыға резервтік көшірменің кіріктірілген қызметтерін қолдана отырып жасауға болады. Windows 8 және 10 үшін **«Файлдар тарихы»** («File History») қызметін қолданамыз. Windows 7 ОЖ-де **«Windows резервтік көшірмесі»** қызметін, ал Mac-құрылғыларда **«Time Machine»** қызметін қолданамыз. Ол үшін сыртқы жинақтауышты компьютерге жиі қосып, резервтік көшірме жасау қызметін қосып отыру керек.

Артықшылығы: жылдамдығы жоғары.

Кемшілігі: егер сыртқы жинақтауыш жоғалып не бұзылатын болса, онда барлық ақпарат көшірмелері де жойылады.

Интернет көмегімен резервтік көшірме жасау

Егер өз файлдарымыздың қауіпсіздігіне сенімді болғымыз келсе, онда Backblaze, Carbonite және Mozy секілді серверлер көмегімен резервтік көшірме жасай аламыз. Бұл программалар автоматты түрде файлдар көшірмесін құрады, олардың жұмысы дербес компьютердің фондық режимінде жүзеге асырылады. Егер файлдар мүлде жойылып кетсе, кез келген уақытта оларды қалпына келтіруге болады.

Артықшылығы: онлайн түрде жүзеге асатын резервтік көшірме файлдармен орын алатын кез келген қиындықтан сақтайды.

Кемшілігі: бұл серверлердің қызметі ақылы. Сонымен қатар файлдар көлемі үлкен болса, алғашқы резервтік көшірме сыртқы жинақтауышпен салыстырғанда көп уақыт алады.

Бұлттық қызметті қолдану арқылы резервтік көшірме жасау

Кейбіреулер бұлттық технологиялар техникалық тұрғыдан алғанда резервтік көшірме жасау қызметін атқара алмайды деп санайды. Файлдарды сыртқы жинақтауыштарға сақтағаннан Dropbox, Google Диск, Microsoft OneDrive немесе осы секілді сервистерге жүктеу тиімді. Егер келеңсіз жағдай орын алатындай болса, осы сервистерде файл көшірмелері сақталатын болады.

Артықшылығы: қарапайым, жылдам және көп жағдайда тегін қызмет.

Кемшілігі: көптеген бұлттық сервистер бірнеше гигабайт көлемді ғана тегін ұсынады, ал оны тек қосымша ақы төлеп, кеңейтуге болады.

Өз файлымызды жоғалтпай, үнемі сақтап жүру үшін Dropbox, Google Дискіге не OneDrive сервистерінде және сыртқы жинақтауыштарға тұрақты түрде резервтік көшірме жасау керек.

Қауіпті программалар ақпаратқа зиянын тигізбес үшін электронды поштамен келетін бейтаныс хаттарды, әсіресе, қосымша файлдары бар хаттарды ашпау қажет. Сонымен қатар хат мәтіндегі сілтемені басу арқылы басқа бетке көшу де қауіпті. Себебі олар вируспен зақымданған веб-беттерге алып баруы мүмкін. Ақпаратты қорғау шараларының тағы бірі – **шифрлау**, яғни арнайы түрде кодтау. Шифрлауды ең алдымен, құпия ақпаратты қорғалмаған байланыс арналары арқылы тасымалдау кезінде қолданады. Мәтін, сурет, дыбыс, деректер қоры және басқа да ақпаратты шифрлауға болады. Деректерді шифрлау мен шифрдан ашу әдістерімен төрт мыңжылдық тарихы бар **криптология** ғылымы айналысады. Ол екі тармақтан тұрады: криптография және криптоанализ.

Криптография – ақпаратты шифрлау әдістері туралы ғылым.

Криптоанализ – шифрдан ашу әдіс-тәсілдері туралы ғылым.

Әдетте, шифрлау алгоритмі барлығына белгілі, ал оны шифрдан ашуға арналған кілт белгісіз болады. Бұл шифрлаудың кодтаудан басты айырмашылығын көрсетеді.

Кілт – шифрлау алгоритмінің параметрі. Оны білу арқылы хабарламаны жасыруға және ашуға болады. Барлық шифрлар (шифрлау жүйесі) екі топқа бөлінеді: симметриялық және асимметриялық (ашық кілтпен). **Симметриялық шифр** – хабарламаны шифрлау мен шифрдан ашуда бір ғана кілт қолдану.

Ассиметриялық шифрда екі кілт қолданылады: олар – бірбірімен математикалық тәуелділік арқылы байланысқан ***ашық*** және ***жабық*** кілттер. Ақпарат барлығына белгілі ашық кілттің көмегімен шифрланады, ал шифрдан ашу хабарламаны алушыға ғана белгілі жабық кілт көмегімен орындалады.

Шифрдың криптоберіктігі – криптографиялық алгоритмнің шифрдан ашуды болдырмауы.

Алгоритм беріктігі – шифрдан ашуда өте көп көлемді есептеулерді талап ететін алгоритм. Жасырылған ақпарат шешілген уақытта өзекті болмай қалады.

Қазіргі программаларда деректерді пароль арқылы шифрлауға болады. Мысалы, OpenOffice.org пен Microsoft Office офистік пакеттері барлық құрылған құжаттарды шифрлауға мүмкіндік береді, яғни ол құжаттарды өзгерту және қарау үшін пароль енгізу керек. Деректерді архивтеуде, мысалы, WinRAR, WinZip архиваторлары файлды архивтен қайта шығару үшін пароль орнату қызметін ұсынады. GnuPG программасы (gnupg.org) ашық программалық жасақтамаға жатады. Мұнда симметриялық, асимметриялық шифрлар, сонымен қатар электронды сандық қолтаңбаның түрлі алгоритмдері де қолданылады.

1 Сұрақтарға жауап берейік

1. Ақпараттық қауіпсіздік шаралары қалай жүзеге асырылады?
2. Резервтік көшірме деген не?
3. Шифрлау көп жағдайда қайда қолданылады?
4. Криптография ғылымының өмірлік жағдайларда алатын орны қандай?
5. Криптоанализ қалай орындалады?
6. Кілтті шифрлауды қолдану қаншалықты маңызды?

2 Ойланайық, талқылайық

1. Ақпараттың қауіпсіздік шараларын сақтаудың маңызы неде?
2. Неліктен кез келген құнды ақпараттың резервтік көшірмесін сақтау маңызды?
3. Шифрлауда алгоритм беріктігі қаншалықты маңызды?

3 Талдап, салыстырайық

Резервтік көшірме жасау әдістерінің артықшылықтары мен кемшіліктерін талдаңдар.

Резервтік көшірме жасау әдістері	Артықшылығы	Кемшілігі
Сыртқы жинақтауыштарға резервтік көшірме жасау		

Резервтік көшірме жасау әдістері	Артықшылығы	Кемшілігі
Интернет көмегімен резервтік көшірме жасау		
Бұлттық қызметті қолдану арқылы резервтік көшірме жасау		

4 Дәптерде орындайық

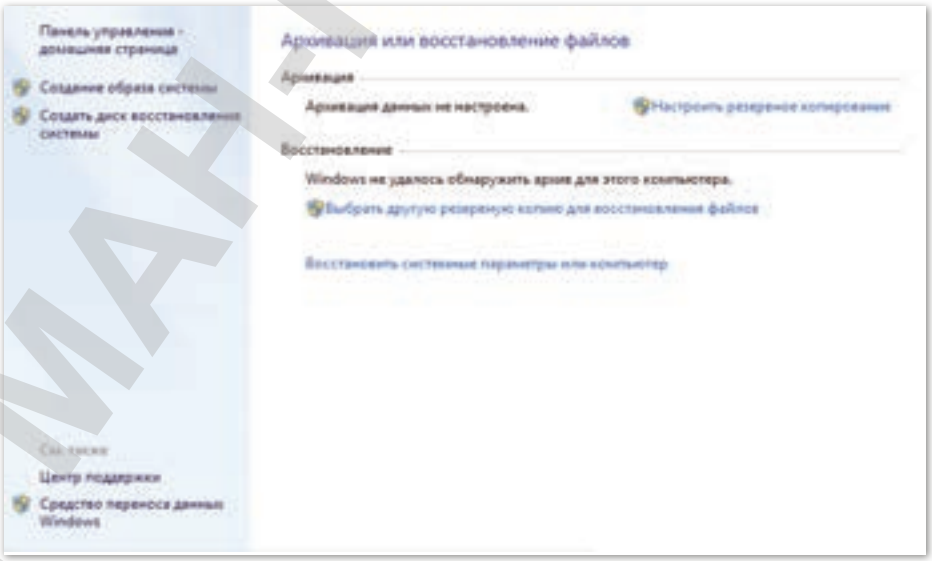
Кестені ақпаратты қорғаудың қауіпсіздік шараларының сипаттамасымен толтырыңдар:

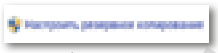
Ақпаратты қорғаудың қауіпсіздік шаралары	Сипаттамасы
Техникалық	
Программалық	
Ұйымдастырушылық	

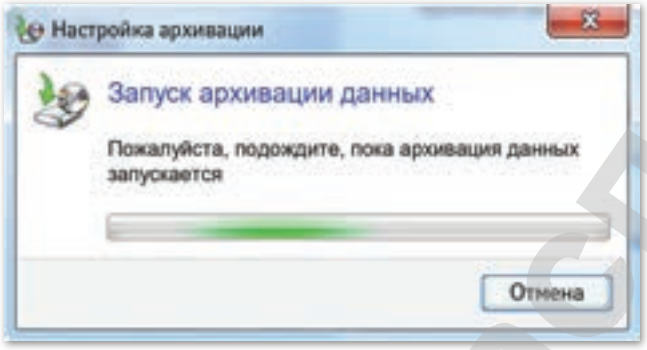
5 Компьютерде орындайық

Архивтеу және қалпына келтіру кіріктірілген құралын қолданып, тапсырманы орындаңдар.

1. Пуск ⇒ Барлық программалар ⇒ Қызмет көрсету ⇒ Архивтеу және қалпына келтіру командаларын орындаңдар.

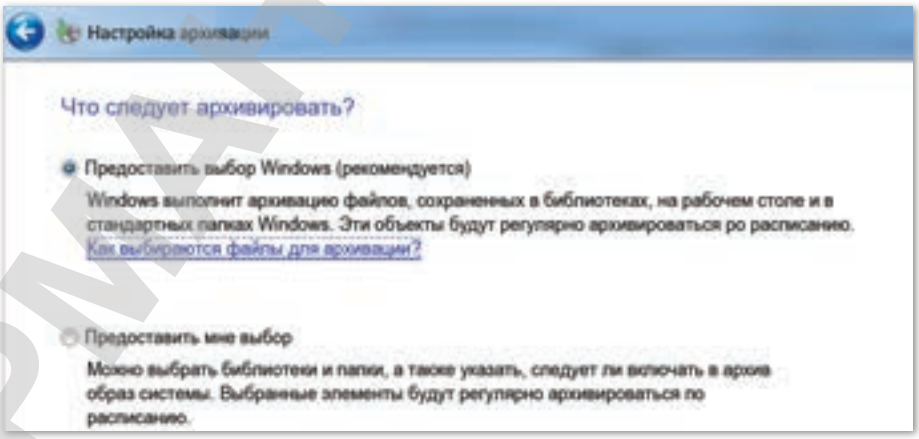


2. Резервтеу қызметін баптаңдар. Ол үшін  сілтемесін шертіңдер. Бұл әрекет «backup» шеберін іске қосады.

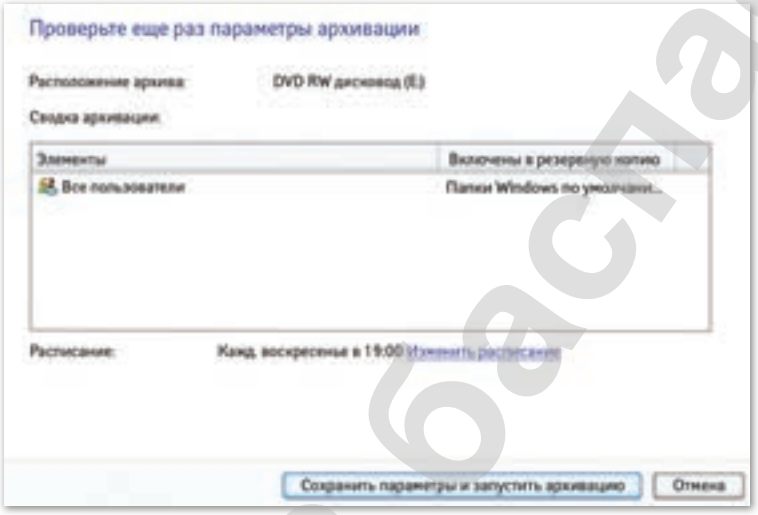


3. Болашақ көшірменің орналасу орнын таңдаңдар:
   - қолжетімді томдар;
   - DVD диск;
   - желілік орналасу және т.б.

4. Қандай құжаттарды архивтеу керектігін Windows ОЖ-нің өзіне қалдырыңдар.



5. Келесі қадамда жүйе құралына архивтейтін нысандарды сеніп тапсырасыңдар ма, әлде өздерің таңдайсыңдар ма? Көрсетіңдер.
6. Архивтеу параметрін тағы да бір рет тексеріңдер.



7. Архивтеу жұмысына біраз уақыт қажет.

8. Тапсырма орындалды. Құжаттардың резервтік көшірмесі жасалды.

6 Ой бөлісейік

Өздерің ақпаратты қорғаудың қандай шараларын қолданасыңдар? Неліктен?

§ 7–8. Идентификациялау әдістері

Естеріңе түсіріңдер:

- *қауіпсіздік шараларына нелер жатады?*
- *деректерді резервтік көшіру дегеніміз не?*
- *деректерді шифрлау деген не?*

Меңгерілетін білім:

- *идентификация;*
- *аутентификация;*
- *сәйкестендіру әдістерін қолдану.*

Ақпараттық технология құралдарын қолданушы қандай да бір ресурс немесе сервиске кіру үшін белгілі бір ақпаратты енгізеді. Мұндай әрекеттер Интернетте логин мен парольді енгізу кезінде орын алады, алайда бұдан басқа да әдіс түрлері бар.

Жүйеге кіру мен жеке деректерді енгізу үрдісі 2 деңгейлі: *идентификация* және *аутентификация.*

Идентификация – серверге тіркелген қолданушының өзіне ғана тән жеке деректерді енгізуі.

Аутентификация – серверде енгізілген жеке деректерді тексеру және қабылдау.

Кейде бұл әдістердің орнына тіркелу сөзі қолданылады.

Тіркелу үрдісі қарапайым, кез келген әлеуметтік желіде **тіркелу** үрдісін алып қарастыруға болады, мысалы:

- **Тіркелу** – мұнда қолданушы электронды поштасын, телефон нөмірі мен парольді енгізеді. Бұл ақпараттар жүйеде қайталанбауы керек, сондықтан бір адам тек қана бір аккаунтпен тіркеле алады.
- **Идентификация** – тіркеуден өту кезінде көрсетілген ақпаратты енгізу, мысалы, электронды пошта не телефон нөмірі.
- **Аутентификацияда Кіру** батырмасы басылғаннан кейін парақша сервермен байланысып, мұндай логин мен парольдің бар-жоқтығы тексеріледі. Егер барлығы дұрыс болса, онда әлеуметтік желі парақшасы ашылады.

Сәйкестендіру әдістері

Сәйкестендіру әдістерінің бірнеше түрі бар, олар бір-бірінен қорғау мен қолдану деңгейлері арқылы ажыратылады.

Пароль арқылы қорғау. Қолданушы өзінен басқа ешкім білмейтін қандай да бір кілтті немесе парольді біледі. Мұнда sms хабарлама алу арқылы идентификациядан өтуді жатқызуға болады. Қолданушы өз аты мен паролін енгізген

кезде сервер осы ақпаратты желіде сақталған ақпаратпен салыстырады. Енгізілген ақпараттар толық сәйкестендірілсе, жүйеге кіру қолжетімді болады.

Парольдің екі түрі бар: *динамикалық* және *тұрақты*. Тұрақты пароль қолданушы талабы бойынша ғана өзгереді. Ал динамикалық парольдер белгілі бір параметр бойынша өзгертіледі. Мысалы, пароль ұмытылып қалған жағдайда сервер жүйеге кіру үшін қолданушыға динамикалық пароль ұсынады.

Арнайы заттарды қолдану. Фирма немесе белгілі бір ұйымдарда қолданылады. Бұл әдісте карточка, арнайы білезік, флеш-жинақтауышты қолдануға болады. Ол заттар жүйеге жақындатылғанда тексеру жұмыстары жүргізіліп, сервер қолданушыны не кіргізеді немесе кіруге рұқсат бермейді.

Биометриялық тексеру. Мұнда көздің ішкі тор қабығы, дауыс, саусақтың ізі қолданылады. Бұл – қорғау жүйелерінің ішіндегі ең қуаттыларының бірі. Сонымен қатар ең сенімді, бірақ ең қымбат жүйе. Заманауи құралдар тек түрлі нүктелерді ғана емес, адамның түрі мен мимикасын да ажырата алады.

Құпия ақпаратты қолдану. Негізінен, программалық жасақтаманы қорғау үшін қолданылады. Бұл жерде дербес компьютер құралдарына орнатылған кэш-браузер, орналасу мекені және тағы басқалар тексеріледі.

Идентификация, аутентификация және тіркелу түсініктерін білу оларды қызметі бойынша дұрыс қолдануға мүмкіндік береді. Ал бұл барлық Интернет пен оның қолданушыларының қауіпсіздігін сақтауға өз ықпалын тигізеді.

1 Сұрақтарға жауап берейік

1. Идентификация қай кезде жүзеге асырылады?
2. Аутентификация қандай жағдайда орындалады?
3. Сәйкестендіру әдістері қалай жүзеге асады?
4. Пароль арқылы қорғау әдісі қалай орындалады?
5. Арнайы заттарды қолдану әдісіне күнделікті өмірден қандай мысал келтіре аласыңдар?

6. Биометриялық тексерудің тиімділігі неде?
7. Құпия ақпаратты қолдану әдісі қай кезде жүзеге асырылады?

2 Ойланайық, талқылайық

1. Жеке тұлғаны идентификациялау не үшін маңызды?
2. Сәйкестендіру әдістерін қолдану тиімді ме?

3 Талдап, салыстырайық

Қарапайым өмірлік мысалды алайық.

Жаңадан жұмысқа орналасқан қызметкер есік алдында отыратын күзетшіге өзінің осы мекемеде менеджер болып жұмыс істейтінін айтады.

Күзетші әдетте жай ғана айтылған сөзге сенбейді, сондықтан жаңа менеджерден шынымен осы мекемеде қызмет ететіндігін дәлелдейтін құжатты талап етеді. Сол кезде қызметкер өзінің сәйкес құжатын күзетшіге көрсетіп, осында қызмет ететінін дәлелдейді.

Мекеме есігі ашылып, күзетші қызметкерді ішке кіргізеді.

Осы мысалдағы мәтіннің қай бөлігі идентификация, қай бөлігі аутентификация, ал қай бөлігі тіркелу екендігін талдаңдар.

4 Дәптерде орындайық

Динамикалық және тұрақты парольдердің сипаттамасын кестеге толтырыңдар.

Пароль түрлері	Сипаттамасы
Динамикалық пароль	
Тұрақты пароль	

5 Компьютерде орындайық

Қолданушының есептік жазбасын құрыңдар.

1. Бастау батырмасын басып, Басқару тақтасын таңдаңдар. Ашылған терезеден Есептік жазбалар және отбасылық қауіпсіздік бөлімінен Қолданушылардың есептік жазбасы жолын таңдаңдар.

2. Қолданушылардың есептік жазбасын қосу немесе өшіру жолын таңдаңдар.
3. Ашылған терезеден Жаңа есептік жазба құру бөлімін таңдаңдар.
4. Қолданушының есептік жазбасының атын енгізіп, типін көрсеткеннен кейін Есептік жазба құру батырмасын таңдаңдар.
5. Есептік жазбаны өзгерту үшін үстінен екі рет шертіңдер.
6. Ашылған терезеде есептік жазбаның атын өзгертуге, пароль орнатуға, суретін өзгертуге, есептік жазбаны өшіруге болады. Пароль орнату бөлімін таңдаңдар.
7. Жаңа пароль енгізу жолағына парольді жазып, келесі жолға орнатылған парольді қайталап жазыңдар.
8. Парольді ұмытып қалған жағдайда еске түсіру үшін кілттік сөзді енгізу жолағына қажетті сөз немесе санды жазыңдар. Пароль құру батырмасын басыңдар.
9. Компьютерді қайта жүктеп, өздерің құрған есептік жазбаға орнатылған пароль арқылы кіріп, жұмысын тексеріңдер.

6 Ой бөлісейік

Ақпараттық технологиялардың дамуына байланысты оқулықта көрсетілген сәйкестендіру әдістерінің қайсысы болашақта қолданыстан шығып қалуы мүмкін?

2-БӨЛІМ

ДЕРЕКТЕРДІ ҰСЫНУ

Күтілетін нәтижелер:

- ондық жүйедегі бүтін сандарды екілік, сегіздік, он алтылық санау жүйесіне және кері аудару;
- логикалық операцияларды қолдану (дизъюнкция, конъюнкция, инверсия);
- берілген логикалық элементтер үшін ақиқат кестесін құру;
- маңызды логикалық элементтердің мақсатын түсіндіру: конъюнктор, дизъюнктор, инвертор;
- логикалық өрнекті логикалық сызбаға өзгерту немесе керісінше;
- басқару құрылғысының (БҚ), арифметикалық-логикалық құрылғының (АЛҚ) және жад регистрінің процессордың бір бөлігі ретінде функциясын сипаттау;
- Unicode және ASCII символдарын кодтау кестесін салыстыру.

§ 9–10. Бір санау жүйесінен екінші санау жүйесіне сандарды аудару

Естеріңе түсіріңдер:

- *идентификациялау деген не?*
- *пароль мен тіркеулік жазбалар не үшін керек?*
- *аутентификация дегеніміз не және қандай түрлері бар?*

Меңгерілетін білім:

- *санау жүйесі;*
- *санау жүйесінің түрлері;*
- *екілік санау жүйесінің ерекшеліктері;*
- *ондық жүйедегі бүтін сандарды екілік, сегіздік, он алтылық санау жүйесіне аудару.*

Терминдер:

- позициялық санау жүйесі;
- позициялық емес санау жүйесі.

В.Лейбниц 1666 жылы алғаш рет кез келген санның екілік санау жүйесінде жазылатыны туралы ұсынысын енгізіп, есептеу құрылғысына екілік жүйені пайдалануға болатындай мүмкіндік барын білді.

Компьютердің жадында сақталатын ақпараттың барлық түрлері – сөздер, сандар, суреттер, компьютер жұмысын басқару программалары – бәрі де екілік сандар тізбегі түрінде жазылады. Сондықтан есептеу техникасында 0 мен 1-ден тұратын екілік сан таңбалары арнайы терминмен *бит* деп аталады да, ол ақпараттың өлшем бірлігі болып табылады.

Санау жүйесі – сандарды жазуға арналған ережелер мен арифметикалық операцияларды орындау мүмкіндігін беретін арнайы сандар жиыны. Санау жүйесі – санды атау және жазу әдістерінің жиынтығы. Санау жүйесі **позициялық** және **позициялық емес** болып, екіге бөлінеді.

Позициялық емес жүйеде сандардың мәні (салмағы) оның тұрған орнына (позициясына) байланысты болмайды, мысал ретінде латын алфавитін пайдаланып, жазылатын рим сандарын айтуға болады: CCLXVII (100 + 100 + 50 + 10 + 7), қай жерде тұрса да C – жүз, L – елу, т.с.с.

Позициялық санау жүйесінде әр санның мәні оның тұрған орнына сәйкес, мысалы, 777,7 санындағы алғашқы 7 саны – 7 жүздікті, екіншісі – 7 ондықты, үшіншісі – 7 бірлікті, соңғысы – бірдің 7/10 бөлігін көрсетеді.

Санау жүйесінің 4 түрі бар:

1) ондық санау жүйесі;
2) екілік санау жүйесі;
3) сегіздік санау жүйесі;
4) он алтылық санау жүйесі.

Ондық санау жүйесіндегі сандарды өрнектеу үшін 0–9 аралығындағы араб цифрлары қолданылады: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Мысалы: 234 = 200 + 30 + 4. 2 – жүздіктер разрядынан, 3 – ондықтар разрядынан, 4 бірліктер разрядынан тұрады. Ондық жүйе позициялық болып табылады, себебі ондық санды жазуда цифрдың мәні оның позициясына немесе санда орналасқан орнына байланысты. Санның цифрына бөлінетін позицияны *разряд* деп атайды. Егер 234 санын қосынды түрінде жазсақ: $2*10^2 + 3*10^1 + 4*10^0$. Бұл жазбадағы 10 саны – санау жүйесін *негіздеуші*. Санның әрбір цифры үшін 10 негіздеуші цифрының орнына байланысты дәрежеленеді және осы цифрға көбейтіледі. Бірліктер үшін – 0; ондықтар үшін – 1, жүздіктер үшін – 2-ге тең негіздеуші дәреже және т.с.с.

Екілік санау жүйесі – негізіне 2 саны алынған позициялық түзіліс бойынша құрылған санау жүйесі. Бұл санау жүйесінде тек екі таңба 0 (нөл) және 1 ғана болады. 2 саны 2-разрядтық бірлігі болып есептеледі де, 10 («бір–нөл» деп оқылады) түрінде жазылады. Келесі разрядтың әрбір бірлігі алдыңғы разрядтан 2 есе артық болады, яғни осы бірліктер 2, 4, 8, 16, …, 2n, … сандар тізбегін құрады.

Сегіздік санау жүйесі, яғни сегіздік негіздеуші санау жүйесі, сегіз цифрдың көмегімен санды көрсетеді: 0, 1, 2, 3, 4, 5, 6, 7. Мысалы, 356 санын негіздеуші 8 қосындысы түрінде жазайық:

$$356 = 3*8^2 + 5*8^1 + 6*8^0.$$

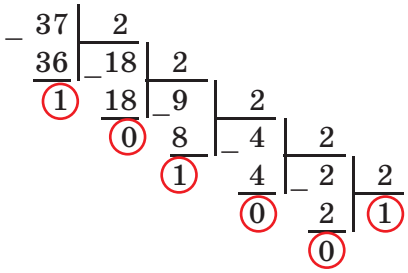
Он алтылық санау жүйесінде санды жазу үшін ондық санау жүйесінің цифрлары 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 және жетпейтін алты цифрды белгілеу үшін ондық сандарының мәні 10, 11, 12, 13, 14, 15 болатын сәйкес латын алфавитінің алғашқы үлкен әріптері: A, B, C, D, E, F қолданылады. Сондықтан он алтылық сандарда, мысалы, 3E5A түрі болуы мүмкін. Осы санды негіздеуші 16 қосынды түрінде жазайық:

$$3E5A = 3*16^3 + E*16^2 + 5*16^1 + A*16^0.$$

Санды ондық санау жүйесінен кез келген санау жүйесіне аудару үшін, бастапқы санды, аударуға қажетті санау жүйесінің негіздеушісіне бүтін санды бөлу қажет. Мұнда қалдық пен бөліндіні ескеру керек. Себебі шыққан бөлінді мәні 0-ге тең болғанша негіздеуішке бөлеміз. Содан кейін барлық қалдықтарды кері ретпен жазсақ, қажетті санау жүйесіндегі санды аламыз.

Мысалы:

$37_{10} \to ?_2$ $\quad$ $103_{10} \to ?_8$ $\quad$ $419_{10} \to ?_{16}$



$37_{10} \to 100101_2$ $\quad$ $103_{10} \to 147_8$ $\quad$ $419_{10} \to 1A3_{16}$

Санды кез келген санау жүйесінен ондық санау жүйесіне аудару үшін олардың әрбір разрядын тұрған орнына байланысты ауысып отырған санау жүйесінің дәрежелеріне көбейтіп, шыққан сандарды қосу арқылы жүргізіледі. Мысалы:

$10110_2 \to ?_{10}$
$10110_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 1 = 16 + 4 + 2 = 22$
$10110_2 \to 22_{10}$

$721_8 \to ?_{10}$
$721_8 = 7 \cdot 8^2 + 2 \cdot 8^1 + 1 \cdot 1 = 448 + 16 + 1 = 465$
$721_8 \to 465_{10}$

$3FA_{16} \to ?_{10}$
$3FA_{16} = 3 \cdot 16^2 + 15 \cdot 16^1 + 10 \cdot 1 = 768 + 240 + 10 = 1018$
$3FA_{16} \to 1018_{10}$

Сонымен қатар сандарды санау жүйелерінің арасындағы сәйкестік кестесі (*1-кесте*) көмегімен аударуға болады.

1-кесте. Санау жүйелерінің арасындағы сәйкестік кестесі

Санау жүйелері			
Ондық	**Екілік**	**Сегіздік**	**Он алтылық**
0	0	0	0
1	1	1	1
2	10	2	2

Санау жүйелері			
Ондық	Екілік	Сегіздік	Он алтылық
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Мысалы, екілік санау жүйесіндегі 1011_2 саны ондық санау жүйесінде 11_{10}-ге тең екенін кестеден көруге болады.

1 **Сұрақтарға жауап берейік**

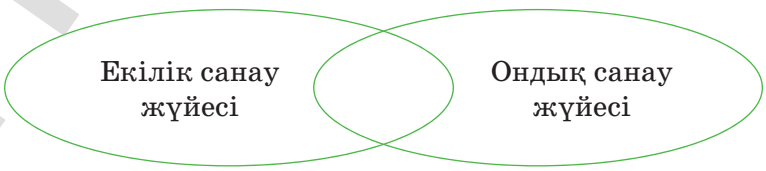
1. Санау жүйесінің қолданылу аясы қандай?
2. Санды ондық санау жүйесінен кез келген санау жүйесіне қалай аударамыз?
3. Санды екілік санау жүйесінен кез келген санау жүйесіне аудару механизмі қандай?

2 **Ойланайық, талқылайық**

1. Дербес компьютер неліктен екілік санау жүйесінде жазылған кодпен жұмыс істейді?
2. Неліктен есептеу техникасында 0 мен 1 ғана қолданылады?
3. Кез келген санды екілік санау жүйесіне аудара аламыз ба?

3 **Талдап, салыстырайық**

Екілік санау жүйесі мен ондық санау жүйесін Венн диаграммасында салыстырып, айырмашылықтарын атаңдар.



4 Дәптерде орындайық

Берілген тапсырманы орындаудың баламасын ұсыныңдар.

Ондық санау жүйесінде берілген сандарды екілік, сегіздік, он алтылық санау жүйесіне аударыңдар.

$12_{10} =$
$57_{10} =$
$642_{10} =$
$841_{10} =$
$123_{10} =$
$456_{10} =$

5 Компьютерде орындайық

Тапсырманы рет-ретімен орындаңдар.

1. Іске қосу ⇒ Барлық программалар ⇒ Стандартты ⇒ Калькуляторды ашыңдар.
2. Калькулятордың түрін ауыстырыңдар. Ол үшін Вид стандартты мәзірінен Программалаушы калькуляторды таңдаңдар.
3. Калькулятордың бұл түрі төрт санау жүйесінде жұмыс істейді: ондық, сегіздік, екілік және он алтылық. Олардың қажеттісін таңдау үшін төрт батырманың бірін шерту керек:
   Hexadecimal (он алтылық),
   Decimal (ондық),
   Octa (сегіздік)
   Binary (екілік).

   Егер санды ондық жүйеде теріп, содан кейін қалған үшеуінің бірінің батырмасын бассаңдар, индикация өрісіндегі ондық сан жаңа жүйеге автоматты түрде аударылады.
4. Осыны пайдаланып, жоғарыда шығарған есептеріңді тексеріңдер.

6 Ой бөлісейік

Сабақта не білдіңдер? Нені үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер.

§ 11–12. Логикалық операциялар (дизъюнкция, конъюнкция, инверсия). Ақиқат кестесін құру

Естеріңе түсіріңдер:

- *санау жүйесі дегеніміз не?*
- *санау жүйесінің қандай түрлері бар?*
- *ондық санау жүйесінен екілікке ауыстыру ережесі қандай?*

Меңгерілетін білім:

- *логика;*
- *логикалық операция түрлері;*
- *логикалық операциялардың ақиқат кестесі;*
- *ақиқат кестесін құру.*

Терминдер:

- дизъюнкция;
- конъюнкция;
- инверсия.

Логика – адам ойлауының түрлері мен заңдары туралы, оның ішінде дәлелдеуге болатын пікірлердің заңдылықтары туралы ғылым. Ғылыми пән ретінде логиканың формальды, математикалық ықтималдықты логика және т.б. түрлері қалыптасқан.

Формальды логика – сөйлеу тілімен білдіретін біздің мазмұнды пікірімізді талдауға байланысты логика.

Ықтималдық логика – кездейсоқ параметрлермен жасалатын сынақтың бірнеше сериясын қолдануға негізделген логика.

Математикалық логика – формальды логиканың бөлігі. Оның дәлме-дәл анықталған нысандары мен пікірлері бар. Олардың ақиқаттығын немесе жалғандығын шешуге болатын ойларды ғана зерттейді.

Пікір – жалған немесе ақиқат болуы мүмкін қандай да бір пайымдау. Мысалы, «Қазақстан Республикасының астанасы – Нұр-Сұлтан», «2*3 = 6» деген ақиқат, ал «тау тегіс», «2*2 = 5» деген – жалған пікірлер.

Математикадағы логикалық жалғаулықтар – күрделі айтылымдарды сипаттайтын **логикалық операциялар**.

Логикалық айтылымдармен жұмыс істеу үшін оларға атау қояды. «Алуа жазда теңізге барады» айтылымы *A* арқылы белгіленсін, ал *B* арқылы «Алуа жазда тауға барады» айтылымы белгіленсін. Сонда «Алуа жазда теңізге де, тауға да барады» құрамды айтылымын *A* және *B* түрінде қысқаша жазуға болады.

Қызықты ақпарат

Логика алгебрасы алғаш рет XIX ғасырдың ортасында ағылшын математигі Джордж Бульдің еңбектерінде пайда болды. Бұл – дәстүрлі логикалық есептерді алгебралық әдістермен шешуге талаптанудың нәтижесі.

Мұндағы «және» – логикалық жалғаулық, A, B – логикалық айнымалылар, олар тек екі мәнде болады: «ақиқат» немесе «жалған», сәйкесінше олар «0» не «1» арқылы белгіленеді.

Математикалық логикада ЖӘНЕ, НЕМЕСЕ, ЕМЕС логикалық операциялары бар және олар ақиқат кестесімен анықталады.

Ақиқат кестесі – логикалық операцияның кестелік түрде ұсынылуы, онда кірістік операндалардың (айтылымдардың) ақиқаттың мәндерінің барлық мүмкін терулері осы терулердің әрқайсысына арналған операцияның шығыстық нәтижесінің ақиқаттық мәнімен бірге аталған.

Логикалық көбейту (конъюнкция)

ЖӘНЕ жалғауының көмегімен қарапайым екі A және B айтылымдарының бір құрамдасқа бірігуі *логикалық көбейту* немесе *конъюнкция*, ал операцияның нәтижесі – *логикалық көбейтінді* деп аталады.

ЖӘНЕ операциясы үшін «$\wedge$», «$\cdot$» немесе «&» белгілерінің бірі пайдаланылады.

ЖӘНЕ логикалық операциясының ақиқат кестесі.

A	B	A&B
1	1	1
0	1	0
1	0	0
0	0	0

Мұндағы A мен B – *иә* немесе *жоқ* мәндерін қабылдай алатын екі айтылым.

Пікірлердің екеуі де ақиқат болғанда, A және B конъюнкциясы ақиқат.

A немесе B пікірлерінің біреуі немесе екеуі де жалған болса, онда A және B конъюнкциясы жалған.

Логикалық қосу (дизъюнкция)

Біріктіруші мағынада қолданылатын НЕМЕСЕ жалғауының көмегімен қарапайым A және B айтылымдарының бір құрамдасқа бірігуі *логикалық қосу* немесе *дизъюнкция*, ал операцияның нәтижесі *логикалық қосынды* деп аталады.

НЕМЕСЕ операциясы үшін «|», «∨» немесе «+» белгілерінің бірі пайдаланылады.

НЕМЕСЕ логикалық операциясының ақиқат кестесі.

A	B	AvB
1	1	1
0	1	1
1	0	1
0	0	0

A немесе *B* пікірлерінің біреуі ақиқат болғанда, *A* немесе *B* дизъюнкциясы ақиқат болады. Ал *A* және *B* пікірлерінің екеуі де жалған болғанда, *A* немесе *B* дизъюнкциясы жалған болады.

Логикалық терістеу (инверсия)

Қарапайым *A* айтылымына ЕМЕС шылауын қосу – *логикалық терістеу* немесе *инверсия* операциясы деп аталады, операцияның орындалу нәтижесінде жаңа айтылым пайда болады.

ЕМЕС операциясы айтылымның үстіне сызықша салу арқылы $\overline{A}$ немесе (¬) белгісі арқылы белгіленеді.

ЕМЕС операциясының ақиқат кестесі.

A	$\overline{B}$
0	1
1	0

Егер бастапқы айтылым жалған болса, онда терістеу ақиқат және керісінше, бастапқы айтылым ақиқат болса, онда терістеу жалған болады. Кейбір айтылымдарды терістегенде *емес* сөзінің орнына *жалған* сөзі қолданылады.

1-мысал. **A·($\overline{B}$)** есебінің ақиқат кестесін құрайық.

A	B	$\overline{B}$	A·($\overline{B}$)
1	1	0	0
1	0	1	1
0	1	0	0
0	0	1	0

2-мысал. $(\overline{A}) \cdot (\overline{B}) \cdot (\overline{C})$ есебінің ақиқат кестесін құрайық.

A	B	C	$\overline{A}$	$\overline{B}$	$\overline{C}$	$\overline{A}\cdot(B)$	$\overline{A}\cdot(\overline{B})\cdot(\overline{C})$
1	1	1	0	0	0	0	0
1	0	1	0	1	0	0	0
0	1	1	1	0	0	0	0
0	0	1	1	1	0	1	0
1	1	0	0	0	1	0	0
1	0	0	0	1	1	0	0
0	1	0	1	0	1	0	0
0	0	0	1	1	1	1	1

1 Сұрақтарға жауап берейік

1. Логика қандай ғылым?
2. Логиканың күнделікті өмірдегі маңызы қандай?
3. Ықтималдық логиканың формальды логикадан қандай айырмашылығы бар?
4. Логикалық операциялардың қолданылу себебі?
5. Логикалық операциялардың орындалу тәртібі қандай?
6. Пікірдің қажеттілігі неде?

2 Ойланайық, талқылайық

1. Формальды логика неге байланысты?
2. Ықтималдық логика не үшін қажет?
3. Математикалық логика мен формальды логиканың арасында қандай байланыс бар?
4. ЖӘНЕ, НЕМЕСЕ, ЕМЕС логикалық операциялары информатикаға не үшін керек?

3 Талдап, салыстырайық

1. ЖӘНЕ, НЕМЕСЕ, ЕМЕС логикалық операцияларын салыстырып, қорытынды жасаңдар.
2. Кестеде берілген терминдерді сипаттамаларымен сәйкестендіріңдер.

Логика	әрқашан құраушы пікірлердің бәрін ақиқат деп ұйғарады.
Пікір	теріске шығаруды тұжырымдау үшін қолданылады.

ЖӘНЕ	адам ойлауының түрлері мен заңдары туралы, оның ішінде дәлелдеуге болатын пікірлердің заңдылықтары туралы ғылым.
ЕМЕС	кездейсоқ параметрлермен жасалатын сынақтың бірнеше сериясын қолдануға негізделген.
Ықтималдық логика	жалған немесе ақиқат болуы мүмкін қандай да бір пайымдау.

4 Дәптерде орындайық

Берілген логикалық айтылымдар үшін ақиқат кестесін құрыңдар.

1. $F(x1, x2, x3) = \overline{x}3 \vee (\overline{x}2 \,\&\, x1 \,\&\, x3)$
2. $F(x1, x2, x3) = \overline{x}1 \,\&\, \overline{x}2 \vee x2 \vee x1 \,\&\, x3$
3. $F(x1, x2, x3) = \overline{x}1 \,\&\, x2 \,\&\, x3 \vee \overline{x}1 \vee x2 \vee x3$

5 Компьютерде орындайық

$F = (A \vee B) \,\&\, (\overline{A} \vee B)$ логикалық функциясының ақиқат кестесін мәтіндік немесе кестелік редакторда бейнелеңдер.

6 Ой бөлісейік

1. Қалай ойлайсыңдар, біз қоршаған ортада логикалық операцияларды қолданып жүрміз бе? Өзара пікірталас жасаңдар.
2. Осы тақырып информатика ғылымындағы өзекті тақырыптардың бірі деп ойлайсыңдар ма?

§ 13–14. Практикум. Логикалық операцияларды қолдану

Мақсаты:

1) негізгі логикалық операциялар мен ақиқат кестесі туралы білімді бекіту;
2) MS Excel электронды кестесін пайдалану арқылы ақиқат кестесін құру дағдыларын қалыптастыру.

Өткен сабақта біз ең жиі қолданылатын логикалық операцияларды қарастырдық. Алайда импликация (ұстану) және эквиваленттілік (тепе-теңдік) сияқты логикалық операциялар да бар. Олардың әрқайсысын егжей-тегжейлі қарастырайық. Сипаттау үшін ақиқат кестелерін қолданамыз.

Логикалық операция / қазақ тіліндегі сәйкестік	Белгісі	Ақиқат кестесі		
импликация (ұстану) / «егер..., онда...», «... болса, онда...»	$\rightarrow$	A	B	$A \rightarrow B$
		0	0	1
		0	1	1
		1	0	0
		1	1	1
эквиваленттілік (тепе-теңдік) / «сонда және тек қана сонда, болса»	$\leftrightarrow$, $\equiv$	A	B	$A \leftrightarrow B$
		0	0	1
		0	1	0
		1	0	0
		1	1	1

Мысалы, $A \vee \overline{B} \wedge C \rightarrow D \leftrightarrow E$.

Орындалу реті:

1. $\overline{B}$
2. $(\overline{B}) \wedge C$
3. $A \vee ((\overline{B}) \wedge C)$
4. $(A \vee ((\overline{B}) \wedge C)) \rightarrow D$
5. $((A \vee ((\overline{B}) \wedge C)) \rightarrow D) \leftrightarrow E$

Енді Excel электрондық кестесі арқылы ақиқат кестесін толтыру үшін практикалық тапсырмаларды орындаңдар.

Жұмысты орындау реті.

1. Excel кестесінде берілген логикалық функциялардың белгіленуін анықтаңдар.
2. Функциялар шеберін пайдаланып, кестені толтырыңдар:

	A	B	C	D	E
1	A	B	$\overline{A}$	A&B	A немесе B
2	жалған	жалған			
3	жалған	ақиқат			
4	ақиқат	жалған			
5	ақиқат	ақиқат			

3. Функциялар шеберін (Мастер функций) пайдаланып, кестені толтыруды жалғастырыңдар:
   А) C2 ұяшығына =НЕ(A2) формуласын,
   D2 ұяшығына =И(A2;B2) формуласын,
   E2 ұяшығына =ИЛИ(A2;B2) формуласын жазыңдар.
   Ә) C2:E2 ұяшықтар диапазонын ерекшеленді.
   Б) Таңдалған блокты C3:E5 ұяшықтарына көшіріңдер.
4. Алынған кестені тексеріңдер.
5. 2-ші параққа көшіңдер.
6. Функциялар шеберін (Мастер функций) пайдаланып, $A \vee A \vee A \vee A$, $A \,\&\, A \,\&\, A \,\&\, A$ функциясының ақиқат кестесін құрыңдар:

A	B	$A \vee A \vee A \vee A$	$A \,\&\, A \,\&\, A \,\&\, A$
жалған	жалған	=ИЛИ(A2;A2;A2;A2)	=И(A2;A2;A2;A2;)
жалған	ақиқат		
ақиқат	жалған		
ақиқат	ақиқат		

7. 3-ші параққа көшіңдер.
8. Функциялар шеберін (Мастер функций) пайдаланып, мына функциялардың ақиқат кестелерін құрыңдар:
   1. $F = (A \vee B) \wedge (\overline{A} \vee \overline{B})$
   2. $F = X \vee Y \wedge \overline{Z}$
   3. $F = X \wedge Y \vee (X \vee Y) \vee X$
   4. $F = A \wedge (B \to C)$
   5. $F = (B \wedge \overline{B}) \leftrightarrow (A \vee D)$

§ 15. Компьютердің логикалық элементтері

Естеріңе түсіріңдер:

- *конъюнкция, дизъюнкция, инверсия деген не?*
- *ақиқат кестесін құру қалай орындалады?*

Меңгерілетін білім:

- *логикалық элементтер;*
- *логикалық сызба.*

Терминдер:

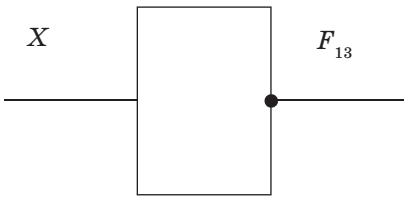
- инвертор;
- конъюнктор;
- дизъюнктор.

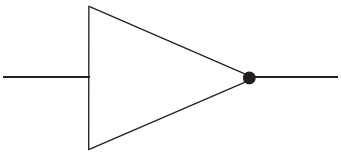
Қызықты ақпарат

1867 жылдан бері американдық логик ғалым Чарльз Сандерс Пирс (оның есімімен логикалық операциялардың бірі – Пирс бағыттауышы аталған) буль алгебрасын кеңейту мен түрлендіру жұмыстарымен айналыса бастайды. Ол – бинарлы логиканың электрлік ауыстырып-қосқыш жұмысымен ұқсас екенін алғаш түсінгендердің бірі. Электрлік ауыстырып-қосқыш токты не өткізеді (ақиқат мәнін қабылдау), не өткізбейді (жалған мәнін қабылдау). Кейінірек Пирс қарапайым электрлік логикалық сызба ойлап табады, бірақ оны жинай алмайды.

Есептеуіш техника дамыған сайын, математикалық логика элементтері компьютерді құрастыру мен программалау мәселелерінде кеңінен қолданыла бастады. Компьютердің логикалық сызбасы, нақты логикалық операцияларды жүзеге асыратын электронды элементтерді біріктіру негізінде жасалады. Бұл электронды элементтерді логикалық элементтер деп атаймыз. **Логикалық элемент** – логикалық функциялардың біреуін орындайтын электронды құрылғы. Элементтің түріне байланысты оның кірісіне бір немесе бірнеше кіріс сигнал (1 – сигнал бар, 0 – сигнал жоқ) беріледі, ал шығысында – бір шығыс сигнал алынады. Логикалық элементтердің атаулары мен шартты белгілері стандартты болып табылады және компьютердің логикалық сызбаларын жасау мен сипаттауда пайдаланылады. Компьютердің процессоры мен жедел жадысы негізгі логикалық элементтер негізінде жасалады. Негізгі логикалық элементтерді қарастырайық:

ИНВЕРТОР терістеу (инверсия) операциясын орындайды. Сызбада былайша суреттеледі: инверторда бір кіріс және бір шығыс болады. Егер кірісте сигнал жоқ болған жағдайда, шығыста сигнал бар болады, немесе керісінше (*7–8-суреттер*).

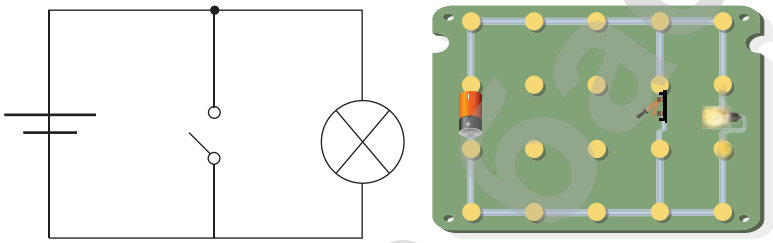




7-сурет. *Ресейлік стандарт*

8-сурет. *ANSI стандарты*

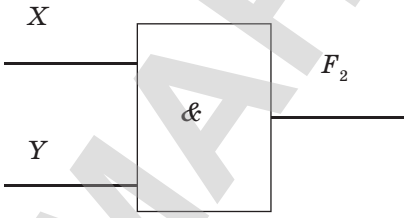
«ЕМЕС» логикалық элементінің қарапайым моделі ретінде электрлік элементтерден тұратын электрлік сызба болуы мүмкін (*9-сурет*).

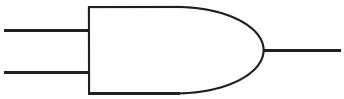


9-сурет. *«ЕМЕС» логикалық элементінің электрлік сызбасы*

Сызбадан көрініп тұрғандай, егер ауыстырып-қосқыш тұйықталса (кірісте 0), онда шам жанады (шығыста 1) және керісінше.

КОНЪЮНКТОР конъюнкция (логикалық көбейту) операциясын орындайды. Сызбада былайша суреттеледі: конъюнкторда екіден көп кіріс және бір ғана шығыс болады. Егер барлық кірістерге сигнал берілсе, шығыста сигнал пайда болады (*10–11-суреттер*).

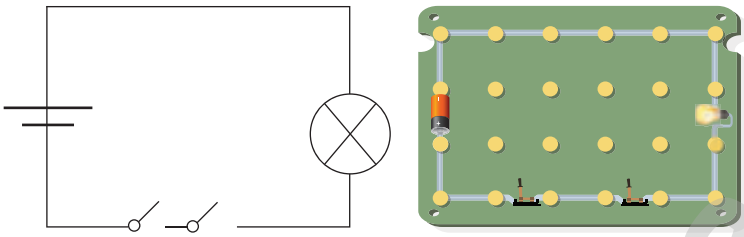




10-сурет. *Ресейлік стандарт*

11-сурет. *ANSI стандарты*

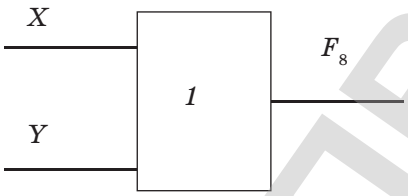
«ЖӘНЕ» логикалық элементінің қарапайым моделі ретінде ток көзінен, шамнан және екі ажыратқаштан тұратын электрлік сызбаны қарастыруға болады (*12-сурет*).



12-сурет. *«ЖӘНЕ» логикалық элементінің электрлік сызбасы*

Сызбадан көрініп тұрғандай, егер екі ажыратқыш та тұйықталған болса (екеуінің де кірістерінде 1), онда тізбек бойымен ток өтіп, шам жанады (шығыста 1).

ДИЗЪЮНКТОР дизъюнкция (логикалық қосынды) операциясын орындайды. Сызбада былайша бейнеленген: дизъюнкторда екіден кем емес кіріс және бір ғана шығыс болады. Егер барлық кірістерге сигнал берілмесе, шығыста сигнал болмайды (*13–14-суреттер*).

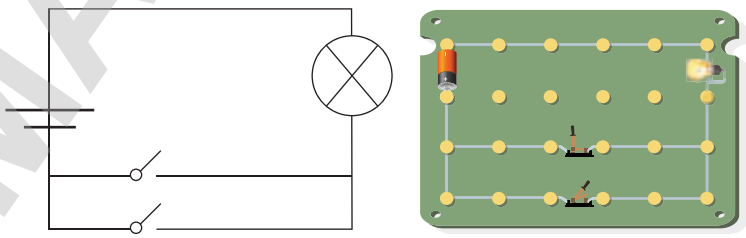


13-сурет. *Ресейлік стандарт*

14-сурет. *ANSI стандарты*

Осы логикалық элементтердің көмегімен кез келген логикалық функцияны логикалық сызба түрінде көрсетуге болады.

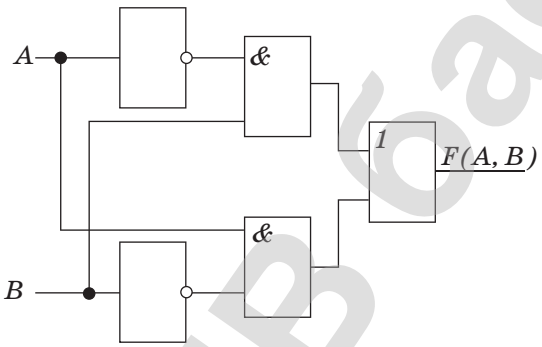
«НЕМЕСЕ» логикалық элементінің қарапайым моделі ретінде электрлік элементтерден тұратын электрлік сызба болуы мүмкін (*15-сурет*).



15-сурет. *«НЕМЕСЕ» логикалық элементінің электрлік сызбасы*

Сызбадан көрініп тұрғандай, егер ажыратқыштардың ең болмаса біреуі тұйықталған болса (шығыста 1), онда тізбек бойымен ток өтіп, шам жанады (шығыста 1).

Мысалы: $F(A, B) = (\overline{A}\&B) \vee (A\&\overline{B})$ логикалық функциясы үшін комбинациялық сызба (*16-сурет*) құру керек. Сызбаны құруды ең соңғы орындалу қажет логикалық операциядан бастаймыз. Біздің жағдайымызда ондай операция логикалық көбейту болып тұр, демек, логикалық сызбаның шығысында дизъюнктор болу қажет. Оған екі конъюнктордан сигналдар беріледі, ал сол конъюнкторларға бір қалыпты сигнал және бір инверттелген сигнал беріледі.



16-сурет. *$F(A, B) = (\overline{A}\&B) \vee (A\&\overline{B})$ функциясының комбинациялық сызбасы*

1 Сұрақтарға жауап берейік

1. Логикалық элементтердің қажеттілігі неде?
2. Компьютердің қандай құрылғылары логикалық элементтер негізінде жасалады?
3. Логикалық элементтердің қандай түрлері бар?
4. Конъюнктор қайда қолданылады?
5. Дизъюнктор қайда қолданылады?
6. Инвертор қайда қолданылады?

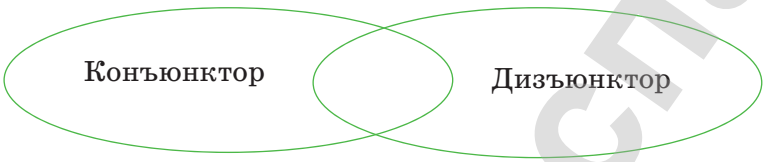
2 Ойланайық, талқылайық

1. Есептеуіш техниканы неліктен логикалық элементтер негізінде құрастырады?
2. Не себепті логикалық элементтер маңызды болып табылады?

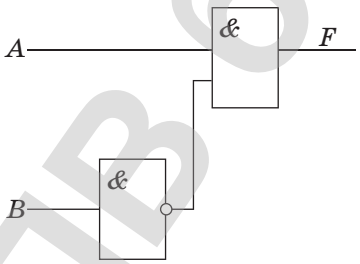
3. Неге инвертор моделі қарапайым электрлік сызба арқылы түсіндіріледі?

3 Талдап, салыстырайық

1. Конъюнктор мен дизъюнктордың бір-бірінен айырмашылықтары қандай? Венн диаграммасы бойынша салыстырыңдар.

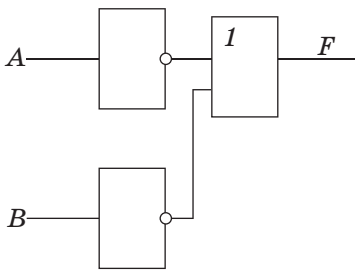


2. Берілген электрлік сызбаның F шығысында қандай нәтиже алынатынын талдаңдар.



4 Дәптерде орындайық

1. Әр кірістің мүмкін болатын барлық сигналдар жиынын ескере отырып, электрлік сызбаның шығысында қандай сигнал болатынын және сызба қандай логикалық пайымдаумен сипатталып тұрғанын анықтап, дәптерге жазыңдар.



2. Төмендегі логикалық функциялар үшін электрлік сызбалар құрыңдар:
   1) $B \,\&\, (A \wedge B)$
   2) $A \,\&\, (B \vee \overline{B})$
   3) $\overline{A \,\&\, (A \vee B \vee C)}$
   4) $A \vee B \vee \overline{C}$

5 Компьютерде орындайық

1. Интернет желісі көмегімен бульдік алгебраның негізгі заңдарын (эквивалентті қатынастарын) оқып, біліңдер.
2. $F = \overline{A} \vee B$ логикалық функциясының сызбасын компьютерде сызып, көрсетіңдер.

6 Ой бөлісейік

1. Тұрақты ток заңдары тұрғысынан логикалық сызбалар моделін жүзеге асыратын электрлік сызбалардың әрекеттерін түсіндіріңдер.
2. Бүгінгі сабақта алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.

§ 16. Компьютердің логикалық негіздері

Естеріңе түсіріңдер:

- *компьютердің логикалық элементтеріне не жатады?*

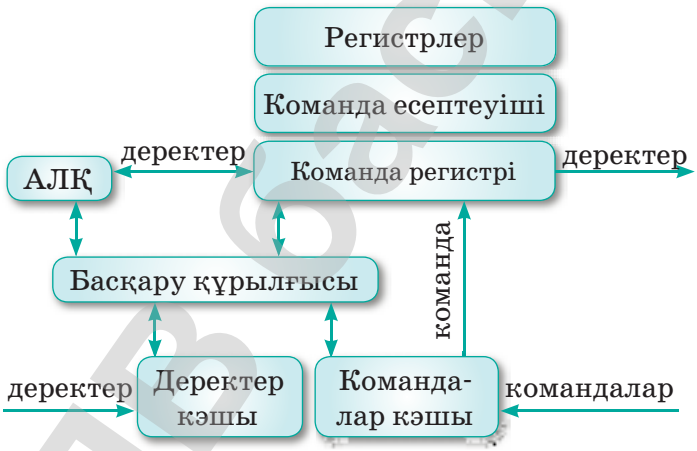
Меңгерілетін білім:

- *процессордың құрылымы;*
- *арифметикалық-логикалық құрылғы;*
- *басқару құрылғысы;*
- *жад регистрлерінің түрлері және атқаратын қызметтері.*

Терминдер:

- АЛҚ;
- БҚ;
- регистр;
- сумматор;
- шина.

1959 жылы алғаш микросызба құрылғаннан бері әлемде мыңдаған түрлі әмбебап және арнайы процессорлар жасалып жатыр. Алайда қазіргі күнге дейін кез келген процессор арифметикалық-логикалық құрылғыдан (АЛҚ), басқару құрылғысынан (БҚ) және жад регистрлерінен тұрады *(2-сызба)*.



2-сызба. *Процессор құрылымы*

Басқару құрылғысы компьютердің барлық құрылғыларының жұмысын басқарады. Ол командалар регистрінен кезекті команданы алып, деректермен не істеу керектігін анықтап, қойылған міндетті орындайтын әрекеттер бірізділігін береді.

Арифметикалық-логикалық құрылғы ақпаратты түрлендірудің арифметикалық және логикалық операцияларын орындайды.

Регистрлер – процессордың ішкі жадысы. Әр регистрдің өз қызметі бар. Мысалы, процессорда 2 санның қосындысын орындау қажеттілігі туындады делік. Ол үшін жадыдан бірінші қосылғышты, содан кейін екінші қосылғышты оқып, екеуін қосып, қажет болса, жедел жадыға жіберу қажет. Процессорға бірінші, екінші қосылғышты және қосынды нәтижесін сақтайтын орын керек. Ол үшін процессордың өзінде *сумматор* деп аталатын ішкі ұяшық болады. Сонымен қатар процессорға келесі команданы жедел жадының қай ұяшығынан оқу керектігі

туралы ақпарат беретін *командалар есептеуіші* болады. Ал, команданың өзі жедел жадыдан алынғаннан кейін *командалар регистрі* деп аталатын ұяшыққа орналастырылады. Командалардың барлығы орындалғаннан кейін, нәтиже регистрден жедел жады ұяшығына көшіріледі.

Мысалдан көріп отырғанымыздай, орындайтын операцияларына байланысты регистрлердің бірнеше түрі болады:

- **сумматор** – әр операцияны орындауға қатысатын арифметикалық-логикалық құрылғының регистрі;
- **командалар есептеуіші** – мазмұны келесі орындалатын команданың адресіне сәйкес келетін басқару құрылғысының регистрі;
- **командалар регистрі** – команданы орындауға қажетті уақыт аралығында, команданың кодын сақтап тұратын басқару құрылғысының регистрі.

Заманауи процессорларда басқа да көптеген құрылғылар кездеседі, алайда жоғарыда аталған бөліктер мен оларды байланыстырушы ішкі деректер шинасы қажетті минимум болып табылады.

Процессордың барлық құрылғылары деректердің ішкі шинасы арқылы бір-бірімен өзара байланысады.

Шина – сандық ақпаратты таратуда байланыстырушы ретінде пайдаланылатын өткізгіштер тобы. Процессордың ішінде 3 негізгі шина бар: деректер шинасы, адрестік шина, басқару шинасы.

Деректер шинасы. Бұл шина арқылы түрлі құрылғылар арасындағы деректер таратылады. Мысалы, жедел жадыдан оқылған ақпарат процессорға өңделуге жіберіліп, өңделген деректер қайтадан жедел жадыға сақталу үшін жіберіледі. Осылайша, деректер шинасы арқылы бір құрылғыдан екіншісіне кез келген бағытта деректерді жіберуге болады.

Адрестік шина процессор жүгінетін құрылғының немесе жады ұяшығының адресін жіберуге арналған.

Басқару шинасы арқылы ақпаратпен алмасу сипаттамасын анықтайтын оқу, жазу, дайындық тәрізді сигналдар жіберіледі.

Жоғарыда аталған процессор құрылғыларының жұмысы процессор өнімділігіне әсер етеді. Сонымен қатар процессордың жылдам жұмыс жасауы мына сипаттамаларға да тәуелді:

- *процессордағы ядролардың саны* бір мезетте бірнеше қосымшаның жұмыс жасауына мүмкіндік береді.
- *процессор жиілігі* – жүйелік шина мен процессор арасында жіберілетін деректердің жылдамдығы.

- *процессордың жылу бөлгіштігі* Ватпен өлшенеді, ол желдеткіштің (кулердің) қандай қуатпен жұмыс жасайтынын көрсетеді.

1 Сұрақтарға жауап берейік

1. АЛҚ және БҚ-ның компьютердің логикалық құрылымындағы маңызы қандай?
2. Регистрлердің қандай түрлері бар?
3. Сумматор қандай әрекеттер орындайды?
4. Командалар регистрі мен командалар есептеуіші арасында қандай айырмашылық бар?

2 Ойланайық, талқылайық

1. Арифметикалық-логикалық құрылғы неліктен негізгі бөлікке жатады?
2. БҚ компьютердің барлық құрылғылар жұмысын қалай басқарады?

3 Талдап, салыстырайық

1. Фон Нейман сәулетінің негізгі бөліктерін талдаңдар (пікір алмасу).
2. Регистрлердің түрлерін салыстырып, зерттеңдер.

4 Дәптерде орындайық

Басқару құрылғысы (БҚ) және арифметикалық-логикалық құрылғы (АЛҚ) (әдетте, олар орталық процессорға біріктірілген), жады, сыртқы жады, енгізу және шығару құрылғыларының байланысын сызба арқылы көрсетіңдер.

5 Компьютерде орындайық

Бүгінгі күнгі жаңа процессорларды Интернет желісі көмегімен іздеп-тауып, осы процессорлар туралы ақпарат беріп, сипаттамаларын жазыңдар.

6 Ой бөлісейік

1. Сабақта не білдіңдер? Нені үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады?
2. Өздерің қандай процессорды таңдайсыңдар? Неліктен?

§ 17–18. Мәтіндік ақпараттарды кодтау принциптері

Естеріңе түсіріңдер:

- *логика дегеніміз не?*
- *логиканың қандай түрлері бар?*

Меңгерілетін білім:

- *кодтау;*
- *мәтінді кодтау түрлері;*
- *ASCII;*
- *Unicode.*

Терминдер:

- кодтау;
- ASCII;
- Unicode.

Қызықты ақпарат

Unicode кодтау кестесіндегі маңызды өзгеріс соңғы рет 1998 жылы Euro символының енгізілуі болды.

Компьютерде мәтін жекелеген символдардан тұрады. Әрбір символды дисплейге шығару үшін оның белгілі бір ережеге сай жазылған машиналық коды қажет болады.

Пернелерде бізге үйреншікті әріптер, сандар, тыныс белгілері және басқа да символдар жазылған. Жедел жадыға олар екілік код арқылы түседі. Сондықтан әр символ 8-разрядты екілік код болып саналады.

Кодтау – символдардың машиналық кодқа сәйкестігі, сонымен қатар ақпаратты компьютерлік тілден адамға түсінікті тілге аудару және керісінше.

Кодтаудың ерекшелігі – мұнда әрбір символға 0-ден 255-ке дейінгі ондық код немесе сәйкесінше 00000000 мен 11111111 аралығындағы екілік код қойылуында. Осылайша, адам символдарды оның жазылуы бойынша, ал компьютер код бойынша ажыратады.

Символдарды байттық кодтау өте ыңғайлы. Себебі байт – жадының ең аз адрестік бөлігі, соған байланысты процессор мәтінді өңдеп отырып, әр символға бөлек мән бере алады. Екінші жағынан, 256 символ әртүрлі символдық ақпараттарды таныстыру үшін әбден жеткілікті.

Компьютердегі барлық символдар 0-ден 255-ке дейін нөмірленген. Әр нөмірге 00000000-ден 11111111-ке дейін 8-разрядты екілік код сәйкес келеді. Бұл код – жай екілік есептеу жүйесіндегі символдардың реттік нөмірі.

Компьютерлік алфавиттегі реттік нөміріне сәйкес барлық символдар кестесі – **кодтау кестесі** деп аталады.

Компьютерді ең алғаш ойлап тапқандар ағылшын тілділер болғандықтан, олар мониторға ағылшын алфавитінің 26 әрпін (үлкен, кіші), 9 тыныс белгісін (. , : !“ ; ? ( )), бос орынды, 10 цифрды, 5 арифметикалық өрнекті (+, –, * , /, ^) және арнайы символдарды

(№, %, _, #, $, ^, &, >, <, |, \) шығару керек болды. Барлығы 100-ден астам символды 0-ден 2^7 (128 орын) екілік сандар жиынымен кодтау келісілді. Бұл кодтау кестесіне **ASCII** (American Standard Code for Information Interchange) аты берілді.

ASCII кестесіндегі *(2-кесте)* символдар қолайлылық үшін он алтылық санау жүйесімен (0–7F) нөмірленді. Кестедегі алғашқы орындарды терілмейтін символдар (0-ден 1F-ке дейін), содан кейін терілетін символдар (20-дан 7F-ке дейін) орналасқан.

***2-кесте*. ASCII кодтау** кестесі

ASCII Code Chart																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Орысша мәтінді бірінші компьютерден екіншісіне ауыстырғанда, бір жүйелік программадан екіншісіне ауысқанда көптеген қиындықтар туындады. Сол себепті орыс әріптеріне ең бірінші хронологиялық стандарт бойынша KOI8 кодтау түрі жасалды. Бұл кодтау түрі сонау 70-жылдары компьютердің БЖ ЭЕМ (ЕС ЭВМ) сериясында қолданылды, ал 80-жылдардың ортасында UNIX операциялық жүйесі қолданысқа енді.

1990 жылдың басында, MS DOS операциялық жүйесінің кезінде, CP866 («CP» «Code Page», мағынасы «кодтау беті») кодтау түрі қолданылды.

Apple фирмасының компьютерлері Mac OS операциялық жүйесінің басқаруымен өздерінің жеке Mac кодтау түрін пайдаланады.

Одан басқа стандарттау жөніндегі Халықаралық ұйым (International Standards Organization, ISO) орыс тілі үшін ISO 8859-5 деп аталатын кодтау түрін ұсынды.

1991 жылдың қаңтар айында символдық кодтауды стандарттау мәселесі шешіліп, жаңа **Unicode** деп аталатын кодтау шығарылды. Ол 16-разрядты кодтау және мұнда әр символға 2 байт алынады. Әрине одан жадының көлемі 2 есе өседі. Бұл кодтау кестесіне 65536-ға дейін символ сыяды. Unicode кестесіне дүниежүзілік алфавит, сонымен қатар көптеген математикалық, музыкалық, химиялық және де басқа символдар енгізілген.

Кириллицаға арналған Unicode кодтау кестесі (Unicode Consortium сайтында жарияланған UNICODE 4.0 кестесінің бір бөлігі) *3-кестеде* бейнеленген.

3-кесте. UNICODE 4.0 кестесінің бір бөлігі

0400	Cyrillic															04
	040	041	042	043	044	045	046	047	048	049	04A	04B	04C	04D	04E	04
0	Ѐ 0400	А 0410	Р 0420	а 0430	р 0440	ѐ 0450	Ѡ 0460	Ѱ 0470	ҁ 0480	Ґ 0490	Қ 04A0	Ұ 04B0	І 04C0	Ӑ 04D0	З 04E0	Ӯ 040
1	Ё 0401	Б 0411	С 0421	б 0431	с 0441	ё 0451	ѡ 0461	ѱ 0471	ҁ 0481	ґ 0491	қ 04A1	ұ 04B1	Ж 04C1	ӑ 04D1	з 04E1	ӯ 041
2	Ђ 0402	В 0412	Т 0422	в 0432	т 0442	ђ 0452	Ъ 0462	Ѳ 0472	≠ 0482	Ғ 0492	Ң 04A2	Ҳ 04B2	ж 04C2	Ӓ 04D2	Ӣ 04E2	Ӳ 042
3	Ѓ 0403	Г 0413	У 0423	г 0433	у 0443	ѓ 0453	ъ 0463	ѳ 0473	◌҃ 0483	ғ 0493	ң 04A3	ҳ 04B3	Ҕ 04C3	ӓ 04D3	ӣ 04E3	ӳ 043
4	Є 0404	Д 0414	Ф 0424	д 0434	ф 0444	є 0454	Ѥ 0464	Ѵ 0474	◌҄ 0484	Ҕ 0494	Ҥ 04A4	Ҵ 04B4	ҕ 04C4	Ӕ 04D4	Ӥ 04E4	Ӵ 044
5	Ѕ 0405	Е 0415	Х 0425	е 0435	х 0445	ѕ 0455	ѥ 0465	ѵ 0475	◌҅ 0485	ҕ 0495	ҥ 04A5	ҵ 04B5	Ӆ 04C5	ӕ 04D5	ӥ 04E5	ӵ 045
6	І 0406	Ж 0416	Ц 0426	ж 0436	ц 0446	і 0456	Ѧ 0466	Ѷ 0476	◌҆ 0486	Җ 0496	Ҧ 04A6	Ҷ 04B6	ӆ 04C6	Ё 04D6	Ӧ 04E6	
7	Ї 0407	З 0417	Ч 0427	з 0437	ч 0447	ї 0457	ѧ 0467	ѷ 0477		җ 0497	ҧ 04A7	ҷ 04B7	Ҋ 04C7	ё 04D7	ӧ 04E7	
8	Ј 0408	И 0418	Ш 0428	и 0438	ш 0448	ј 0458	Ѩ 0468	Ѹ 0478	◌҈ 0488	Ҙ 0498	Ҩ 04A8	Ҹ 04B8	ҋ 04C8	Ә 04D8	Ө 04E8	Ы 048
9	Љ 0409	Й 0419	Щ 0429	й 0439	щ 0449	љ 0459	ѩ 0469	ѹ 0479	◌҉ 0489	ҙ 0499	ҩ 04A9	ҹ 04B9	Ҍ 04C9	ә 04D9	ө 04E9	ы 049

0400															04
A	Ҍ 040A	К 041A	Ъ 042A	к 043A	ъ 044A	ҍ 045A	Ӂ 046A	О 047A	Й 048A	Қ 049A	Ҫ 04AA	Ң 04BA	ӊ 04CA	Ә 04DA	Ө 04EA	
B	Ћ 040B	Л 041B	Ы 042B	л 043B	ы 044B	ћ 045B	ӂ 046B	о 047B	й 048B	қ 049B	ҫ 04AB	ң 04BB	Ӌ 04CB	ә 04DB	ө 04EB	
C	Ќ 040C	М 041C	Ь 042C	м 043C	ь 044C	ќ 045C	Ӄ Ӂ 046C	Ꙋ 047C	Ь 048C	К 049C	Ҭ 04AC	Ҽ 04BC	ӌ 04CC	Ж 04DC	Ӛ 04EC	
D	Ѝ 040D	Н 041D	Э 042D	н 043D	э 044D	ѝ 045D	ӄ ӂ 046D	ꙋ 047D	Ь 048D	к 049D	ҭ 04AD	ҽ 04BD	Ӎ 04CD	ж 04DD	ӛ 04ED	
E	Ў 040E	О 041E	Ю 042E	о 043E	ю 044E	ў 045E	Ӡ 046E	Ꙍ 047E	Р 048E	Ҟ 049E	Ү 04AE	Ҿ 04BE	ӎ 04CE	Ӟ 04DE	Ӯ 04EE	
	Ц 040	П 041	Я 042	п 043	я 044	ц 045	ӡ 046	ѿ 047	р 048	ҟ 049	ү 04A	ҿ 04B		ӟ 04D	ӯ 04E	

1 **Сұрақтарға жауап берейік**

1. Кодтаудың қандай түрлері бар?
2. Кодтау кестесінің қызметі қандай?
3. Кодтау кестесі қалай жасалады?
4. Кодтау әрекетіне күнделікті өмірден қандай мысал келтіруге болады?

2 **Ойланайық, талқылайық**

1. Символдарды байттық кодтау не үшін ыңғайлы?
2. Unicode кодтау кестесінің пайда болу себебін түсіндіріңдер.

3 **Талдап, салыстырайық**

Кодтаудың түрлерін салыстырыңдар.

4 **Дәптерде орындайық**

Бос орындарды толтырыңдар:

1. Компьютерлік алфавиттегі реттік нөміріне сәйкес барлық символдар кестесі – … деп аталады.

2. … – бұл символдардың машиналық кодқа сәйкестігі, сонымен қатар ақпаратты компьютерлік тілден адамға түсінікті тілге аудару және керісінше.
3. ASCII кестесіндегі символдар қолайлылық үшін … санау жүйесімен нөмірленді.

5 Компьютерде орындайық

Excel кестелік процессорының көмегімен ASCII кодтау кестесін құрыңдар, онда символдар автоматты түрде экран бетінде өздерінің ондық нөмірлеріне сәйкес бейнелетін болады (сәйкес мәтіндік функцияны қолдану).

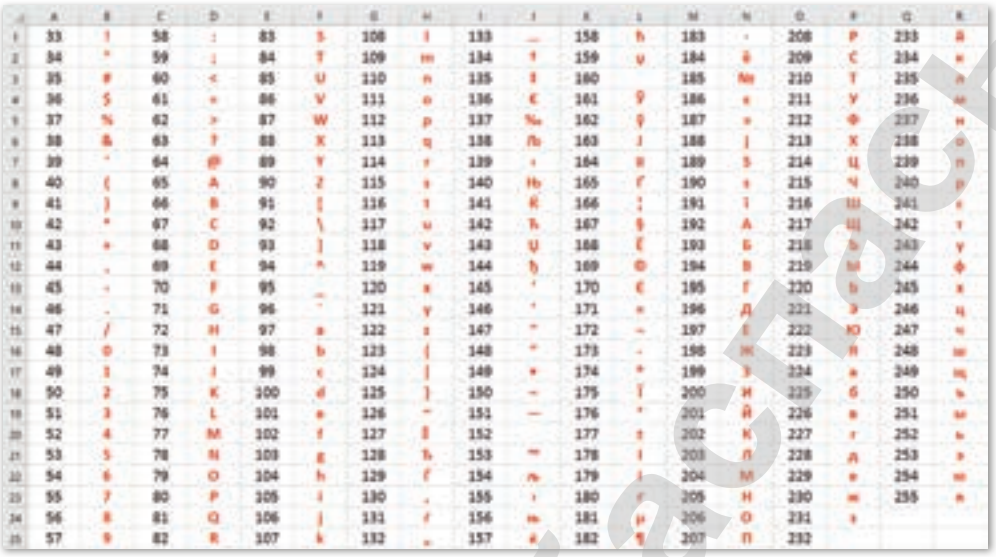
1. MS Excel программасын ашыңдар.
2. 33-тен 255-ке дейінгі сандарды A1 ұяшығынан бастап енгізіңдер (әрбір бағанға 25 жолдан бір баған тастап отырып: A, C, E, … , S)

	A	C	E	G	I	K	M	O	Q
1	33	58	83	108	133	158	183	208	233
2	34	59	84	109	134	159	184	209	234
3	35	60	85	110	135	160	185	210	235
4	36	61	86	111	136	161	186	211	236
5	37	62	87	112	137	162	187	212	237
6	38	63	88	113	138	163	188	213	238
7	39	64	89	114	139	164	189	214	239
8	40	65	90	115	140	165	190	215	240
9	41	66	91	116	141	166	191	216	241
10	42	67	92	117	142	167	192	217	242
11	43	68	93	118	143	168	193	218	243
12	44	69	94	119	144	169	194	219	244
13	45	70	95	120	145	170	195	220	245
14	46	71	96	121	146	171	196	221	246

3. B1 ұяшығына =СИМВОЛ(A1) формуласын енгізіп, Enter пернесін басыңдар.

	A	B	C	D	E
1	33	=СИМВОЛ(A1)	58		83
2	34		59		84

4. Автотолтыру қызметін қолдана отырып, бағанның қалған ұяшықтарына осы формуланы көшіріп, қойып шығыңдар: B, D, F,…, T.
5. Нәтижесінде ASCII кодтау кестесі құрылды.



6 Ой бөлісейік

1. Сабақта не білдіңдер? Нені үйрендіңдер?
2. Қазақ тілі латын алфавитіне толық көшкенде, кодтау кестесінде өзгерістер бола ма?

3-БӨЛІМ

АЛГОРИТМДЕУ ЖӘНЕ ПРОГРАММАЛАУ

Күтілетін нәтижелер:

- функциялар мен процедураларды пайдаланып, программалау тілінде код жазу;
- жолдарды өңдеу үшін процедуралар мен функцияларды пайдалану;
- ақпаратты оқу және жазу үшін файлдарды пайдалану;
- практикалық есептерді шешу үшін сұрыптау алгоритмдерін іске асыру;
- практикалық есептерді шешу үшін графтардағы алгоритмдерді іске асыру.

§ 19. Пайдаланушы функциялары мен процедуралары. Процедуралар

Естеріңе түсіріңдер:

- *мәтіндік ақпаратты кодтау принциптері қандай?*
- *кодтау кестесі қалай қолданылады?*
- *UNICODE және ASCII кодтарының айырмашылығы қандай?*

Терминдер:

- процедура;
- параметр;
- қызметші сөз: def.

Меңгерілетін білім:

- *процедуралар;*
- *процедуралар мен функциялардың қолданылуы.*

Машиналық кодпен программа құру күрделі. Сондықтан қазіргі кезде барлық программалар программалау тілдері арқылы құрылады. Python тілінде программалау – компилятор және интерпретаторда бірге жұмыс істеуге мүмкіндік беретін программалау тілі.

Процедура — бірнеше іс-әрекетті орындайтын көмекші алгоритм. Python тілінде процедура `def` қызметші сөзінен басталып, бос не бос емес жақшадан және қос нүктеден тұрады. Оның жазылуын мысалда қарастырайық:

```
def Err(): # процедураны анықтау
   print ("Қате: дұрыс емес деректер")
n = int (input('оң сан енгізіңдер'))
if n < 0:
   Err() # процедураны шақыру
```

- процедура коды негізгі программада шақырылмай тұрып жазылады;
- программада бірнеше процедура болуы мүмкін;
- процедура дұрыс жұмыс істеуі үшін, оны негізгі программадан немесе басқа процедурадан шақыруымыз керек;
- процедура шақырылғанға дейін анықталған болуы керек. Процедураны анықтау қызметтік `def` сөзінен басталады;
- процедураны шақыру қос жақшаның алдындағы атымен жүзеге асырылады. Мысалы, `Err()`;
- процедураны программа барысында пайдалану кодты қысқартады және программаның тез оқылуын қамтамасыз етеді.

Процедура параметрі

Python программалау тілінде параметрлердің пайдаланылуын мысалда қарастырайық:

Мысал: енгізілген символды жаңа жолдан шығаратын процедура программасын жазу.

```
def printChar(s):
   print (s)
sim = input ('символды енгіз')
printChar(sim) # бірінші шақыру, енгізілген символды шығару
printChar('*') # екінші шақыру, * нәтиже
```

Ауқымды айнымалы – негізгі программада процедураға меншіктелетін мән. **Жергілікті айнымалыны** (ішкі) негізгі программадан және басқа процедурадан алу мүмкін емес, тек сол процедура деңгейінде пайдаланылады. Процедура параметрі – жергілікті айнымалылар.

1 Сұрақтарға жауап берейік

1. Процедура деген не?
2. Процедураны қай кезде қолдана аламыз?
3. Қандай программаларды білесіңдер?
4. Python тілінде процедура неден басталады?
5. Ауқымды айнымалы деген не?
6. Процедура параметрлеріне нелер жатады?

2 Ойланайық, талқылайық

1. Процедура не үшін қажет?
2. Не себепті программалар программалау тілдері арқылы құрылады?
3. Неліктен процедураны программа барысында пайдалану қажет?

3 Талдап, салыстырайық

1. Процедураның қолданылу қызметтерін талдаңдар.
2. Процедура қызметтерін салыстырып, ұқсастықтарын анықтаңдар.

4 Дәптерде орындайық

1. Кестені дәптерлеріңе толтырыңдар.

Атауы	Қызметі
Процедура	
Ішкі программа	
Параметр	

2. Процедура анықтамасын және түрлерін дәптерге жазыңдар.

5 Компьютерде орындайық

Енгізілген санның барлық бөлгіштерін процедура арқылы экранға шығаратын программа құрыңдар (бір жолда).

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер. Программаның күнделікті өмірде қолданылуына қандай мысал келтіресіңдер?

§ 20. Практикум. Процедураларды пайдаланып программалау тілінде код жазу

1-мысал. Енгізілген оң сандарды процедураны пайдаланып, баған бойынша жауапқа шығаратын программа құрыңдар.

Шешімі:

```
n = int(input())
n = abs(n)
def printDigits (n):
   n = str (n)
   for i in range (0, len(n)):
      print (n[i])
printDigits(n)
```

Программа нәтижесі:

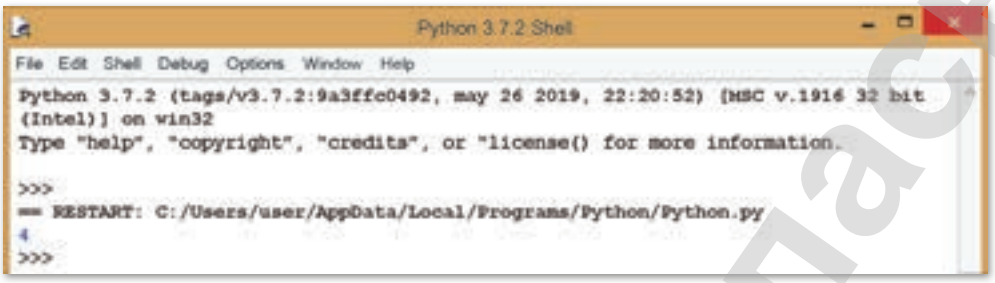
```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, may 26 2019, 22:20:52) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits", or "license() for more information.

=== RESTART: C:/Users/user/AppData/Local/Programs/Python/Python.py ***
5487231
5
4
8
7
2
3
1
>>>
```

2-мысал. Ауқымды айнымалының параметрін шығаратын программа құрыңдар.

Шешімі:

```
x = 3 # ауқымды айнымалы
def pr(a): # параметрлі процедура
   a = 4 # жергілікті айнымалы
   print (a) # 4
      pr(x) # ауқымды айнымалы параметрін шығару (3)
```

Программа нәтижесі:



1-тапсырма. $p_i x^2 + q_i x + r_i = 0$ квадрат теңдеулер тобының шешімін табудың программасын жазыңдар, мұндағы p, q, $r - k$ элементтен тұратын нақты сандар массиві. Бір теңдеуді шешуді процедура түрінде бейнелеңдер.

2-тапсырма. n натурал саны берілген. 1, 2, 3..., n екі натурал сандардың квадраттарының қосындысын көрсете алатын барлық сандарды табыңдар (толық квадраттарды таба алатын процедураларды анықтау керек).

3-тапсырма. a, b, c массивтеріндегі максимум элементтерін және олардың нөмірлерін табу.

4-тапсырма. Параллелепипед бетінің көлемі мен ауданын есептейтін процедура жазыңдар.

5-тапсырма. Аргумент ретінде алынған екі бүтін саннан ең үлкенін қайтаратын функцияны жазыңдар.

6-тапсырма. Енгізілген сандарды баған бойынша соңғысынан бастап экранға шығаратын процедура құрыңдар.

§ 21. Пайдаланушы функциялары мен процедуралары. Функциялар

Естеріңе түсіріңдер:

- *процедуралар деген не?*
- *процедуралар не үшін пайдаланылады?*
- *программа барысында процедураны неге пайдаланады?*

Меңгерілетін білім:

- *ішкі программалар;*
- *функциялар туралы түсінік;*
- *функцияның түрлері.*

Терминдер:

- функция;
- оператор;
- айнымалы.

Нақты есептерді шешу алгоритмін сипаттауда есептеу процесінің әртүрлі кезеңінде қайталанатын іс-әрекет бөлігін бірнеше рет қайталап орындау қажеттілігі туады. Әрине қайталанатын бөлікті орындайтын операторлар тобын программада әр кезде жазуға болады, бірақ ол тиімді емес. Бұл әрекеттерді ішкі программаға біріктіру тиімді, оларды бір рет сипаттап және қажет уақытта оған кіретін бастапқы берілгендерді өзгертуге болады.

Ғалым В.Ф.Очков ішкі программа ұғымына аса поэтикалық түсініктеме берді: «Ішкі программа – бірнеше рет айтатын ән қайырмасы, ал ән мәтінінде оны тек бір рет жазады».

Математикалық тұрғыдан қарағанда кез келген ішкі құрылым – өзіндік құрылым ретінде қарастыруға болатын бүтіннің тұйық бөлігі: ішкі топ – топ, ішкі алгебра – алгебра, ішкі кеңістік – кеңістік және т.б.

Python тілінде ішкі программалардың екі түрі бар: **процедуралар** мен **функциялар**. Ортақ белгілерін сипаттай отырып, біз «ішкі программа» терминін қолданамыз. Егер мәтінде «*процедура*» немесе «*функция*» термині кездессе, онда бұл ақпарат тек бір ішкі программаның нақты түріне, яғни процедураға немесе функцияға тән болады.

Функция – атауы бар программа барысында шақырылатын программа бөлігі.

Функция шақырылғанға дейін анықталған болуы керек.

- Функцияның процедурадан айырмашылығы – мәнді қайтаруында.
- Функция мәнін қайтару үшін `return` операторы пайдаланылады.

- Функцияны шақыру оның атын жазып, мәнді шығарумен аяқталады.

Python тілінде функцияның жазылу синтаксисі:

```
def Функция_Аты (Параметрлер_Тізімі):
        Өрнектер
```

Python тілінде функцияны анықтаудан бұрын `def` қызметші сөзі жазылады. `def` (функция тақырыбы) арқылы функцияны шақыруды көрсетеміз.

Функция тақырыбынан кейін жабылған жақша қос нүктемен аяқталады және одан кейінгі қатарда есептеуге қатысты өрнектер жазылады (TAB пернесін бір шерту арқылы):

– функцияға атау бергенде айнымалыға атау бергендей, алдымен, функцияның аты латын әріптерімен жазылу керек;
– параметрлер тізімі функцияға берілетін мәндерден тұрады. Бұл параметрлер үтір арқылы жазылады.

Python тіліндегі функциялардың бір бөлігі тілдің синтаксисінде пайдаланылатын **кіріктірілген функциялар** болып табылады. Оларға мысалы, `int`, `input`, `randint`, `print` қызметші сөздерін жатқызамыз.

Кіріктірілген функцияларды екі топқа бөлеміз:

1. Символдармен жұмыс жасайтын функциялар: – `ord()`, `chr()`, `len()`.
2. Математикалық функциялар: – `abs()`, `round()`, `divmod()`, `pow()`, `max()`, `min()`, `sum()`.

Пайдаланушы функциясын құрудың мысалын қарастырайық:

Сандардың қосындысын есептеу.

```
def sumD(n): # функцияны параметрімен анықтау
sum = 0
        while n! = 0:
            sum + = n % 10
            n = n // 10
        return sum # функцияның мәнін қайтару
# негізгі программа
    print(sumD(1075)) # функцияны параметрімен
шақыру
```

Қайтымды мәндер

Функциялар мен процедуралардың негізгі айырмашылығы – олардың қайтаратын мәндерінің санында.

Кез келген функция өз жұмысын аяқтаған соң, негізгі программаға (немесе шақырылған ішкі программаға) мән қайтаруы керек.

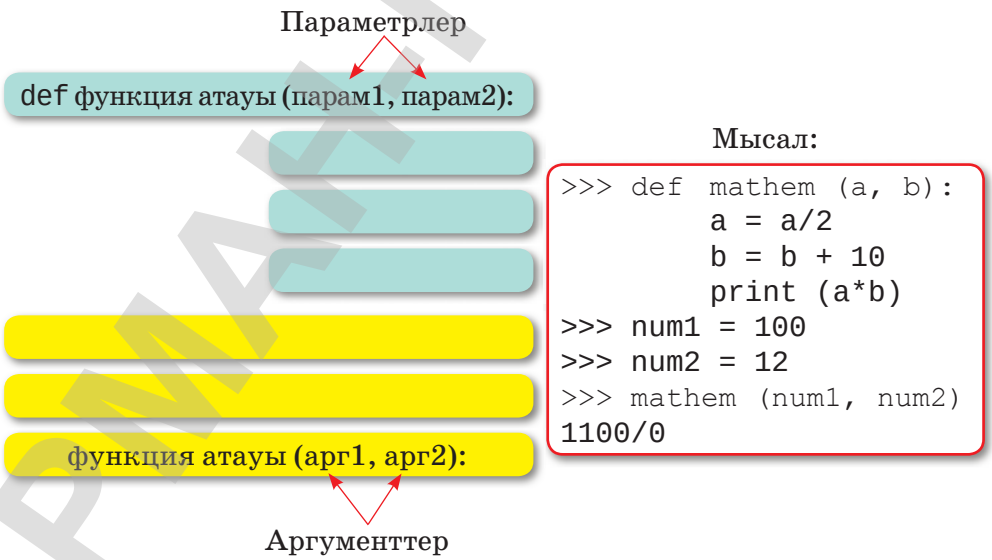
Нәтижені қайтару үшін функцияның атымен сәйкес келетін аты бар арнайы «айнымалы» қолданылады. «Айнымалыға» мән меншіктеу операторы міндетті түрде тым болмағанда бір рет функция денесінде қолданылуы тиіс.

Функция әрқашан теңдіктің оң жағында орналасатынын және атынан кейін жақша ішінде функция аргументінің мәні жазылатынын көреміз. Теңдік белгісінің сол жағында соңғы кезде функция мәні меншіктелетін айнымалы тұрады.

Ауқымды айнымалылар – программа басында кез келген ішкі программаларды хабарлағанға дейін хабарланған деректер типтері, тұрақтылар мен айнымалылар. Бұл нысандар барлық программада, сонымен қатар оның барлық ішкі программаларында көрініп тұрады. Ауқымды айнымалылар программаның барлық жұмысы кезінде бар болады.

Жергілікті айнымалылар белгілі бір ішкі программаның ішінде хабарланады және тек осы ішкі программаға ғана көрінеді. Олар ішкі программа шақырғанша орындалмайды.

Функция синтаксисі (*17-сурет*):



17-сурет. *Функцияның синтаксистік жазылуы*

Python программалау тілінде аргументті қабылдайтын және мәнін қайтаратын нысан **функция** деп аталады. Функциялар `def` қызметші сөзі арқылы сипатталады.

Қарапайым функцияны сипаттау:

```
def add(x, y):
   return x + y
```

Мұндағы `return` нұсқаулығы мәнді қайтаруын көрсетіп тұр. Мысалда функция x пен y мәнінің қосындысын қайтарады.

Енді осы функцияны есеп шығару барысында шақыру:

```
>>>
>>> add(1, 10)
11
>>> add('abc','def')
'abcdef'
```

Программалау кезінде функциялар тек деректерді қайтармайды, сонымен қатар функциядағы параметрлерді қолдануды жүзеге асырады. Кез келген сан параметр бола алады.

Параметрлер функцияны шақыру кезінде мәні меншіктелетін жергілікті айнымалылар болып табылады. Функция шақырылған кезде берілетін нақты мәндер **аргумент** деп аталады.

Функция аргументтері нақты көрсетілмеуі немесе келісім бойынша аргумент қабылдауы мүмкін. Функцияны әртүрлі аргументтер санымен шақыра аламыз. Бұл жағдайда аргументтің алдына * белгісі қойылады және негізінен, бұл аргумент тізбек түрінде анықталады.

Функция денесі функцияның өзін шақырса, онда бұл жағдайды **рекурсия** деп атаймыз.

Функция атауының анықталмай тұрып құрылуы Python тілінің ерекшелігіне жатады. Мұндай функцияларды **Лямбда-функциялар** деп атайды.

Лямбда-функцияның анықталуы:

lambda аргументтер: нәтижелер

Лямбда-функция функция аргументі түрінде де беріледі:

```
fun1(lambda x,y:x*y + pow(x,2) + pow(y,2), 1,4)
```

Лямбда-функция өрнекте қолданылады:

```
z = 1 + (lambda x, y: x*y + pow(x, 2) + pow(y, 2))(1,3)**2
```

1 Сұрақтарға жауап берейік

1. Функция деген не?
2. Функция қандай жағдайларда қолданылады?
3. Функцияның қандай түрлері бар?
4. Қайтымды мәндер деген не?

2 Ойланайық, талқылайық

1. Неліктен ауқымды айнымалылар программа басында хабарланады?
2. Не себепті жергілікті айнымалылар ішкі программа шақырғанша орындалмайды?

3 Талдап, салыстырайық

Жергілікті айнымалы мен ауқымды айнымалының қызметтерін салыстырып, ұқсастықтарын анықтаңдар.

4 Дәптерде орындайық

1. Төмендегі кестені дәптерлеріңе толтырыңдар.

Атауы	Қызметі
Функция	
Айнымалылар	
`return`	
`Def()`	
`Print()`	

2. Функцияға берілген есепті компьютерде орындап, функцияның анықтамасын және түрлерін дәптерге жазыңдар.

5 Компьютерде орындайық

Функцияны пайдалана отырып, *b*, *c*, *d* массив элементтерінің қосынды мен көбейтіндісін есептеңдер.

6 Ой бөлісейік

1. Сабақта не білдіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қайда қолдануға болады? Мысал келтіріңдер.
2. Жаңа терминдермен кроссворд немесе ребус құрастырыңдар: функция, аргумент, параметр және т.б.

§ 22. Практикум. Функцияларды пайдаланып, программалау тілінде код жазу

1-мысал. Функцияны пайдаланып, цилиндр ауданын табу керек.

```
#функцияны сипаттау
def cylinder():
    #бастапқы айнымалылар
    r = float(input())
    h = float(input())
    # цилиндрдің бүйір бетінің ауданы:
    side = 2 * 3.14 * r * h
    # цилиндрдің бір бөлігінің ауданы:
    circle = 3.14 * r * 2
    # цилиндрдің ауданы:
    full = side + 2 * circle
        return full
            square = cylinder()
            print(square)
```

Программа нәтижесі:

```
3
7
188.4
```

Программада `full` жергілікті айнымалысының мәні функциядан негізгі программа бөлігіне қайтарылады. Бұл жағдайда ол – цилиндр ауданын есептегенде алынған мән.

Негізгі программа бөлігінде осы мән `square` ауқымды айнымалысына меншіктеледі. `square = cylinder()` өрнегі былай орындалады:

`cylinder()` *функциясы шақырылады.*
Одан мән қайтарылады.
Бұл мән `square` *айнымалысына меншіктеледі.*

Айнымалы нәтижесін меншіктемей, бірден экранға шығаруға болады:

```
...
print(cylinder())
```

Мұнда, `cylinder()` функциясынан алынған сан `print()` функциясына беріледі. Егер программада айнымалының алынған мәндері меншіктелмей, жай ғана `cylinder()` функциясы жазылса, онда бұл деректер жоғалады. Бірақ синтаксистік қате болмайды.

2-мысал. Фибоначчи сандарын есептеу мысалында функцияны шақыруды пайдаланып, программа құру.

```
def fib(n):
    a, b = 0, 1
    while a < n:
        print(a, end = ' ')
        a, b = b, a + b
    print()
    fib (400)
```

`a, b = 0, 1` қысқартылуы мынаны білдіреді:

```
a = 0
b  = 1
```

`a, b = b, a + b` қатары:

```
a = b
b = a + b
```

Қатар бойынша кодты қарастырайық:

1

`def fib(n)` – параметрлері жақшаға алынған `fib` функциясын анықтау. *n* параметрі үшін есептеу үшін мән береміз. Бұл сан функцияға аргумент ретінде беріледі. Қос нүктеден кейін Python интерпретаторында енгізілетін сандар қадам арқылы шығады. Бұдан шығатыны – деректердің функцияға қатысы бар екені.

2

`a, b = 0, 1`

айнымалыларды мәндеріне сәйкес баптаймыз: `a = 0`

`b = 1`

3

`while a < n:`

`while` **циклдік операторы** – `a < n` қанағаттандырылғанша орындалатын болады. Мұнда да, қос нүктеден кейін тек циклге

ғана қатысты жаңа блок ашылады. Бұл блок 8 бос орыннан кейін жазылады.

4

```
print(a, end = ' ')
```

a айнымалысының деректерін жауапқа шығарады және әрбір цикл нәтижесінен кейін бос орын жазылады.

5

```
a, b = b, a + b
```

Айнымалыларды сәйкес мәндеріне меншіктейміз: `a = b`
`b = a + b`

бастапқы берілгендерді өзгерту және Фибоначчи сандарын есептеу үшін **6**-шы әрекетті орындаймыз.

6

```
print()
```

Назар аударсақ, `print()` циклден кейін шығарылуы керек. Бұл `while` цикл денесіне **емес,** `fib` функция денесіне қатысты. Бірақ екінші бос `print()` не үшін керек? Мына жағдайда бос қатарды тасымалдайды. Бұл функцияның қызметі жаңа бос жолды экранға шығарады. Программаны тексеру үшін функцияны шақырып, параметрін беру керек.

Функцияны шақырып, аргумент мәніне 40 санын береміз. Нәтижесінде 40-қа дейінгі Фибоначчи сандарын алуымыз керек:

Python интерпретаторында жазамыз:

```
fib(40)
```

Нәтиже:

```
0 1 1 2 3 5 8 13 21 34
```

`fib()` функциясын басқа параметр беру арқылы тағы да шақыруымызға болады. Мысалы,

```
fib(400)
```

Нәтиже:

```
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

осындай жолмен функцияны бірнеше мәрте шақыруымызға болады.

1-тапсырма. Функцияны пайдаланып, сандардың дәрежесін есептейтін программа құрыңдар. Кіріс параметрлері: (сан және дәрежесі).

2-тапсырма. Сан дәрежесін есептейтін функцияны жазыңдар.

3-тапсырма. Натурал N санның факториалын есептейтін программа құрыңдар.

4-тапсырма. n натурал сан берілген. Олардың қосындысын есептеңдер:

$$1 - \frac{1}{2!} + \frac{1}{3!} - \frac{1}{4!} + \frac{1}{5!} - \ldots (-1)^{n+1} \frac{1}{n!}$$

5-тапсырма. Өз цифрларының факториалдарына тең болатын барлық үш орынды санды табыңдар.

6-тапсырма. Өз координаттарымен берілген екі үшбұрыш берілген. Герон формуласы арқылы үшбұрыш ауданын есептеңдер және қай үшбұрыштың ауданы үлкен екенін анықтаңдар.

§ 23. Жолдармен жұмыс жасау

Естеріңе түсіріңдер:

- *функция дегеніміз не?*
- *процедуралар мен функциялар қалай қолданылады?*

Меңгерілетін білім:

- *жолдар, символ туралы түсінік;*
- *жолдарға қолданылатын амалдар.*

Терминдер:

- жол;
- символ.

XX ғасырдың ортасында алғашқы компьютерлер күрделі математикалық есептеулерді орындауда басты рөл атқарса, қазіргі уақытта олардың басты жұмысы мәтіндік (символдық) ақпараттарды өңдеу болып табылады.

Символдық жолдар – тізбектей орналасқан символдар тізбегі.

Біз жолдарды қарапайым деректер түрі ретінде бүтін сандарды нақты сандармен бірге қарастырдық және жолдың бір немесе қос тырнақшаға алынған таңбалар тізбегі екенін білеміз.

Python тілінде нысаны бір символ болатын символдық деректер типі жоқ. Дегенмен программалау тілі жолдарды ұзындығы бір немесе бірнеше символдан тұратын нысандары ретінде қарастыруға мүмкіндік береді. Мұнымен қоса, жолдардың тізімнен айырмашылығы – деректер құрылымына жатпайтынында. Шамасы, деректер құрылымдары қарапайым деректер түрлерінен тұрады, ал Python тілінде жолдар үшін қарапайым (символдық) тип жоқ.

Көптеген программалау тілдерінде символдық жолдармен жұмыс жасау үшін арнайы айнымалылар типі бар: символдар, символдық массивтер және символдық жолдар (массивтерден айырмашылығы: тұтас бір нысан ретінде қарастырылады). Python программалау тілінде символдық өлшемдерді өңдеуде қолданылатын негізгі деректер типі – бұл символдық жолдар, **string** типі.

Жолға мән жазу үшін меншіктеу операторы пайдаланылады.

```
s = "Гүлден сабақ оқып отыр"
```

Жолдар қос тырнақшаға немесе бір дәйекшеге алынып жазылады. Егер жол дәйекшемен шектелсе, оның ішінде дәйекше болуы да, болмауы да мүмкін.

Жолды пернетақтадан енгізу үшін `input` функциясы пайдаланылады:

```
s = input("Атын енгіз:")
print(s)
```

Жолдың ұзындығы `len` (ағылш. *length* – ұзындық) функциясы арқылы анықталады. Келесі мысалда *n* айнымалысы `s` жолдың ұзындығын анықтайды:

```
n = len(s)
```

Жолдан жеке символды бөліп алу үшін массив элементімен жұмыс жасағандай, тік жақшаға символ нөмірі жазылады. Мысалы, `s` жолының индексі 5 символын экранға шығару төмендегідей (бұл жағдайда жол саны 6-дан кем болмауы керек):

```
print(s[5])
```

Теріс индекс есептеудің жол соңынан басталатынын білдіреді. Мысалы, `s[-1]` **символы** `s[len(s)-1]`-ді білдіреді, жолдың соңғы символы.

Қазіргі программалау тілдерімен салыстырғанда, Python программалау тілінде символдық жолды өзгертуге болмайды. Жол – өзгермейтін нысан.

Бір жағынан алғанда, жолдар тізімдер секілді – реттелген элементтер тізбегінен тұрады. Соған сәйкес, одан символдар мен жеке бөліктерді алуға болады.

```
>>> s = "Hello, World!"
>>> s[0]
'H'
>>> s[7:]
'World!'
>>> s[::2]
'Hlo ol!'
```

Соңғы жағдайда көріп тұрғанымыздай, бөліп алу қадамы 2-ге тең, яғни әрбір 2-ші тұрған символ бөлініп алынады.

Ескерту. Тізімдерден бөліктерді қадаммен бөліп алуға болады.

Python программалау тіліндегі жолдардың тізімдерден маңызды айырмашылығы – өзгермейтіндігі. Қандай да бір жеке символды не бөлікті қайтадан жазуға болмайды:

```
>>> s[-1] = '.'
Traceback (most recent call last):
File "< stdin >", line 1, in < module >
TypeError: 'str' object does not support item assignment
```

Интерпретатор хабарлауынша, бұл нысан типі `str` емес элементтерді қосуды қолдамайды.

Егер жолды өзгерту керек болса, ескі бөліктерден жаңа жол құру керек:

```
>>> s = s[0:-1] + '.'
>>> s
'Hello, World.'
```

Бұл мысалда бөлік бастапқы жолдан алынып, екінші бір басқа жолға қосылады. S айнымалысына меншіктелген жаңа жол алынады. Оның бұрынғы мәні жойылады.

Енгізілген жолдан жаңа жол құруға болады. Ол үшін керек өзгерістерді енгізу керек. Пернетақтадан енгізілген жолдағы "а" әрпін "б" әрпіне ауыстыратын программа жазайық.

```
s = input("жолды енгіз:")
s1 = ""
for c in s:
if c = = "а":
c = "б"
s1 = s1 + c
print (s1 )
```

Мұнда `for c in s` циклінде `s` жолына кіретін барлық символдар орналасады. Әрқайсысы кезекпен с айнымалысына жазылады. **Сосын осы айнымалының мәнін** тексереміз: егер мән "а" әрпімен сәйкес келсе, онда оны "б" әрпіне ауыстырамыз, қосу операторы арқылы `s1` жаңа жолына жазамыз.

Жолдарға қолданылатын амалдар

- Екі жол үшін қосу амалы **(конкатенация)** және жолды санға көбейту амалы бар:

```
a = "рақ"
b = "мет"
print(a +b) # рақмет
a = "қар"
print (a*4) # қарқарқарқар
```

Жолдар массивтерге ұқсас индекстеледі: (индекстеу 0-ден басталады):

```
a = "сәлем"
print (a[2]) # л
```

- Жол ұзындығы `len()` функциясымен анықталады:

```
a = "информатика"
print (len(a)) # 11
```

Бөліктер

Жолдан бөлік бөлу амалы – [*X*:*Y*]
X – бөлік басы индексі, ал *Y* – соңы.

```
tday = 'morning, afternoon, night'
tday[0:7] # 'morning'
```

Python-да бөліктердің қолданылуы:

```
s = 'spameggs' информатика
s[3:5] # 'ор'
s[2:-2] # 'форм'
s[-4:-2] # 'рм'
s[:6] # 'информ'
s[1:] # 'нформатика'
s[:] # 'информатика'
```

Unicode әртүрлі тілдегі мәтіндерде пайдаланылатын барлық символдарды енгізуге мүмкіндік береді. Бұрын біз белгілі

кодтық парақшадағы 256 символды пайдаланатын едік. Жолдың алдына u спецификаторын қою керек. Мұнда әрбір символ 2 байтты береді.

Python-да форматтау – жолдарды басқарудағы басты құрал. Оның бірнеше тәсілі бар – *шаблондарды пайдалану* және *стандартты*. Python-дағы жолдарды форматтауда стандартты оператор – % символы пайдаланылады. Процент белгісінің сол жағына жолды көрсетеміз, ал оң жағында мәні немесе мәндер тізімі жазылады:

```
>>> s = 'Hello %s' % 'word'
>>> s
'Hello word'
>>> s = 'one  %s %s' % ('two','three')
>>> s
'one  two three'
```

Санды жолға айналдыру үшін сандық спецификатор – %d немесе %f символдары пайдаланылады:

```
>>> s = 'one  %d %f' % (2,3.5)
>>> s
'one  2 3.500000'
```

Жолды форматтау типінің кестесі (*4-кесте*):

4-кесте. Жолды форматтау

Код	Мәні
s	Жолдық
c	Символдық
d	Ондық
i	Бүтін
u	Ондық (no longer unsigned)
o	Сегіздік
x	Он алтылық
X	Үлкен регистрдегі оналтылық
e	Floating-point exponent, кіші регистр

Код	Мәні
E	Floating-point exponent, үлкен регистр
f	Floating-point decimal
F	Floating-point decimal
g	Floating-point e немесе f
C	Floating-point E немесе F
%	Символдық %

1 Сұрақтарға жауап берейік

1. Символдық жолдар деген не?
2. Символдық жолға мәнді қалай береміз? Әртүрлі тәсілдерін қарастырыңдар.
3. Жол элементіне берілген нөмір арқылы қалай байланыс орнатуға болады?
4. Жол ұзындығының есептелу принципі қандай?
5. «+» операторы қандай мағына білдіреді?
6. Жолдарға қолданылатын қандай негізгі амалдарды білесіңдер?
7. Символдық түрде берілген санды сандық түрге қалай алмастыруға болады?

2 Ойланайық, талқылайық

1. Символдық жолдар не үшін қажет?
2. Неге жаңа мәнді берілген жол позициясында бірден жазуға болмайды? Бұл мәселені қалай шешеміз?
3. Неге жолды әрқашан сандық түрге ауыстыруға болмайды?

3 Талдап, салыстырайық

1. Символдық жолдарға қолданылатын амалдарды анықтаңдар.
2. Массивтер, тізімдер және жолдарды салыстырып, ұқсастықтарын анықтаңдар.
3. Жолдардың массивтермен қандай ұқсастықтары бар?
4. Символдық жолдардың тізімдерден айырмашылығы қандай?

4 Дәптерде орындайық

1. Символдық жолдарға қолданылатын амалдарды кесте түрінде дәптерге жазыңдар.
2. Берілген есептердің алгоритмі мен блок-сызбасын дәптерге түсіріңдер.

5 Компьютерде орындайық

1. Ұзындығы N жол берілген. Жолдың таңбаларын кері тәртіпте жауапқа шығаратын программа құрыңдар. (Циклді қолданбаңдар).
2. Берілген символдық жолдағы барлық бас әріппен және кіші әріппен жазылған "а" әрпін "б" әрпіне және керісінше ауыстыратын программа құрыңдар. 'абсАБС' жолын енгізгенде нәтиже 'басБАС' түрінде болуы керек.
3. Символдық жолды енгізіп, оның палиндром екенін тексеретін программа құрыңдар (палиндром екі бағытта да бірдей оқылатын сөздер, мысалы, *қазақ, 626*).
4. Пернетақтадан бос орын арқылы аты, жөні және тегін енгізіңдер. Жауапқа тегін толық, аты мен әкесінің атының бірінші әріптерін шығарыңдар. Мысалы, "Мұхтар Есенұлы Орманов" жауапқа "М.Е.Орманов" түрінде болуы керек.

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.

§ 24. Жолдарды өңдеу үшін пайдаланылатын процедуралар мен функциялар

Естеріңе түсіріңдер:

- *жолдар, символ туралы түсінік;*
- *жолдарға қолданылатын амалдар.*

Меңгерілетін білім:

- *жолдарды өңдеу функциялары;*
- *жолдарды өңдеу әдістері;*
- *әдіс түрлері.*

Терминдер:

- жол;
- символ;
- функция.

Жолдарға қолданылатын әдістер

Әдіс — нысанға қолданылатын, яғни жолдарға арналған функция.

Python программалау тілінде жолдармен жұмыс жасаудың көптеген әдістері бар. Оларды `dir(str)` командасын орындау арқылы көруімізге болады. Әдістер туралы жеке ақпаратын алу үшін `help(str.әдіс_аты)` командасын орындау керек. Солардың ішіндегі қызықтыларын қарастырайық.

`split()` және `join()` әдістері

`split()` әдісі жолды бос орындар арқылы бөлуге мүмкіндік береді. Нәтижесінде сөздер тізімі пайда болады. Егер пайдаланушы бір жолға бірнеше сөзді немесе сандарды кіргізсе, олардың әрқайсысы программада бөлек өңделуі керек, `split()` әдісінсіз бұл мүмкін емес.

```
>>> s = input()
red blue orange white
>>> s
'red blue orange white'
>>> sl = s.split()
>>> sl
['red', 'blue','orange','white']
>>> s
'red blue orange white'
```

`split()` әдісі арқылы қайтарылған тізімді s айнымалысына меншіктей аламыз, яғни `s = s.split()`.

`join()` жолдық әдісі кері әрекет жасайды. Бұл әдіс болғанымен, алдына «–»белгісі қойылады. Ал тізім дәйекшеге алынады:

```
>>> '-'.join(sl)
'red-blue-orange-white'
```

`find()` және `replace()` әдісі

Бұл жол әдістері ішкі жолдармен жұмыс жасайды. `find()` әдісі жолдан ішкі жолды іздейді және ішкі жолдан табылған бірінші элементтің индексін қайтарады. Егер ішкі жол табылмаса, –1-ді қайтарады.

```
>>> s
'red blue orange white'
>>> s.find('blue')
4
>>> s.find('green')
-1
```

`replace()` әдісі ішкі жолды басқасына ауыстырады:

```
>>> letters.replace('DA', 'NET')
'ABCNETCFNET'
```

`format()` әдісі

`format()` жол әдісі `print()` функциясы арқылы нәтижені экранға шығару кезінде қарастырылады:

```
>>> print("This is a {0}. It's {1}.".format("ball",
"red"))
This is a ball. It's red.
```

Жолдар `input()` стандартты енгізу функциясы арқылы енгізіледі. Есімізге түсірейік, екі жол үшін қосу (біріктіру) амалы бар.

Python-да кез келген басқа нысанды сәйкес жолдарға ауыстыруға болады. Бұл үшін параметр ретінде жолға айналдыратын нысанға `str()` функциясын шақырамыз (*5-кесте*).

Негізінде, Python-мен қарасақ, әрбір жолдар – `str` класының нысандары. Нысан бойынша басқа кластың нысанын алу үшін, шығару функциясын қолданамыз. Бұл функцияның аты нысан әкеліп жатқан класс атымен сәйкес келуі керек (бұл функция — осы класс нысандарының құрастырушысы). Мысалы: `int` — бүтін сандар класы. Жолдарды сандарға ауыстыру `int()` функциясы арқылы жүзеге асырылады.

```
s  = input()
print(len(s))
t  = input()
number  = int(t)
u  = str(number)
print(s * 3)
print(s  + ' '  + u)
```

5-кесте. Жолдарды өңдеудегі әдістер мен функциялар

Функция немесе әдіс	Сипаттамасы

`S = 'str'; S = "str";`
`S = '''str''';`
`S =  """str"""`
	Жолдың литералы

`S = "s\np\ta\nbbb"`	Экрандалған тізбек
`S = r"C:\temp\new"`	Форматталмаған жолдар
`S = b"byte"`	Байттық жолдар
`S1 + S2`	Конкатенация (жолдарды қосу)
`S1 * 3`	Жолдарды көбейту
`S[i]`	Индекс бойынша іздеу
`S[i:j:step]`	Бөліктен алу
`len(S)`	Жолдың ұзындығы
`S.replace` (шаблон, ауыстыру)	Шаблонды ауыстыру
`S.split` (символ)	Бөлгіш арқылы жолды бөлу
`S.isdigit()`	Жолда сандардың бар-жоғын тексеру
`S.isalpha()`	Жолдың әріптерден тұратынын тексеру

Функция немесе әдіс	Сипаттамасы
`ord (символ)`	Оның ASCII коды символы
`chr (сан)`	Символдағы ASCII коды
`S.lstrip([chars])`	Жол басындағы бос орындары өшіру
`S.rstrip([chars])`	Жол соңындағы бос орындарды өшіру
`S.strip([chars])`	Жол басындағы және соңындағы бос орындарды өшіру
`S.format(*args,**kwargs)`	Жолды форматтау
`str.isupper()`	Жолдағы символдардың барлығы жоғарғы регистрде орналасқанын тексереді
`str.islower()`	Жолдағы символдардың барлығы төменгі регистрде орналасқанын тексереді

Жолдарды параметр секілді процедура мен функциямен беруге болады, функция нәтижесі сияқты қайтаруға да болады.

1 Сұрақтарға жауап берейік

1. Әдіс деген не?
2. Символдарға қолданылатын қандай әдістерді білесіңдер?
3. Берілген нөмір арқылы жолдың элементіне қалай байланыс орнатуға болады?
4. Жол ұзындығын қандай функция сипаттайды?
5. Жолдарға қолданылатын қандай негізгі амалдарды білесіңдер?

2 Ойланайық, талқылайық

1. Жолдық әдістер не үшін қажет?
2. Функциялар мен процедураларды не үшін қолданамыз?

3 Талдап, салыстырайық

1. Жолдық әдістердің ерекшелігі неде?
2. Берілген әдістер мен функцияларды салыстырып, ұқсастықтарын анықтаңдар.

3. Функциялардың процедуралардан айырмашылығы қандай?
4. Жолдардың массивтермен қандай ұқсастықтары бар?

4 Дәптерде орындайық

1. Жолдарға қолданылатын функцияларды кесте түрінде дәптерге жазыңдар.
2. Берілген есептердің алгоритмі мен блок-сызбасын дәптерде орындаңдар.

5 Компьютерде орындайық

1. Жолдық әдіс `isdigit ()` жолдың тек сандардан тұратындығын тексереді. Екі бүтін санды енгізіп, олардың қосындысын есептейтін программа жазыңдар. Қате енгізу жағдайында программа қатемен аяқталмауы керек, санды сұрастыруды жалғастыруы керек. try-except ерекшелігін пайдалануға болмайды.
2. Кіші әріптер мен бас әріптерді қамтитын жолды енгізіңдер. Сол жолды көрсету қажет, оның ішінде кіші әріптерді бас әріппен және бас әріптерді кіші әріптермен ауыстырыңдар. Мысалы, бастапқы жол `"aB!cDEf"`, жаңа жол `"Ab!CdeF"`. Кодта жолдың немесе таңбаның регистрін тексеретін `for` циклі, `upper()` және `lower()` (төменгі регистрге түрлендіру) әдістерін, сондай-ақ `isupper()` және `islower ()` әдістерін пайдаланыңдар.

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.

§ 25. Практикум. Жолдарды өңдеу үшін процедуралар мен функцияларды пайдалану

1-мысал. Жолдан индекстері 3-ке еселі символдарды шығаратын программа құру.

Шешімі:

Есепті циклді пайдалану арқылы шығарамыз (есептеудің күрделі түрі):

```
s = 'процедура'
x = 3
l = len(s)//3
for i in range(l):
print(s[x:x + 1:3]) # ц у
x + = 3
```

Қарапайым тәсіл – бөлікке бөлу амалын пайдаланып шығаруға болады:

```
s = 'процедура'
print(s[1::3]) # ц у
```

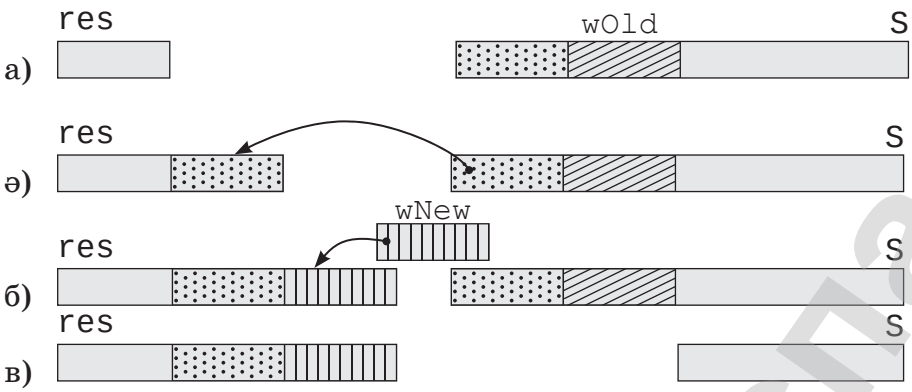
2-мысал. S жолындағы барлық `wOld` сөзін `wNew` сөзіне ауыстыратын процедура программасын құрастыру керек (Мұндағы `wOld` және `wNew` – айнымалы аттары).

Алдымен, есепті шешу алгоритмін жасайық. Ең алдымен, басымызға мынадай псевдокод келеді: `while wOld` **сөзі** `s` **жолында бар:**

жолдан `wOld` **сөзін өшіру**
оның орнына `wNew` **сөзін қою**

`wOld wNew` құрамына кіреді, мысалы, «12»-ні «`A12B`»-ге ауыстыру керек (шексіз циклге әкелетінін көрсетеміз).

Осындай жағдайға келмес үшін, `s` жолынан өңделген бөлікті өшіре отырып, нәтижені басқа бір `res` символдық жолға жазамыз. Мысалға, жолдың қалған бөлігінің бірнеше қадамында `s` жолында `wOld` сөзі кездесетін болсын (*а-сурет*).



Енді мына әрекеттерді орындау керек:

1) `wOld` сөзінің сол жағында тұрған `s` жолының бөлігін `res` жолының соңына жалғастыру керек (*ә-сурет*);
2) `wNew` сөзін `res` жолының соңына тіркеу керек (*б-сурет*);
3) `s` жолынан бастапқы бөлігін `wOld` сөзімен қоса өшіру (*в-сурет*).

Осы амалдардың барлығы `s` жолы бос болғанша орындала береді. Егер кезекті сөзді таппаса, `s` жолының қалған бөлігі нәтижеге шығып, цикл аяқталады.

Алгоритм жұмысының басында `res` жолына ешқандай символы жоқ `""` бос жол жазылады. Келесі кестеде `"12.12.12"` жолына арналған `"12"` деген сөзді `"A12B"` деген сөзбен ауыстырылуы тиіс ауыстыру алгоритмі берілген:

S жолының жұмысы	Res нәтижесі
`"12.12.12"`	`""`
`".12.12"`	`"A12B"`
`".12"`	`"A12B.A12B"`
`""`	`"A12B.A12B. A12B"`

Енді Python тілінде функциясын жазсақ болады. Оның параметрі – бастапқы жол `s`, үлгі-жол `wOld` және ауысу-жолы `wNew`:

```
def replaceAll (s, wOld, wNew):
lenOld = len(wOld)
res = ""
```

```
while len(s) > 0:
p = s.find (wOld)
if p < 0:
return res + s
if p > 0:
res = res + s[:p]
res = res + wNew
if p + lenOld > = len(s):
s = ""
else:
s = s[p + lenOld:]
return res
```

p айнымалысы – wOld сөзінде бірінші табылған бірінші символдың нөмірі, lenOld айнымалысына осы сөздің ұзындығы жазылады. Егер сөзді іздеуден кейін p мәні 0-ден кіші болса, циклден шығу орындалады:

```
if p < 0: res = res + s; return
```

Егер p > 0 болса, онда үлгі-сөздің сол жағында белгілі бір символдар болады, оларды res: if p >0: res = res + s[:p] жолына тіркеу керек.

p + lenOld > = len(s) шарты «үлгі сөздің соңында тұр» дегенді білдіреді. S жолының қалдығы – бос жол. Программа соңында нәтиже бастапқы s жолына жазылады.

Осы функцияны пайдалануға мысал келтірейік:

```
s = "12.12.12"
s = replaceAll (s, "12", "A12B")
print (s)
```

Ішкі жолды келесі бір басқа жолға ауыстыру амалы өте жиі қолданылады. Python-да осы амалды орындайтын кіріктірілген функция бар. Ол айнымалылардың жолдық типі (str) үшін әдіс ретінде жарияланған және нүктелік жазба арқылы шақырылады:

```
s = "12.12.12"
s = s.replace("12", "A12B")
print (s)
```

1-тапсырма.

Жолдан көрсетілген бөліктерді алыңдар:

- алғашқы 8 символды;
- жолдың ортасынан 4 символды;
- жолдың соңынан 5 символды.

2-тапсырма.

Ұзындығы *N* жол берілген. Жолдағы символдарды кері ретпен экранға шығарыңдар (циклді қолданбау керек).

3-тапсырма.

Ұзындығы *N* жол берілген. (*N* – жұп сан). Индексі жұп символдарды өсу ретімен экранға шығарыңдар.

```
a2, a4, a6, ... an
```

Шартты операторды пайдаланбау керек.

4-тапсырма.

Ұзындығы *N* жол берілген. Алдымен, жұп нөмірлі символдарды (нөмірдің өсу ретімен), сосын тақ нөмірлі сандарды экранға шығарыңдар:

```
a2, a4, a6,...a1, a3, a5...
```

Шартты операторды пайдаланбау керек.

5-тапсырма.

Жол берілген. Тақ нөмірлі символдарды кему ретімен экранға шығаратын программа құрыңдар (0-дік символды қарастыру керек).

§ 26–27. Файлдармен жұмыс жасау

Естеріңе түсіріңдер:

- *жолдарды өңдеу функциялары;*
- *жолдарды өңдеу әдістері;*
- *әдіс түрлері.*

Меңгерілетін білім:

- *файл түрлері;*
- *мәтіндік файл;*
- *файлдармен жұмыс.*

Термин:

- файл.

Программалауда «файл» сөзінің әртүрлі мағынасы кездеседі. Біріншіден, бұл операциялық жүйе – «атауы берілген дискінің бөлігі». Екіншіден, деректердің жүйелі түрде енетін абстрактілі құрылымы, үшіншіден, нақты программалау тілінде осы деректер құрылымын іске асыратын файлдық типті айнымалылар.

Сонымен, **файл** – компьютердің тұрақты жадысында жазылған символдар тізбегі.

Ағылшын тілінде «*file*» сөзі кез келген файлдың ішкі құрылымын жақсы көрсететін «тізбек» деген мағынаны береді. Файл – бұл белгілі көпмүшелікпен байланысқан символдар тізбегі: файлдардың символдары өз еркімен бір орыннан екінші орынға ауыса алмайды.

Файлдардың «дербестігі» қандай да бір программаның жұмысына тәуелді болмайды. Тіпті компьютер өшірулі болатын болса, файлдар қатқыл дискіде сақталады.

Файлдар өзіне кодталатындардың барлығын сақтай алады:

- программаның шығатын мәтіндерін немесе кіретін деректерін;
- программаны орындайтын машиналық кодтарды (ойындар, вирустар, оқытушы және сервистік программалар, т.б.);
- қандай да бір әрекет туралы ақпаратты;
- әртүрлі құжаттарды, сонымен қатар интернет-беттерді;
- суреттерді (фотосуреттер, видео);
- музыканы.

Программалауда қолдану саласы бойынша:

- қажетті файлдар, егер енгізілетін деректердің көлемі қолмен енгізуге болатын болса;
- егер бірнеше рет аз ғана өзгеріспен немесе ешқандай өзгеріссіз бір ғана деректі енгізу керек болса (мысалы, программаны жөндеуде).

- әртүрлі енгізілетін деректерді енгізу барысында алынған программа жұмысының нәтижесі туралы ақпаратты сақтау үшін файлдар қажет (яғни, программадағы қателіктерді іздеуде).

Мысалы, егер біздің программамызда екі немесе үш санды (бес – көбірек болады) алу керек болса немесе он символдан тұратын жолды алу үшін, осындай деректерді пернетақтадан қолмен енгізе аламыз. Айталық, егер бізге 10x10 массивін енгізу керек болса, онда қолмен енгізуде қателік саны бірнешеге көбеюі мүмкін. Енді осы болатын қателікті жою керек: қажет болғанда өте оңай өңделетін деректерді файлға жазу керек. Сонымен қатар бір рет құрылған файлды бірнеше рет қолдануға болады (маңызды емес өзгерістер болуы мүмкін).

Жалпы файлдар және олармен жұмыс істеу екі типке бөлінеді:

- ұзындығы белгісіз мәтіндік файлдар;
- екілік (бинарлы) файлдар (суреттер, дыбыстар, бейнефильмдер сияқты деректердің кодтарын сақтайды).

Файлмен жұмыс істеу кезеңдері:

1) файлды ашу;
2) файлмен жұмыс;
3) файлды жабу.

Файлды ашу. Open( ) әдісі

Бір нәрсені оқу және файлға жазу алдында оны ашып алу керек. Файлды ашу үшін кіріктірілген open ( ) функциясы қолданылады. Шақыру кезінде, бұл функция болашақта жұмыс істеуге болатын файл типті нысанды жасайды.

Python-да файлды екі параметрі бар open( ) функциясымен ашуға болады:

- *файл аты* (файл жолы);
- *файлды ашу режимі:*
  - "*r*" – оқу үшін ашу,
  - "*w*" – жазбаға ашу (файл бар болса, оның мазмұны жойылады),
  - "*a*" – қосу үшін ашу.

Open( ) функциясының жазылу синтаксисі:

```
Fin = open ("input.txt")
Fout = open("output.txt", "w")
```

```
#файлдармен жұмыс

Fout.close ()
Fin.close ()
```

Мәтіндік файлдармен жұмыс

Read () әдісі ашық файлдан жолды оқиды.
Read () әдісінің синтаксисі:

```
my_file.read([count])
```

Қосымша параметр count – ашық файлдан оқылатын байт саны. Бұл әдіс ақпаратты файлдың басынан бастап оқиды, ал егер count параметрі көрсетілмеген жағдайда, файлдың соңына дейін оқиды.

Мысалы, some.txt файлын оқимыз:

```
my_file = open("some.txt")
my_string = my_file.read()
print("Оқылды:")
print(my_string)
my_file.close()
```

Файлдан оқу екі жолмен жүзеге асырылады:
readline() әдісі арқылы жолма-жол оқу:
Файл input.txt:

```
str1 = Fin.readline () # str1 =1
str2 = Fin.readline() # str2 =2
```

read () әдісі ақпаратты файлдың соңына дейін оқиды:
файл input.txt:

```
str = Fin.read ()
""
str = 1
2
3
""
```

`Write()` әдісі жолды файлға жазу үшін қолданылады:

```
Fout = open("D:/out.txt","w")
Fout.write("hello")
```

Файлға жазуды белгілі бір **шығыс үлгісін** пайдалану арқылы жүзеге асыруға болады. Мысалы:

```
Fout.write("{:d} + {:d} = {:d}\n".format(x, y, x + y))
```

Бұл жағдайда {:d} үлгілерінің орнына `format` (алдымен x, содан кейін y, одан кейін x + y) әдісінің параметрлерінің мәндері жуықталған түрде қойылады.

Файлды жабу. `Close()` әдісі

`Close()` файлдық нысан әдісі, файлды автоматты түрде жабады, бұл ретте кез келген сақталмаған ақпарат жоғалады. Осыдан кейін файлмен жұмыс істей (оқу, жазу) алмаймыз.

Python программалау тілінде басқа файлға бекітілген файлдық нысан болса, файлды автоматты түрде жабады. Дегенмен файлды `close()` командасы арқылы қолдан жабу жақсы тәжірибе болады (*6-кесте*).

```
my_file = open("some.txt")
print("Файлдың атауы:", my_file.name)
print("Файл жабылды:", my_file.closed)
my_file.close()
print("Енді жабылды:", my_file.closed)
```

***6-кесте.** Файлды жабу әдісі*

Атауы	Қызметі
`file.closed`	Егер файл жабық болса, `True` мәнін қайтарады.
`file.mode`	Ашылған файлға кіру режимін қайтарады.
`file.name`	Файл атын қайтарады.
`file.softspace`	Файлдың мазмұнын көрсету кезінде бос орынды бөлек қосу қажет болса, `False` мәнін қайтарады.

Python программасында файлға кіру режимдерінің тізімі (*7-кесте*).

***7-кесте.** Файлға кіру режимдерінің тізімі*

Режим	Қызметі
r	Файлды оқу үшін ашады. Көрсеткіш файл басында тұрады
rb	Екілік форматта оқу үшін файл ашады
r +	Оқу және жазу үшін файл ашылады
rb +	Екілік форматта оқу және жазу үшін файл ашылады
w	Тек жазу үшін файл ашылады
wb	Екілік форматтағы жазу үшін файл ашылады
w +	Оқу және жазу үшін файл ашылады. Көрсеткіш файл басында тұрады. Файл_аты атты файл құрады.

1 Сұрақтарға жауап берейік

1. Файл деген не?
2. Бірдей файл айнымалысын бірнеше файлмен жұмыс істеу үшін пайдалануға бола ма? Мысал келтіріңдер.
3. «Деректерге тізбектей қол жеткізу» жағдайлары қай кезде орын алады?
4. Файлдық айнымалылар деген не?

2 Ойланайық, талқылайық

1. Басынан бастап файлдан деректерді оқуды қалай бастауға болады?
2. Файлдағы деректердің аяқталғанын қалай анықтауға болады?
3. Қандай жағдайда бір уақытта бірнеше файлды ашу керек?
4. Неліктен программаның ашық файлы ережеге сай бұғатталған кезде басқа программалар оған қол жеткізе алмайды?
5. Неліктен программа жабық болғанда автоматты түрде жабылады, бірақ файлдарды қолмен жабу ұсынылады? Бұл қандай жағдайларда маңызды болуы мүмкін?
6. Неге файлдармен жұмыс істейтін кезде, олар оның атауын емес, файл айнымалысын қолданады?

3 Талдап, салыстырайық

1. Файлдардың функциядан айырмашылығын анықтаңдар.
2. Мәтін мен екілік файлдар арасындағы ішкі мазмұнмен қандай айырмашылық бар? Мәтіндік файл екілік файлдың ерекше жағдайы деп айтуға бола ма?

4 Дәптерде орындайық

Кестені дәптерлеріңе толтырыңдар.

Атауы	Қызметі
`Close ()` **әдісі**	
`Readline ()` **әдісі**	
`Read ()` **әдісі**	
`Open ()` **әдісі**	

5 Компьютерде орындайық

1. Файлға жазылған оң жұп сандардың үлкені мен кішісін табатын, нәтижесін басқа файлға шығаратын программа құрыңдар. ***Ескерту***. Мұндай сандар жоқ болуы да мүмкін.
2. Файл бағанына бүтін сандар жазылған, соңғы саннан бастап өсу ретімен жазатын, нәтижесін басқа файлға шығаратын программа құрыңдар.

6 Ой бөлісейік

1. Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.
2. Файлдармен жұмыс істеген кезде «сэндвич қағидасын» қалай қолданамыз?

§ 28. Практикум. Ақпаратты оқу және жазу үшін файлдарды пайдалану

1-мысал. Жаңа some мәтіндік файлын ашу. Оның жұмыс жасау режимін және ашық екендігін тексеретін программа құру.

Программасы:

```
my_file = open("some.txt", "w")
print("Файл аты:", my_file.name)
print("Файл жабық:", my_file.closed)
print("Файлдың ашық режимі:", my_file.mode)
print("Бос орын:", my_file.softspace)
```

Нәтижесі:

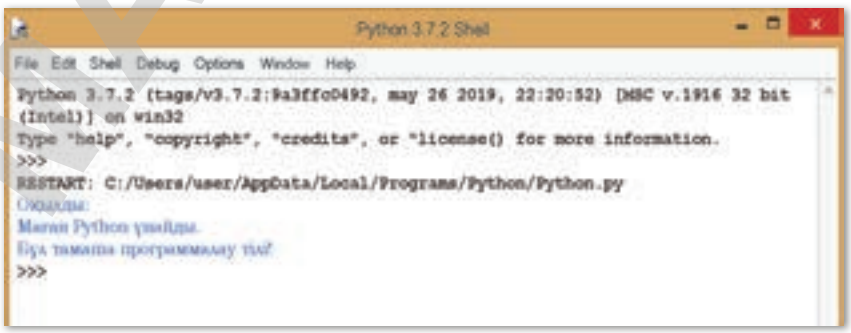
Файл аты: some.txt
Файл жабық: False
Файлдың ашық режимі: w

2-мысал. Файлға жазба жазу, оны оқып, экранға шығару.

Программасы:

```
my_file = open("some.txt", "w")
my_file.write("Маған Python ұнайды!\n Бұл – тамаша
программалау тілі!")
my_file.close()
my_file = open("some.txt")
my_string = my_file.read()
print("Оқылды:")
print(my_string)
my_file.close()
```

Нәтижесі:



1-тапсырма.

Файлда бүтін сандар жазылған. Ең көп және ең аз санды табу және нәтижесін басқа файлға жазу.

2-тапсырма.

Файлда бүтін сандар бағанға жазылған. Сандарды өсу реті бойынша сұрыптап және нәтижесін басқа файлға жазу.

3-тапсырма.

Файлға белгілі бір фирманың қызметкерлері туралы деректер былайша жазылған:

```
Есжанов 45 есепші
```

Жасы 40-тан аз қызметкерлер туралы деректерді мәтіндік файлға жазу қажет.

4-тапсырма.

Файлға балабақша балалары туралы деректер жазылған:

```
Абай Қасымұлы 5 жас
```

Жасы үлкен және жасы кіші балалар жайлы деректі мәтіндік файлға жазу қажет.

```
age = int (s.split()[1])
if age < 5:
Fout.write (s)
```

5-тапсырма.

Файлға бағаң бойынша енгізілген сандардың арифметикалық ортасын тауып, нәтижені басқа файлға шығаратын программа құрыңдар.

6-тапсырма.

Файлдан мәтінді оқып, қанша сөз барын есептейтін программа құрыңдар.

§ 29–30. Сұрыптау әдістері

Естеріңе түсіріңдер:

- *файл туралы түсінік;*
- *файл түрлері;*
- *мәтіндік файл;*
- *файлдармен жұмыс.*

Меңгерілетін білім:

- *«сұрыптау» түсінігі;*
- *жылдам сұрыптау түрі;*
- *көпіршікті сұрыптау.*

Терминдер:

- сұрыптау;
- көпіршікті сұрыптау.

Көпіршікті сұрыптау – массивтер мен тізімдерді тізбектей салыстыру және егер алдыңғы элемент кейінгі тұрған элементтен үлкен болса, көрші элементтерін ауыстыратын сұрыптау әдісі. Бұл алгоритмді орындау барысында үлкен мәнді элементтер тізімнің соңында орналасады, ал мәні кіші элементтер біртіндеп тізімнің басына қарай жылжып орналасады. Бейнелеп айтқанда, ауыр элементтер түбіне құлайды, ал жеңілдері көпіршіктер сияқты баяу ауаға ұшады.

Көпіршікті сұрыптау кезінде сыртқы циклдің итерацияларының саны –1-мен анықталады, өйткені екінші элемент орнына түскен кезде, бірінші біреуі бірден минималды және өз орнында орналасады.

Ішкі циклдегі итерация саны сыртқы тізбектің иерархиялық нөміріне байланысты, себебі тізімнің соңы қазірдің өзінде сұрыпталған және бұл элементтерді қайтадан сұрыптаудың қажеті жоқ.

Мысалы, [6, 12, 4, 3, 8] тізімі берілсін.

Сыртқы циклдің алғашқы итерациясы кезінде 12 саны соңына қарай жылжиды. Бұл үшін ішкі циклдегі 4 салыстыруды талап етеді:

- 6 > 12? Жоқ
- 12 > 4? Иә. Орын ауыстырамыз
- 12 > 3? Иә. Орын ауыстырамыз
- 12 > 8? Иә. Орын ауыстырамыз

Нәтижеде: [6, 4, 3, 8, 12]

Сыртқы циклдің екінші итерациясы кезінде 8 саны алдыңғы орынға жылжиды. Бұл 3 салыстыруды қажет етеді:

- 6 > 4? Иә. Орын ауыстырамыз
- 6 > 3? Иә. Орын ауыстырамыз
- 6 > 8? Жоқ

Нәтижеде: [4, 3, 6, 8, 12]

Сыртқы циклдің үшінші итерациясында соңғы екі элемент алынып тасталады. Ішкі циклдің итерациясының саны екі:

- 4 > 3? Иә. Орын ауыстырамыз
- 4 > 6? Жоқ

Нәтижеде: [3, 4, 6, 8, 12]

Сыртқы циклдің төртінші итерациясында тек алғашқы екі элементті ғана салыстыру ғана қалды, сондықтан ішкі циклдің итерация саны бірге тең болады:

- 3 > 4? Жоқ

Нәтижеде: [3, 4, 6, 8, 12]

Бұл қарапайым түрде қатар орналасқан 2 элементті салыстыруға негізделген. Егер де массив элементтері тік бағытта, баған құрай орналасса, онда оларды шыны ыдыстағы көпіршіктер ретінде елестетуге болады. Бұл жағдайда әрбір көпіршік өзінің салмағына қарай тиісті биіктікке дейін көтеріліп, орналасады. Көпіршікті сұрыптау атауына осыған қатысты ие болған. Сұрыптаудың мәні:

1. Алғашқы 2 элемент салыстырылады. Егер 1-ші элемент 2-ші элементтен кіші болса, онда олардың орындары ауыстырылады.
2. 2-ші мен 3-ші элемент, 3-ші мен 4-ші элемент және т.с.с. салыстырылып, қажет болған жағдайда олардың орындары алмастырылады. Нәтижеде ең кіші элемент бастапқы орынға ауыстырылады.

Массив элементтерін толық іріктеп орналастыруда осы әрекет $(n - 1)$ рет орындалады. Мұндағы n – массив элементтерінің саны.

Әр қайталануда алмастырудың орындалуын сипаттайтын ***айнымалы*** мәнін енгізіп, процестің аяқталғанын (аяқталмағанын) бақылап отыруға болады.

Кейде бұл әдіс – ***Алмастыру арқылы сұрыптау әдісі*** деп те аталады. Бұл әдіс бойынша салыстырулар саны $n(n - 1)/2$ -ге тең.

Ескерту:

1. Егер қандай да бір қадамда ешқандай алмастыру орындалмаса, онда алгоритм жұмысы тоқтатылуы тиіс.
2. Ағымдық қадамда алмастыру жасалған массив индексінің ең кіші мәні есте сақталуы тиіс. Массивтің осы индексті элементіне дейінгі бастапқы элементтері сұрыпталып қойғандықтан, массивтің аталған индексті элементі мен көрші элементін салыстырудың қажеті жоқ.

3. Мәні кіші элементтер бір ғана алмастырудан кейін қажетті орынға қойылса, мәні үлкен элементтер тек алгоритм толық орындалғаннан кейін ғана қажетті орынға орналастырылуы мүмкін.

Кірістірілген функциялар

mas.reverse() – массив элементтерін кері тәртіпте қайта реттеудің стандартты әдісі;

mas2 = sorted (mas1) – массивтерді (тізімдерді) сұрыптау үшін бекітілген функция.

Массивті іздеу үшін `While` циклін қолдану түрі:

```
import random # кітапхананы қосу
from random import randint
n = 10; x = 5
   mas = [randint(1,10) for i in range(n)] #мас-
сивті инициализациялау
i = 0
   while i < n and mas[i] ! = x: #егер элемент тең
болмаса,
   i + = 1
if i < n:
   print ("mas[", i, "] = ", x, sep = "")
else:
print ("Табылмады!")
```

Массивті іздеу үшін `for` циклін қолдану түрі:

```
import random
from random import randint
n = 10;x = 5
mas = [randint(1,10) for i in range(n)]

for i in range (n):
   if mas[i] = = x:
      nomer = i
      break
if nomer > = 0:
   print ("mas[",nomer,"] =", x, sep ="")
```

```
else:
   print ("Табылмады!" )
```

Бұл жағдайда **nomer** айнымалысында табылған массив элементінің саны және мәні сақталады.

Бірақ Python тілінде **for** циклінің бірегей қасиеті – онда **else** блогы бар, ол **break** операторы циклде қолданылмаған кезде орындалады.

Сондықтан екінші іздеу әдісін қарастыратын болсақ, ол әлдеқайда қарапайым:

```
import random
from random import randint
n = 10;x = 5
mas = [randint(1,10) for i in range(n)]

nomer = -1
for i in range (n):
   if mas[i] = = x:
      print ("mas[", i, "] = ", x, sep = "")
      break

else:
   print ("Табылмады!")
```

1 Сұрақтарға жауап берейік

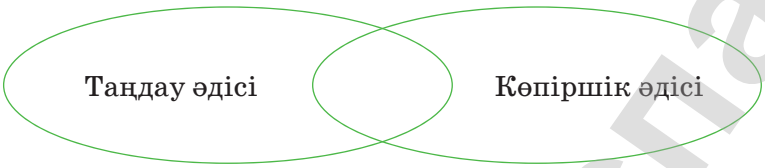
1. Сұрыптаудың мағынасы неде? Күнделікті өмірде нені сұрыптауға болады?
2. Көпіршікті сұрыптау әдісі қандай жағдайларда қолданылады?
3. Жылдам сұрыптау әдісінің негізгі идеясы неде?
4. Python-да массивтерді сұрыптауда қандай кіріктірілген функцияларды білесіңдер?

2 Ойланайық, талқылайық

1. Көпіршікті сұрыптау не үшін қажет?
2. Не себепті көпіршікті сұрыптауда жеңіл элементтер жоғарыда орналасады?
3. Берілген сұрыптау әдістеріне цикл не үшін қажет?

3 Талдап, салыстырайық

1. Жылдам сұрыптау мен көпіршікті сұрыптаудың айырмашылығы қандай?
2. Таңдау әдісі мен көпіршік әдісін Венн диаграммасы арқылы салыстырыңдар.



4 Дәптерде орындайық

1. Сұрыптаудың барлық түрлерінің атауын, дәптерге жазыңдар.
2. Көпіршікті сұрыптауда қолданылатын функцияларды жазыңдар.

5 Компьютерде орындайық

Жиынды сұрыптайтын программаны жазыңдар, содан кейін массивте бірнеше рет пайда болған сандардың ең үлкенін табыңдар. Кірістірілген функцияларды пайдаланбаңдар.

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Алған жаңа білімдеріңді күнделікті өмірде қандай жағдайда қолдануға болады? Мысал келтіріңдер.

§ 31–32. Практикум. Практикалық есептерді шешу үшін сұрыптау алгоритмдерін іске асыру

1-мысал. for циклі көмегімен көпіршікті сұрыптауға программа құру.

Шешімі:

```
from random import randint
   N = 10
a = [ ]
for i in range(N):
        a.append(randint(1, 99))
print(a)
      for i in range(N-1):
         for j in range(N-i-1):
            if a[j] > a[j + 1]:
               a[j], a[j + 1] = a[j + 1], a[j]
print(a)
```

Кодты орындау нәтижесі:

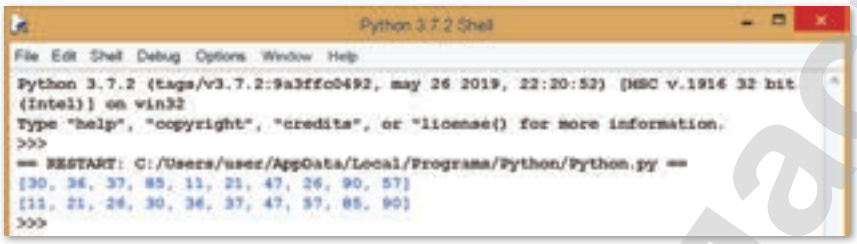
```
[63, 80, 62, 69, 71, 37, 12, 90, 19, 67]
[12, 19, 37, 62, 63, 67, 69, 71, 80, 90]
```

2-мысал. While циклінің көмегімен көпіршікті сұрыптауға программа құру.

Шешімі:

```
from random import randint
   N = 10
a = []
for i in range(N):
      a.append(randint(1, 99))
print(a)
i = 0
while i < N - 1:
   j = 0
   while j < N - 1 - i:
      if a[j] > a[j + 1]:
         a[j], a[j + 1] = a[j + 1], a[j]
      j + = 1
   i + = 1
print(a)
```

Нәтижесі:

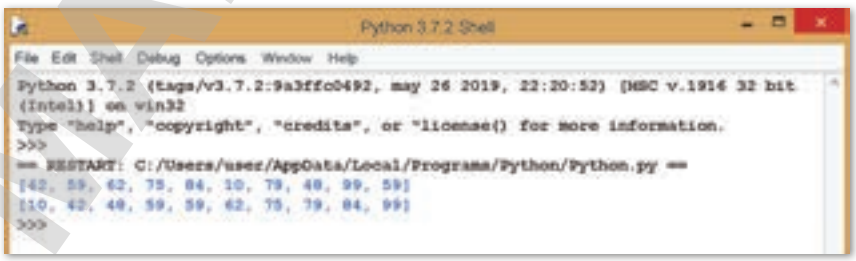


3-мысал. Python-да көпіршікті сұрыптау функциясын пайдаланып, сұрыптау программасын құру.

Шешімі:

```
from random import randint
def bubble(array):
   for i in range(N-1):
      for j in range(N-i-1):
         if array[j] > array[j + 1]:
            buff = array[j]
            array[j] = array[j + 1]
            array[j + 1] = buff
N = 10
a = []
for i in range(N):
   a.append(randint(1, 99))
print(a)
bubble(a)
print(a)
```

Нәтижесі:



4-мысал. Python-да массивті сұрыптауды көпіршік әдісі көмегімен жүзеге асырып, программасын құру.

Шешімі:

```
import random
from random import randint
n = 10
mas = [randint(1,10) for i in range(n)]
for i in range(n):
    print(mas[i], sep = "")
print(" ")
for i in range(n-1):
    for j in range(n-2, i-1 ,-1):
          if mas[j + 1] < mas[j]:
          mas[j], mas[j + 1] = mas[j + 1], mas[j]
for i in range(n):

    print(mas[i], sep ="")
```

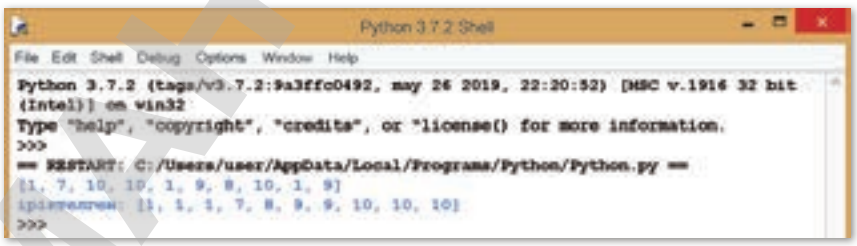
Нәтижесі:

5-мысал. Массивті жылдам сұрыптау программасын құру.

Шешімі:

```
import random
from random import randint
# процедура
def qSort (A, nStart, nEnd ):
   if nStart > = nEnd: return
         L = nStart; R = nEnd
         X = A[(L + R)//2]
         while L < = R:
            while A[L] < X: L + = 1 # бөлу
      while A[R] > X: R - = 1
      if L < = R:
            A[L], A[R] = A[R], A[L]
            L + = 1; R - = 1
         qSort (A, nStart, R) # рекурсиялық
шақыру
      qSort (A, L, nEnd )
N = 10
A = [randint(1,10) for i in range(N)]
print(A)
# процедурадан шақыру
qSort (A, 0, N-1)
print('іріктелген',A)
```

Нәтижесі:



1-тапсырма.

Бірінші нөмірлі массивтен бастап өсу реті бойынша массивті сұрыптайтын программа жазу керек (жылдам сұрыптау арқылы).

2-тапсырма.

Сұрыптаудың «тас әдісі» – «ең ауыр» элемент массивтің соңына дейін түсетін программаны жазу қажет.

3-тапсырма.

Массив берілген. Массивте 3-ке еселі сандар бар екенін сұрыптау арқылы есептеу керек.

4-тапсырма.

0...4 аралықтағы массивті кездейсоқ сандармен толтырыңдар және экранға X мәніне тең барлық элементтердің нөмірін шығарыңдар (пернетақтадан енгізіледі).

5-тапсырма.

Белгіленген массивті өзгертпестен, оның элементтерінің сандарын өсу ретімен экранға шығаратын программаны жазыңдар. Сандардың қосалқы массивін қолданыңдар.

6-тапсырма.

Тізімді сұрыптайтын және әртүрлі сандардың қосындысын табатын программаны жазыңдар. Кірістірілген функцияларды пайдаланбау керек.

7-тапсырма.

Массив берілген. Бірізді бірдей элементтердің қатарын серия деп атайық және серияның ұзындығы – элементтердің саны. Екі жаңа массивті құрастырыңдар, барлық сериялардың ұзындығын олардың біріне жазыңдар және осы сериялары екіншіге айналдыратын элементтердің мәндерін жазыңдар.

8-тапсырма.

Сыртқы циклдің келесі кезеңінде қайта қойылым болмаса, жұмысты тоқтататын көпіршікті әдіс нұсқасын жазыңдар. Кірістірілген функцияларды пайдаланбаңдар.

§ 33–34. Графтағы алгоритмдер

Естеріңе түсіріңдер:

- *сұрыптау деген не?*
- *жылдам сұрыптау түрлері қандай?*
- *көпіршікті сұрыптау деген не?*

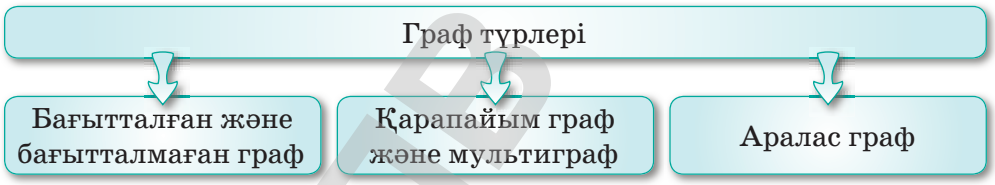
Меңгерілетін білім:

- *«граф» түсінігі;*
- *графтың түрлері;*
- *графтағы іздеу алгоритмдері;*
- *тереңнен іздеу;*
- *көлденеңінен іздеу.*

Графтар теориясы соңғы уақытта ғылым мен техниканың түрлі салаларында кеңінен қолданылады. Графтар теориясы алгоритмдеудің түрлі есептерін шешуге мүмкіндік беретін электронды есептеуіш машинаның пайда болуымен қарқынды дамыды.

Граф – бұл екі жиынның жиынтығы: нүктелер жиыны мен сол нүктелердің кейбірін жұптап қосатын сызықтар жиыны. Нүктелер жиыны *графтың төбелері (түйіндері)* деп аталады. Граф төбелерін қосатын сызықтар жиыны *графтың қабырғалары (доғалар)* деп аталады.

Графтың түрлері *3-сызбада* келтірілген.



3-сызба. *Граф түрлері*

Бағытталған граф – барлық қабырғаларының бағыты бар *граф*, яғни қабырғаларына бағыт берілген.

Бағытталмаған граф – барлық қабырғаларының бағыты жоқ *граф*, яғни қабырғаларына бағыт берілмеген.

Аралас граф – бағытталған қабырғадан да, бағытталмаған қабырғадан да тұратын *граф*.

Ілмек деп графтың өзіне өзін қосатын *қабырғаны* айтамыз. Егер екі төбені қосатын *қабырға* бар болса, онда ол төбелер *көршілес* деп аталады. Бірдей төбелер жұбын қосатын қабырғаларды *еселі* деп атайды.

Қарапайым граф – ілмегі де, еселі қабырғалары да жоқ *граф*.

Мультиграф – кез келген екі төбесі бір қабырғадан артық қабырғамен қосылған *граф*.

Граф маршруты дегеніміз – көршілес төбелерді қосатын сол төбелер мен қабырғалардың соңғы кезектелген тізбегі.

Егер бастапқы және соңғы төбелер әртүрлі болатын болса, онда маршрут **ашық** деп аталады, ал кері жағдайда маршрут **тұйықталған** деп аталады.

Егер маршрут төбелері әртүрлі болса, онда ол маршрут **тізбек** деп аталады. Егер ашық тізбекке кіретін төбелер әртүрлі болатын болса, онда ол тізбек **жол** деп аталады. Егер тұйықталған тізбекке кіретін төбелер (ақырғы төбеден басқа) әртүрлі болса, онда ол тізбек **цикл** деп атады.

Егер графта деп кез келген екі төбе үшін оларды қосатын жол бар болса, онда ол **байланысқан** граф деп аталады.

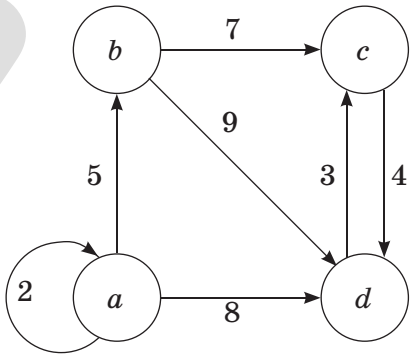
Төбе салмағы – сол төбеге сәйкес (құн, өткізу қабілеті және т.б.) қойылған сан (нақты, бүтін немесе бөлшек).

Қабырға салмағы (ұзындығы) – қабырғаға *ұзындық, өткізу қабілеті* және т.б. қатынаста берілетін сан немесе бірнеше сан.

Өлшенген граф – әрбір қабырғаға қандай да бір мән (қабырға салмағы) қойылған *граф*.

Компьютер жадында графты сақтауға арналған құрылымды таңдау **тиімді алгоритмдерді** жасауда белгілі бір мәнге ие. Графты **ұсынудың** бірнеше **әдісін** қарастырайық.

18-суреттегідей граф берілсін. Оның төбелер саны – n, ал қабырғалар саны m болсын. Әрбір *қабырға* мен әрбір *төбенің салмағы* бар – бүтін оң сан. Егер граф белгіленбеген болса, онда оның *салмағы 1-ге* тең деп есептеледі.



18-сурет. *Граф мысалы*

Қабырғалар тізімі – көршілес төбелер жұбынан құралған жиын. Оны сақтау үшін графтың бір қабырғасымен көршілес төбелер жұбының тізімінен тұратын *бірөлшемді массив* қолданылады. Графтағы түрлі алгоритмдерді жүзеге асыруда басқа әдістермен салыстырғанда қабырғалар тізімі тиімді (*8-кесте*).

8-кесте. Графтың қабырғалар тізімі

a	*a*	*a*	*b*	*b*	*c*	*d*
a	*b*	*d*	*c*	*d*	*d*	*c*
2	5	8	7	9	4	3

Көршілестік матрица – бұл граф төбелерінің көршілестілігі сипаттайтын *n* x *n* өлшемдегі екіөлшемді массив (*9-кесте*). Матрица элементтерінің мәні ретінде төбелерді қосатын қабырғалар саны меншіктеледі. Бұл әдіс берілген екі төбе бойынша қабырға салмағын немесе олардың көршілестігін анықтауда қолданылады.

9-кесте. Графтың көршілестік матрицасы

	a	*b*	*c*	*d*
a	2	5	0	8
b	0	0	7	9
c	0	0	0	4
d	0	0	3	0

Инцидиенттік матрица – графтың инцидиентті элементтері (қабырға мен төбе) арасындағы байланысты көрсететін *n* x *m* өлшемдегі екіөлшемді массив. Матрица бағаны *қабырғаларға*, ал жолы *төбелерге* сәйкес келеді. Матрицадағы нөлдік емес мәндер төбе мен қабырға арасындағы байланысты көрсетеді. Бұл әдіс қиын болғанымен, графтағы циклдерді табуды жеңілдетеді (*10-кесте*).

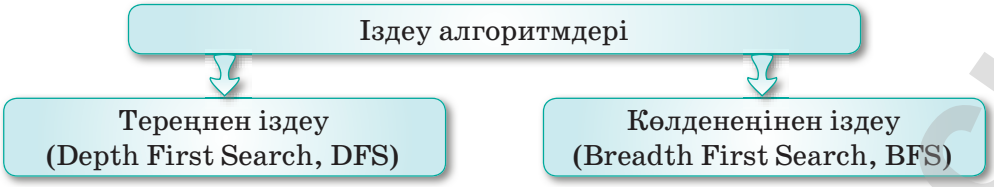
10-кесте. Графтың инцидиенттік матрицасы

	a	*b*	*c*	*d*
(*a*, *a*)	2	0	0	0
(*a*, *b*)	0	5	0	0
(*a*, *d*)	0	0	0	8
(*b*, *c*)	0	0	7	0
(*b*, *d*)	0	0	0	0
(*c*, *d*)	0	0	0	4
(*d*, *c*)	0	0	3	0

Графтағы алгоритмдер түрі өте көп. Ол алгоритмдер граф төбелерін кем дегенде бір рет қарап, граф төбелеріне жүйелі талдау жасайды. Сондықтан ең маңызды міндет – графтағы іздеудің жақсы әдістерін табу.

Графтағы іздеу – қандай да бір шартқа сәйкес келетін қабырға немесе төбені іздеп табу үшін графтың барлық төбелері мен қабырғаларын жүйелі түрде қарап шығу үрдісі.

Графтарды қолданатын көптеген есептерді шешуде граф қабырғалары мен төбелерін тұрақты іздеуді жүзеге асыратын тиімді әдістер керек. Стандартты және кең тараған әдіс түрлері *4-сызбада* көрсетілген.



4-сызба. *Іздеу алгоритмдерінің әдістері*

Бұл әдістер көп жағдайда бағытталған графтарда қолданылады, алайда қабырғалары *қосбағытты* деп саналатын бағытталмаған графтарда да қолдануға болады. Тереңнен және көлденеңінен іздеу алгоритмдері графты өңдеудің түрлі есептерін шешуге арналған, мысалы, байланысты тексеру, тұйықталған облысты анықтау, төбелер арасындағы қашықтықты табу және басқалар.

Тереңнен іздеу

Тереңнен іздеуде алдымен, бірінші төбе қарастырылады, әрі қарай граф қабырғасы арқылы *тұйықталған* бөлікке дейін барады. Егер іздеу барлық көршілес төбелерді қарап шыққан болса, граф төбесі *тұйықталған* деп саналады.

Іздеу тұйықталған бөлікке түскеннен кейін келген жолымен артқа қарай қаралмаған төбелері бар төбеге келеді, әрі қарай осы бағытпен іздеу жұмысын жалғастырады.

Үрдіс бастапқы алғашқы төбеге келген кезде және оған көршілес барлық төбелерді қарап шыққан кейін аяқталған болып саналады.

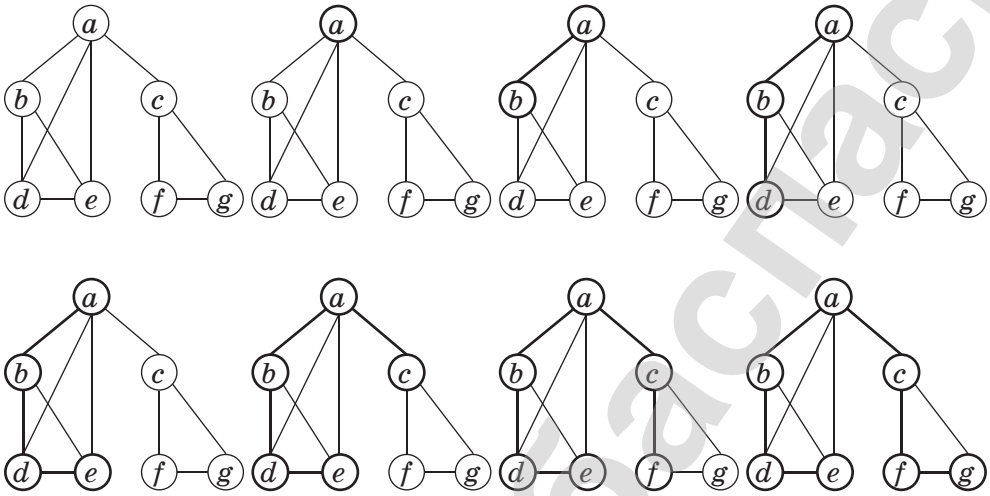
Осылайша, тереңнен іздеу алгоритмінің негізгі идеясы – төбеден шығатын қабырғалардың барлық мүмкін тармақталған жолдары бойынша алдымен бір тармағын, сосын келесі тармағына өту арқылы графты толық зерттеу.

Қадам бойынша тереңнен іздеу алгоритмі

1-қадам. Графтың барлық төбелеріне мәндер меншіктеледі. Бірінші төбені таңдап алып, оны қарастырылған деп белгілейміз.

2-қадам. Ең соңғы қарастырылған деп саналатын төбе бірінші қарастырылған төбенің көршілес төбесі болып табылады. Егер ондай төбе жоқ болса, онда алдыңғы қарастырылған төбе алынады.

3-қадам. Екінші қадамды барлық төбе қарастырылған деп белгіленгенше қайталаймыз (*19-сурет*).



19-сурет. *Тереңнен іздеу алгоритмі*

```
// Тереңнен іздеу алгоритмі функциясын сипатталуы.
#
# 2--0--6--7 1--9 5
# | | |
# 3--4 8
#
n = 10 # төбе саны
adj_list = [[2, 4, 6],
            [9],
            [0, 3],
            [2, 4],
            [0, 3],
            [],
            [0, 7,8],
            [6],
            [6],
            [1]]
s = 0
   visited = [False] * n # "төбе қаралды ма?"
   массиві
```

```
def dfs(v):
   visited[v] = True
   for w in adj_list[v]:
      if visited[w] == False: # қазіргі көрші
      төбе қаралды ма?
         dfs(w)
dfs(s)
print(visited.count(True))
```

Тереңнен іздеудің рекурсивті емес *алгоритмі* де қолданылады. Бұл жағдайда рекурсия *стекке* алмастырылады. Төбе қарастырылып болғаннан кейін ол стекке орналастырылады, ал егер оған көршілес төбелер жоқ болса, ол төбе қарастырылған деп белгіленеді.

Көлденеңінен іздеу

Көлденеңінен іздеуде бірінші төбені қарастырып алғаннан кейін, оған көршілес барлық төбелер қарастырылады. Әрі қарай бастапқы төбеден екі қабырға арақашықтықта жатқан барлық төбелер қарастырылады. Әрбір жаңа қадамда алдыңғы қадамнан бір бірлікке артық қадамдағы төбе қарастырылады. Төбелері қайталанбауы үшін қарастырылған төбелер *тізімін* жасау керек. Алгоритм жұмысына қажетті уақытша ақпараттарды сақтау үшін *кезек* қолданылады, ол – жаңа элементтер соңына қосылып, ескілерінің басынан өшірілетін элементтердің реттелген тізімі.

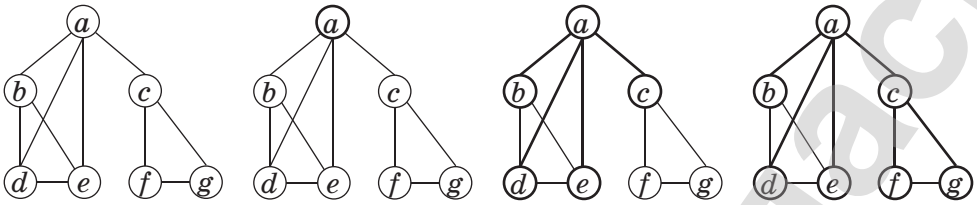
Көлденеңінен іздеудің басты идеясы – алдымен бастапқы төбемен көршілес барлық төбелер қарастырылуы. Бұл төбелер бастапқы төбеден бір бірлікке қашық болады. Әрі қарай екі бірлікке қашық төбелер қарастырылады, одан кейін үш бірлік және т.с.с. Осы жерде әрбір төбеге бастапқы төбеден қысқа маршрут жолының *ұзындығы* табылып отырады (*15-сурет*).

Қадам бойынша көлденеңінен іздеу алгоритмі

1-қадам. Барлық төбеге мәндер тағайындалады. Бірінші төбе қарастырылған деп белгіленіп, кезекке қойылады.

2-қадам. Кезектегі бірінші төбе қарастырылады. Оның барлық төбелері кезекке қойылады. Содан кейін ол төбе кезектен өшіріледі.

3-қадам. Екінші қадам *кезек* босағанға дейін қайталанады (*20-сурет*).



20-сурет. *Көлденеңінен іздеу алгоритмі*

// **Көлденеңінен іздеу алгоритмі функциясын сипатталуы.**

```
adj = [
#көршілес төбелер
      [1,3], # 0
      [0,3,4,5], # 1
      [4,5], # 2
      [0,1,5], # 3
      [1,2], # 4
      [1,2,3] # 5
]
level = [-1] * len(adj)
#төбе деңгейінің тізімдері
def bfs(s):
      global level
      level[s] = 0
# бастапқы төбе деңгейі
      stack = [s]
   # төбені кезекке қою
      while stack:
         v = stack.pop(0)
   # төбені алу
            for w in adj[v]:
# v төбесінен қарастыру
               if level[w] is -1:
# егер төбе қарастырылмаған болса
                  stack.append(w)
```

```
    # төбені кезекке қою
                level[w] = level[v] + 1
    # төбе деңгейін санаймыз
    for i in range(len(adj)):
          if level[i] is -1:
             bfs(i)
       # бірнеше байланыс компоненті болған жағдайда
print(level[2])
```

1 Сұрақтарға жауап берейік

1. Граф деген не?
2. Графтың қандай түрлері бар?
3. Граф маршруты күнделікті өмірде қайда қолданылады?
4. Қабырғалар тізімі қалай құрылады?
5. Графтың көршілестік матрица мен инцидиенттік матрица қалай құрылады?
6. Графтағы іздеу алгоритмдерінің қандай түрлері бар?
7. Тереңнен іздеу алгоритмі қалай орындалады?
8. Көлденеңінен іздеу алгоритмі қалай орындалады?

2 Ойланайық, талқылайық

1. Түрлі есептеулерді жүзеге асыруда графтағы іздеу алгоритмдерін қолдану не үшін қажет?
2. Кезек не үшін қолданылады?
3. «Цикл», «жол», «тізбек» ұғымдары не себепті маңызды?

3 Талдап, салыстырайық

1. Келтірілген теориялық материалдардан басқа да ақпарат көздерін пайдалана отырып, тереңнен іздеу алгоритмі мен көлденеңінен іздеу алгоритмін қадам бойынша жазып, өзара салыстырыңдар.

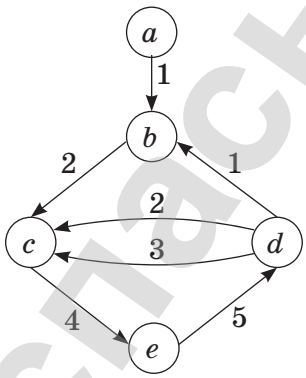
Тереңнен іздеу алгоритмі	Көлденеңінен іздеу алгоритмі

2. Байланысқан граф пен өлшенген графтың қандай айырмашылықтары бар?

4 Дәптерде орындайық

Берілген граф бойынша:

1) графтың қабырғалар тізімін;
2) графтың көршілестік матрицасын;
3) графтың инцидиенттік матрицасын құрыңдар.



5 Компьютерде орындайық

1-тапсырма. Графта тереңнен іздеу алгоритмін жүзеге асыратын программа құрыңдар.

2-тапсырма. Графта көлденеңінен іздеу алгоритмін жүзеге асыратын программа құрыңдар.

3-тапсырма. Графтың берілген төбеден d қашықтықта орналасқан барлық төбелерін анықтау үшін көлденеңінен іздеу алгоритмін қолданыңдар.

Тапсырманы орындауға нұсқаулық

Әрбір тапсырманы графтағы іздеу алгоритмдерінің көмегімен Python программалау тілінде программасын жазыңдар. Графтағы алгоритмдер сипаттамасын, функция мысалын қолданыңдар. Әрбір тапсырма үшін программаны функцияны қолдану арқылы жазыңдар.

Әрбір тапсырманы орындау кезеңі:

- тапсырманың қойылымын толық зерттеу;
- есептің математикалық қойылымын құру;
- есепті шешу әдісін қажетіне қарай таңдау;
- алгоритмнің графикалық сызбасын сызу;
- құрылған алгоритмді Python программалау тілінде жазу;
- программаны бақылау тестін жасау;
- программаны тексеру;
- жұмыстың есебін өткізу.

6 Ой бөлісейік

Қалай ойлайсыңдар, тереңнен іздеу және көлденеңінен іздеу түсініктері байланыспаған графтар үшін қолдана ма? Жауаптарыңды дәлелдеңдер.

4-БӨЛІМ

WEB-ЖОБАЛАУ

Күтілетін нәтижелер:

- web-беттерді әзірлеуде HTML тегтерін қолдану;
- web-беттерді жасауда CSS-ті қолдану;
- web-бетте мультимедиа нысандарын енгізу үшін HTML тегтерін қолдану;
- web-беттерді әзірлеу кезінде дайын скрипттерді пайдалану.

§ 35–36. HTML web-сайттарын әзірлеу әдістері

Естеріңе түсіріңдер:

- *граф дегеніміз не?*
- *графтардағы алгоритмдер қалай орындалады?*

Меңгерілетін білім:

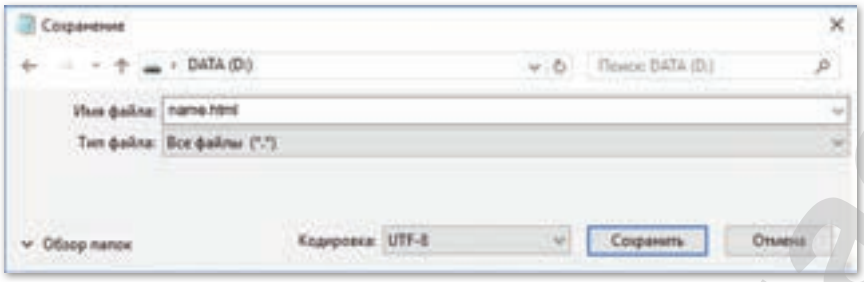
- *HTML тілі;*
- *тег және оның түрлері;*
- *тегтердің атрибуттары;*
- *web-сайт;*
- *web-сайт құрылымы;*
- *web-сайтты жоспарлау кезеңдері.*

Компьютер немесе басқа да электронды құрылғыларда оқылатын мәтіндер **гипермәтін** деп аталады. Гипермәтін терминін алғаш рет американдық әлеуметтанушы, философ Тед Нельсон 1963 жылы айналымға енгізді. Ал 1986 жылы Халықаралық Стандарттау ұйымының «Standart Generalized Markup Language» деп аталатын ISO-8879 стандартын қабылдауы HTML тілінің пайда болуына негіз болды. Гипермәтінді белгілеу тілі (Hyper Text Markup Language) – web-браузерде web-беттерді және басқа да ақпараттарды шығару үшін қолданылатын ең басты белгілеу тілі, яғни құжатқа қойылатын тегтердің көмегімен құжаттың логикалық құрылысын сипаттайды, құжатты форматтауды және нысандарды қоюды басқарады.

Интернеттің бар деректерінің, яғни барлық web-құжаттарының бір ортақ қасиеті – олардың басым көпшілігі HTML тілінде жазылғандығы. HTML тілінде web-құжаттарды жасау программалауға ұқсас болғанымен, ол – қарапайым программалау тілі емес.

HTML – гипермәтінді белгілеу тілі. Ол кәдімгі мәтіндерді web-беттер түрінде бейнелеуге арналған ережелер жиынын анықтайды.

HTML тілінде web-бет немесе сайт құру үшін кәдімгі блокнот қосымшасын қолдануға болады. Кодты блокнотта жазамыз да, сақталу орнын көрсетіп, **Файл ⇒ ... ретінде сақтау** (Сохранить как) ⇒ Файл атауы *name*.html,* ал Файл типтеріне **Барлық файлдарды** (Все файлы), кодировка бөлімінен UTF-8 кодтауын таңдап, **Сақтау** батырмасын басамыз *(21-сурет)*. Жұмыс нәтижесін браузермен ашып қараймыз.



21-сурет. *Веб құжатты сақтау*

Тег дегеніміз не?

HTML тілінде колданылатын командалар «тег» деп аталады. HTML тіліндегі тегтер екі топқа бөлінеді: *жұпты, жұпсыз*.

Жұпты тегтерде бір тег ашылса, келесі тег оны жабады. Мысалы, <html> тегтің жұмысын ашады да, келесі </html> тегі оны жабады (*11-кесте*).

Жұпсыз тегтер – тег ашылып, жабылуды қажет етпей қолданыла беретін тегтер.

11-кесте. Тегтердің қолданылуы

Тег атауы	Қолданылуы	Мысалы
<HTML> ... </HTML>	Бұл тегтер міндетті түрде бетінде болуы керек. Олар браузерлерге және іздеу жүйелеріне бұл HTML беті екенін айтады	<html> <head> ... Тақырыптық тегтер ... </head> <body> ... Беттің негізгі денесі ... </body> </html>
<body> </body>	Осы тегтер арасында парақтың барлық мазмұны көрінеді	
<head> </head>	Осы тегтердің ішінде барлық тақырып тегтері орналасуы керек	
<title> </title>	Бұл тегтер арасында браузердің жоғарғы жағында көрсетілетін бет тақырыбы жазылады	<html> <head> <h2 align = "center"> Тақырып </h2>

Тег атауы	Қолданылуы	Мысалы
	Бұл тегтер ортасында орналасқан барлық нәрселерді ортаға түзетеді	<title> Тақырып 1.1. </title> </head> <body> мәтін 1 <font size = "6" мәтін 2 </h1> </body> </html>
 	Бұл тегтер қаріпті, фонды, өлшемді және т.б. өзгертуге арналған. Бір сөзбен айтқанда, мәтінді пішімдеуге қатысты барлық нәрсе бір тегте конфигурациялануы мүмкін	
	және арасында жазылған барлық қаріптердің қалыңдығын өзгертеді	
	Қаріпті көлбеуге өзгертеді	
<u> </u>	Қаріптің астын сызады	
<s> </s>	Қаріптің бетін сызады	
	Бұл – тақырып тегтерінің тег кластарының бірі болатын <h1> .. <h6> тегтері. Әдетте, бұл беттің атауы болуы мүмкін	html бетін құру үлгісі Тақырып 1 ... Тақырып 1.1 Тақырып 2 ... және т.с.с
 	Бұл – жабу тегін қажет етпейтін жұпсыз тег. Ол мәтінді келесі жолға ауыстырады	<html> <body> ... Тақырып мәтін мәтін </body> </html>
<img alt = "Сілтеме" src = "URL_Сурет">	Бұл – суретті көрсететін жұпсыз тег. src – суреттің мекенжайын көрсететін қажетті параметр (URL_Image орнына, сурет сақталатын мекенжайды тіркеу қажет)	<html> <body> ... <img src = "https://www.pinterest.com/pin/476397085373 47914/_orig.jpg">

Тег атауы	Қолданылуы	Мысалы
		... </body> </html>
	Көлденең сызық беретін жұпсыз тег	
 мәтін_ сілтеме 	Сілтеме құруға арналған тег	a href = "stranica_50.html"> stranica_50.html Беттің толық мекен-жайын жазуға болады
back-ground = "URL"	Фондық суретті анық-тайды. URL орнына фондық сурет мекенжайы жазылады	<html> <body> <table align = "center" width = "100%" border = "1"> <tr> <td colspan = "2"> Кесте мысалы </td> </tr> <tr> <td> Баған 1 </td> <td> Баған 2 </td> </tr> </table> </body> </html>
bgcolor = "түс"	Кестенің түсін анықтайды. Бүкіл кез келген түсті таңдауға болады	
border = "сан "	Кескіннің айналасындағы жақтаудың қалыңдығын анықтайды	
<table> </table>	Кесте жасау тегі. Осы тегтердің арасында кесте орналасады	
<tr>	Жаңа жолды құрады	
<td>	Жаңа бағанды құрады	

Кестеде берілген HTML тегтеріне байланысты мысалдарды өздігінен орындап, нәтижесін браузер терезесінен көріңдер.

Web-құжат – тегтермен толықтырылған мәтіндік файл, оның мәтіндерін бір-бірімен байланыстыра отырып, белгілеуге мүмкіндік беретін HTML тілі.

Web-бет негізгі үш құрамды бөліктен тұрады:

<HTML >

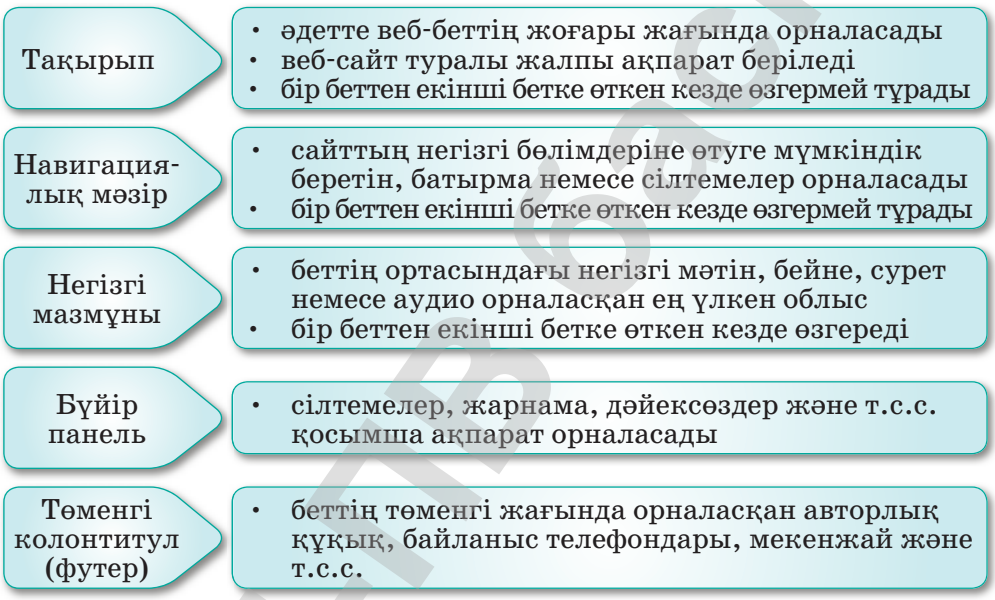
<HEAD >

</HEAD >
<BODY >

Осы тегтердің арасында сайт қолданушысына қолжетімді web-беттің негізгі мазмұны орналасады:

</BODY >
</HTML >

Кез келген web-құжаттың өзіндік ерекшеліктері болады, алайда олардың барлығының құрылымы стандартты компоненттерден тұрады *(5-сызба)*.



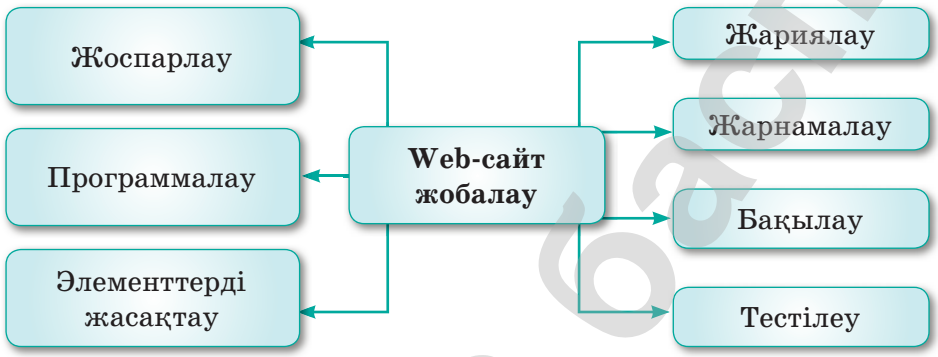
5-сызба. *Web-құжаттың құрылымы*

Web-сайт – бір тақырып бойынша біріктірілген және өзара гиперсілтемелермен байланыстырылған web-беттер жиынтығы. Web-беттер жиынтығы, әдетте, web-сайт атауы мен адресі жазылған бума түрінде серверде орналастырылып қояды. Дайындалған сайттарға web-беттерді енгізу қиын емес. Сондай сайт көлемі үлкен, күрделі құрылымды, иерархиялық түрде болуы мүмкін. Оларға енгізілетін түрлі ақпарат жиі ауыстырыла беретіндіктен, сайтты өңдеп, жаңа деректер енгізіп отыру керек. Мұндай сайттарды тез іздеп табу үшін негізгі беттеріне кілттік сөздер (гиперсілтемелер) енгізіліп қойылады.

Негізінен *web-сайт* – негізгі екі компонентті қамтитын ақпараттық жүйе. Олар:

1) Көрсету компоненті (front – end). Оған мазмұнының көрінісі кіреді (беттер өлшемі, графика, аудио, мәтін).
2) Жүзеге асыру компоненті (back – end). Көрсету компонентіне негіз болып табылатын бейнеленбейтін сценарийлер, серверлік компоненттері бар ағымды кодтарды тиімді жүзеге асыруға байланысты.

Web-сайтты жобалау кезеңдері – сайттың көлемін, функционалдылығын және т.б. анықтау (*6-сызба*).



6-сызба. *Web-сайтты жобалау кезеңдері*

Сайтты жоспарлау кезендері:

1. Алдымен сайттың негізгі міндетін айқындап алу керек.
2. Сайтта қандай ақпарат орналасу керек екендігін анықтау керек.
3. Қажетті ақпаратты жинақтап алу қажет.
4. Сайттың дизайнын нақтылау керек.
5. Сайттың логикалық құрылымын жобалау қажет.
6. Сайттың физикалық құрылымын ойластыру керек, яғни сайтты құрайтын бөлек файлдарды бумаларға бөліп алу керек.
7. Барлығы дұрыс болғандығын тексеру қажет.

Жоспарлау кезеңінде төмендегі мәселелер шешілуі керек:

1) Сайттың орны.
2) Сайттың аудиториясы.
3) Ақпараттың жариялануы.
4) Қолданушылармен қарым-қатынастың ұйымдастырылуы.

Элементтерді жасақтау кезеңінде сайттың программалық өнім түрінде жүзеге асырылуы қарастырылады:

1) навигациялық құрылымын жасау;

2) беттің дизайнын жасау;
3) бетті толтыру үшін мәтіндік және бейне ақпаратты әзірлеу.

Программалау. Бұл кезеңнің мәні сайтты форматтауда.

Тестілеу. Сайт жасаудың негізгі кезеңдерінің бірі – тестілеу. Тестілеу кезеңінде сайттың дұрыс жұмыс істейтіндігі тексеріледі, оның ішінде:

1) сілтеменің жұмысы;
2) мәтіндегі қателер;
3) навигацияның тиімділігі;
4) пошта және басқа формалардың дұрыстығы;
5) графикалық файлдардың ашылуы;
6) сайттың әртүрлі браузерлердегі жұмысы.

Жариялау. Тест аяқталғаннан кейін web-сайт серверде жарияланады және қайтадан тексеріледі.

Жарнамалау. Web-қоғамдастығына жаңадан жарияланған сайт туралы белгілі болу үшін сайттың адресін және ол жердегі материал туралы аннотацияны хабарлау керек. Осы мақсатқа жету үшін көрсетілген мүмкіндіктерді пайдалануға болады:

1) web-сайт адресін әртүрлі баспаларға жазу керек;
2) web-сайтты әртүрлі серверлерде тіркеу;
3) web-сайтқа сілтемелерді басқа web-сайттарға кіргізу;
4) баннерлерді жарнама ретінде қолдану.

Бақылау. Web-сайтта жариялап, жарнамалаған соң оған қатысу деңгейі оның беттерінде орналастырылған ақпараттың қажеттілігімен, жаңалығымен және көкейтестілігімен анықталады. Web-сайт имиджін сақтау үшін ол жердегі ақпаратты әрдайым жаңартып тұру керек.

Web-сайтты **өңдеу** мынадай кезеңдерден тұрады:

- сайттың негізгі және типтік беттерінің дизайны (графикалық редакторда орындалады);
- HTML-кодтау, браузер көмегімен көруге болатын код құрылады;
- сайтты программалау;
- web-те сайтты орналастыру;
- web-те көрінуін жоғарылату мақсатында web-сайтты оптимизациялау;
- сайтты тапсырыс берушіге өткізу.

Сайт құрылған соң, оны web-серверде **жариялау** керек. Оның үш тәсілі бар:

1. **Ең қарапайым тәсіл.** Сайттың барлық файлдары диск немесе басқа ақпарат тасушыға жазылады. Оны web-сервер администраторына апару керек. Администратор біздің дискіміздегі бумаларды сервердің қажетті бумасына жазып, программалық жасақтаманы баптайды.
2. **Өте аз тараған тәсіл.** Кейбір тегін web-серверлер пайдаланушыға оның сайтының файлдарын web-шолушы арқылы жүктеуге мүмкіндік береді. Бұл тәсілдің жетістігі – жүктеудің қарапайымдылығы. Пайдаланушы енгізу өрісіне қажетті файлдардың атауларын жазып, Submit (жіберу) батырмасын басу керек.
3. **Ең кең тараған тәсіл.** Web-сервер администраторы FTP-сервер программасын орнатады. Содан соң автордың сұранысы бойынша осы сайт үшін бума құрады және авторға тек осы бумаға ғана кіру мүмкіндігін береді. Сайттың авторы FTP-клиент программасы арқылы FTP-серверге қосылып, құрылған түпкі бумаға сайтының файлдарын көшіреді. Содан соң администратор web-серверде жаңа сайттың бар екенін жариялайды. Егер авторға кейбір файлдарды жаңарту керек болса, ол қайтадан FTP-серверге қосылып, ескі файлдарды жойып, жаңа файлдарын көшіреді.

1 Сұрақтарға жауап берейік

1. Гипермәтін деген не?
2. HTML қалай пайда болды?
3. Web-беттерді қандай программалар көмегімен жасауға болады?
4. HTML тілінде web-бетті қайда құруға болады?
5. Браузер-программалардың қандай түрлерін білесіңдер?

2 Ойланайық, талқылайық

1. Гипермәтін термині не үшін енгізілді?
2. HTML тілінде web-бетті жасаудың тиімділігі неде?
3. Не себепті HTML тілінде тегтер жұпты және жұпсыз тегтер деп бөлінеді?
4. Негізгі тегтерді қолдануда неліктен реттілікті сақтау керек?

3 **Талдап, салыстырайық**

HTML тегтерін пайдаланып, web-сайт жасаудың маңыздылығын талдап, сайт құрылған соң, оны web-серверде жариялаудың үш тәсілін салыстырыңдар.

4 **Дәптерде орындайық**

Кестені дәптерге толтырыңдар.

Тегтер мен атрибуттар	Пайдаланылатын кезі және атқаратын қызметтері

5 **Компьютерде орындайық**

1. Блокнот программасын ашып, бос құжатты тапсырма 1. HTML деген атпен сақтаңдар.
2. `<HTML>`
   `<HEAD> <TITLE>` Менің алғашқы web-сайтым `</TITLE> </HEAD>` *(Терезенің тақырыбын енгізу).*
3. `<BODY BGCOLOR = yellow TEXT = blue>` *(құжаттың денесі).*
   Мұндағы `BGCOLOR = yellow` құжаттың фоны – сары, ал `TEXT = blue` шығатын мәтіннің түсі көк болатынын анықтайды.
4. `<H1>` Менің алғашқы жұмысыма қош келдіңдер `</H1>`.
5. `<H2>` Сіздерге web-сайт жасау жолдарын көрсетемін `</H2>`.
6. Бұл жолдар әртүрлі деңгейдегі тақырыптарды анықтайды.
7. HTML құжатта көрсетілетін ақпарат таусылғандықтан, `<BODY>` тегін жабу керек, ол үшін `</BODY>` деп жазыңдар.
   HTML құжаттағы жазу аяқталды, сондықтан `<HTML>` тегін де жабу керек, ол үшін `</HTML>` деп жазыңдар.
   Блокнотты жауып, жасаған жұмыстарыңды браузерде ашыңдар.

6 **Ой бөлісейік**

1. HTML web-сайтын әзірлеу үшін өздерің қай әдісті таңдар едіңдер?
2. Тегтердің қолданылуына мысал келтіріңдер.

§ 37. Практикум. HTML web–сайттарын құру

Практикумға арналған тапсырмаларды орындамас бұрын сайтта қолданылатын түстер туралы сөз қозғап өтейік.

Түс – web-сайттың маңызды құраушысы және қолданушыларға ықпал етудің қуатты факторы. Түстер сайт қолданушыларында түрлі эмоциялар, белгілі бір іс-әрекеттерді орындауына себепші болуы мүмкін. Сайт үшін түстік гамманы таңдауда, түстер теориясының негізгі принциптерін басшылыққа алу керек.

Кез келген сайт элементінің түсін қалыптастыруда көбінесе RGB моделі қолданылады. RGB моделі аддитивтілік принципін (ағылшынша *add* – «қосу») негізге ала отырып, үш түрлі Red, Green, Blue түстері арқылы 16,7 миллион түрлі түсті береді.

RGB-түс қызыл, жасыл, сары түстерді түрлі пропорцияда бір-біріне қосу нәтижесінде алынады (*22-сурет*).

22-сурет. *RGB моделі*

HTML-де RGB модель түстерін жазу үшін он алтылық немесе #RrGgBb деп аталатын жазба қолданылады: әр координата екі он алтылық сан түрінде бос орынсыз жазылады. Мысалы, ақ түстің #RrGgBb-жазбасы — #FFFFFF түрінде жазылады. Түстің ашықтығы 0–255 диапазонында анықталады (мысалы, көк түс – 0, 0, 255, қызыл – 255, 0, 0, қара – 0, 0, 0 және ақ – 255, 255, 255).

Негізгі түстердің #RrGgBb-жазбалары

Black	#000000	0, 0, 0
Gray	#808080	128, 128, 128
Silver	#C0C0C0	192, 192, 192
White	#FFFFFF	255, 255, 255
Fuchsia	#FF00FF	255, 0, 255
Purple	#800080	128, 0, 128
Red	#FF0000	255, 0, 0
Maroon	#800000	128, 0, 0
Yellow	#FFFF00	255, 255, 0
Olive	#808000	128, 128, 0
Lime	#00FF00	0, 255, 0
Green	#008000	0, 128, 0

Негізгі түстердің #RrGgBb-жазбалары

Aqua	#ØØFFFF	Ø, 255, 255
Teal	#ØØ8Ø8Ø	Ø, 128, 128
Blue	#ØØØØFF	Ø, Ø, 255
Navy	#ØØØØ8Ø	Ø, Ø, 128

Тапсырмаларды орындауға арналған нұсқаулық

1. «Оқушы» бумасында «Менің алғашқы web-сайтым» атты бума жасаңдар.
2. *Блокнот* программасын ашыңдар.
3. Блокнот редакторында қарапайым HTML файл мәтінін жазыңдар.
4. HTML файлды сақтаңдар: Файл ⇒ ... ретінде сақтау ⇒ Жұмыс үстелі ⇒ «Менің алғашқы web-сайтым» бумасы ⇒ Файлдың атауы «Алғашқы бет.html» ⇒ Сақтау командаларын орындаңдар.
5. Web-бетті көру үшін желі браузерін қосу керек.

1-тапсырма. Браузер бетіне өз аты-жөндерің шығып тұратындай web-бет құрыңдар.

2-тапсырма. «Досыма хат»

Браузер бетінде досыңа арналған хат шығару керек. Дос жайлы нақыл сөздерді қосып, хаттарыңды көркем түрде жазыңдар.

3-тапсырма. Құжаттың тақырыбының түсін өзгертіңдер.

4-тапсырма. Дайындаған құжаттың фон түсін өзгертіңдер.

5-тапсырма. «Менің хоббиім»

«Менің хоббиім» туралы web-бетті құрыңдар. Web-бет ортаға түзетілген «Менің хоббиім» тақырыбынан, өздерің туралы қысқаша сипаттамадан және өздеріңнің қызығушылықтарың туралы (спорт, ән айту, билеу және т.б.) тізімнен тұруы керек.

§ 38–39. Мәтінді форматтау (қаріп, абзац, тізімдер)

Естеріңе түсіріңдер:

- *гипермәтін дегеніміз не және бұл терминді ең алғаш кім енгізді?*
- *web-беттерді қандай программалар көмегімен жасауға болады?*
- *HTML тілінде web-бетті қайда құруға болады?*
- *тег дегеніміз не?*

Меңгерілетін білім:

- *мәтінді форматтау тегтері;*
- *абзац енгізу тегтері;*
- *тізімдерді жазу тегтерінің түрлері;*
- *сайтқа сырғымалы жолдар енгізу.*

Мәтінді форматтау тегтері

Құжатта тақырыптар <H>, </h> тегтерімен жасалады. Мысалы,

```
<H1> Менің атым1 </H1>
<H2> Менің атым2 </H2>
<H3> Менің атым3 </H3>
<H1> Менің атым4 </H1>
<H2> Менің атым5 </H2>
<H3> Менің атым6 </H3>
```

<Hi> белгісі (мұндағы i – 1-ден 6-ға дейінгі бүтін сан) алты түрлі сатыдағы символдар мөлшерін таңдау мүмкіндігін береді. Бірінші сатыдағы тақырып – ең ірісі, алтыншы сатыдағы – ең кішісі.

Менің атым1

Менің атым2

Менің атым3

Менің атым4

Менің атым5

Менің атым6

Абзац енгізу үшін <P>, </ p> тегтері қолданылады, ал мәтіннің сол жақ, ортада немесе оң жақта орнату үшін *align* атрибуты пайдаланылады. Мысалы, <P align = center> Менің бірінші бетім </ p>. Мұнда «Менің бірінші бетім» сөйлемі беттің ортасына орналасады. Align атрибутының мәні left (сол жақ), right (оң жақ) болуы мүмкін.

<FONT> тегінің көмегімен біз мәтіннің қарпін, көлемін, түсін белгілейміз. Ол үшін face, size, color атрибуттары пайдаланылады. Мысалы: <P align = center> Менің бірінші бетім . Мұнда arial қарпімен, көлемі 5-ке тең, көк түсті «Менің бірінші бетім» деген сөйлем шығады.

Жолды бөлу үшін <**BR**> тегін қолдануға болады. Мысалы:

<P> Ана тілің – арың бұл,

 Ұятың боп тұр бетте </p> деген кодта
 тегі екінші сөйлемді келесі жолға түсіреді.

Қарайтылған әріптерді пайдалану үшін мәтінді **<B>, ** тегтерінің ортасына аламыз, көлбеу (курсив) әріптер үшін – **<I>, </i>** . Мысалы:

Көрінетін мәтін, яғни мысалға 10 «А» сыныбы Информатика пәні HTML-де жазылуы:

```
<HTML <HEAD> <H3> 10 "A" сыныбы Информатика пәні
</H3> </HEAD>
<BODY> <P> <B> 10 "A" сыныбы Информатика пәні </B>
<P> <I> 10 "A" сыныбы Информатика пәні </I>
<P> <U> 10 "A" сыныбы Информатика пәні </U>
<P> <S> 10 "A" сыныбы Информатика пәні </S>
<P> <TT> 10 "A" сыныбы Информатика пәні </TT>
</BODY> </HTML>
```

10 «А» сыныбы Информатика пәні

10 «А» сыныбы Информатика пәні

<I> **10 «А» сыныбы Информатика пәні**

10 «А» сыныбы Информатика пәні

~~10 «А» сыныбы Информатика пәні~~

10 «А» сыныбы Информатика пәні

Нөмірленген тізімдер

Нөмірленген тізімдер белгіленген тізім тәрізді шығарылады, олар тек <O<UL> және </O<UL> (ordered list – реттелген тізім) тегтерімен қоршалады да, нәтижесінде тізім нөмірі ретінде бүтін сандар жазылады. Бір мысалды аздап түрлендіріп, нөмірлеп жазып шығайық:

```
<HTML> <HEAD> <TITLE> мысал </TITLE> </HEAD>
   <BODY text = green>
      <H2 ALIGN = CENTER> Нөмірленген тізім жолдары
      </H2> <HR>
      <O<UL>
         <LI> Айдар;
```

```
            <LI> Марат;
            <LI> Айдос;
            <LI> Жанна </LI>
            </O<UL>
        <HR>
    </BODY>
</HTML>
```

Осы HTML тегтері жұмысы нәтижесінде мынадай тізім шығарылады:

Нөмірленген тізім жолдары

1. Айдар;
2. Марат;
3. Айдос;
4. Жанна

Егер тізім нөмірін керекті бір нөмірмен бастау керек болса, онда `start` атрибутын пайдаланамыз, мысалы:

`<OL start = 5>` Қолданылған әдебиеттер тізімі

Тізімдің түрін өзгерту үшін type атрибуты көмектеседі, мысалы, нөмірлерді латын әріптерімен төмендегідей жазамыз:

`<OL type = I>` Қолданылған әдебиеттер тізімі

Маркерлік тізімді жазғанда `<UL>` тегін пайдаланады, маркердің түрін өзгерту үшін type атрибутын қолданамыз.

`<LI type = disk>` – нүкте түріндегі маркер;

`<LI type = circle>` – шеңбер түріндегі маркер;

`<LI type = square>` – шаршы түріндегі маркер.

Нөмірленбеген тізімдер

`<UL>` және `</U<UL>` (unordered list – реті жоқ тізім) тегтері арасында орналасқан мәтіндер нөмірленбей, бірақ белгіленіп жазылған тізімдер ретінде қарастырылады. Мұнда тізімнің әрбір жаңа элементін `<LI>` (list – тізім) белгісінен бастап жазу қажет. Мысалы, экранда жасыл түсті әріптермен терілген мынадай тізім жасау үшін:

- Айдар;
- Марат;

- Айдос;
- Жанна

Төмендегідей түрде HTML тілі мәтінін Блокнотта теріп, кез келген браузерде көріп шығу керек:

```
<HTML> <HEAD> <TITLE> мысал </TITLE> </HEAD>
   <BODY text = green>
      <H2 ALIGN = CENTER> Маркерленген тізім жолдары
      </H2> <HR>
         <UL>
            <LI> Айдар;
            <LI> Марат;
            <LI> Айдос;
            <LI> Жанна
         </U<UL>
         <HR>
   </BODY>
</HTML>
```

Маркерленген тізім жолдары

- Айдар;
- Марат;
- Айдос;
- Жанна

<LI> белгісі үшін жабылу тегінің қажет емес екенін байқаған боларсыңдар.

<UL> тегінің атибуты type = disc | circle | square, олар белгі маркерінің сыртқы пішінін өзгертіп, сәйкесінше дөңгелек | шеңбер | квадрат бейнесінде көрсете алады.

Қабатталған тізімдер

Кез келген тізімдегі элемент ішіне басқа да тізім түрлері енгізіліп, қабаттасқан тізімдер құрастырылуы да мүмкін. Бірақ мұндай қабатталған тізімдерді жиі қолдансақ, ол мәтінді экранға көпсатылы түрде шығарып, оның ұзындығын арттырып жібереді. Сондықтан оларды тек керек кезінде ғана қолданған дұрыс деп саналады.

Қабатталып орналасқан тізімдерді мәтін мазмұндары мен әртүрлі жоспарлар дайындауда қолданған тиімді болып табылады.

Осындай тізімдер ұйымдастыруды қысқаша мынадай мысалмен көрсетейік:

```
<html> <head> <title> мысал </title> </head>
<body>
<H1 ALIGN = center> HTML-де жұмыс жасаған қызықты
</H1>
<DL>
<DT> Нөмірленбеген тізімдер
<DD> Нөмірленбеген тізім элементтері сол жақтан
арнайы таңбамен белгіленіп, мәтін аздап оңға таман
жылжып орналасады:
<UL>
<LI> 1-элемент
<LI> 2-элемент
<LI> 3-элемент
</UL>
<DT> Нөмірленген тізім жолдары
<DD> Нөмірленген тізім жолдары сол жақтан нөмірлер
арқылы белгіленіп орналасады:
<OL>
<LI> 1-элемент
<LI> 2-элемент
<LI> 3-элемент </LI>
</ol>
<DT> Анықтау тізімдері
<DD> Мұндай тізімдер алдыңғы екеуінен күрделірек,
бірақ  оқуға ыңғайлы болады.
<P> Тізімдерді бірінің ішіне бірін жазып қабаттасты-
руға болады,  бірақ көп деңгейлер жасап, бұл тәсілмен
тым  әуестеніп кету қажет емес екені есте болсын.
<P > Тізімдегі бір элемент ішінде бірнеше абзац
тұруы мүмкін.
Ондай абзацтар сол жақ шеттен бірдей қашықтыққа
ығысып  орналасады. </P>
</DL>
</body>
</html>
```

Сырғымалы жолдар

<MARQUEE> және </MARQUEE> тегтері браузер терезесінде жолдың бір шетінен екінші шетіне жылжып отыратын «сырғымалы жол» жасайды және оның мынадай параметрлері болады:

```
<MARQUEE [ALIGN="align'] [BEHAVIOR='behavior"]
[BGCOLOR = #rrggbb]
[DIRECTION="direction"] [HEIGHT="integer"]
[HSPACE="integer"]
[LOOP="integer"] [SCROLLAMOUNT="integer"]
[SCROLLDELLAY="integer"]
[VSPACE="integer"] [WIDTH="integer"]> Кез келген
мәтін </MARQUEE>
```

Осылардың кейбірінің мағыналары мен жазылу түрі төмендегідей болып келеді.

ALIGN – «сырғымалы» мәтінді жолдың жоғарғы шетіне, ортасына немесе төменгі шетіне туралап орналастыру параметрін береді және ол мына мәндердің (сөздердің) біреуін қабылдайды: TOP, MIDDLE, BOTTOM.

BGCOLOR – «сырғымалы жолдың» фон түсін анықтайды, он алтылық RGB форматында немесе ағылшынша белгілі бір түс аты беріледі.

DIRECTION – жол бойынша жылжу, яғни сырғу бағытын анықтайды, оның мүмкін мәндері: LEFT (солға), RIGHT (оңға) және оның мәні көрсетілмеген жағдайда, келісім бойынша LEFT мәні автоматты түрде іске қосылады. Екі жаққа да кезек-кезек сырғыту үшін BEHAVIOR=ALTERNATE атрибуты қолданылады.

HEIGHT – «сырғымалы жолдың» биіктігін пиксель (нүктелер) арқылы анықтайтын бүтін сан, оны пайызбен де (%) көрсетуге болады.

LOOP – «сырғымалы жолдардың» қайталану санын анықтайтын бүтін сан, *INFINITE* (шексіздік) мәнін қабылдауы да мүмкін.

SCROLLAMOUNT – жылжудың бір қадамында мәтіннің қанша пикселге жылжитынын анықтайтын бүтін сан.

SCROLLDELAY – екі сырғудың арасындағы интервалды миллисекундпен көрсететін бүтін сан.

WIDTH – экрандағы «сырғымалы жолдың» енін пиксель арқылы анықтайтын бүтін сан, оны процентпен де (%) көрсетуге болады.

Осы сырғымалы жол жасау үшін бір мысал қарастырайық. Блокнотта теріп, мысал.htm атымен сақтап, оның нәтижесін кез келген браузерде көріп шығу керек:

```
<HTML> <HEAD> <TITLE> 3-1 мысал </TITLE> </HEAD>
   <BODY text=red>
      <CENTER>
      <H2> Сырғымалы жолдар </H2> <HR>
      <H3> <MARQUEE BGCOLOR= "yellow"
      DIRECTION = "RIGHT"
         SCROLLAMOUNT = "10" SCROLLDELAY="200"
      WIDTH="90%">
         Бұл – бірінші сырғымалы жол
            </MARQUEE>
      <P> <MARQUEE BGCOLOR= "green" DIRECTION =
      "LEFT"
      HEIGHT=30 SCROLLAMOUNT="10"
      SCROLLDELAY="100" WIDTH="90%">
         Бұл – екінші сырғымалы жол </MARQUEE>
         </H3> <HR>
      <H3> <MARQUEE BGCOLOR= "blue"
      BEHAVIOR=alternate
            SCROLLAMOUNT = "10" SCROLLDELAY=
            "200" WIDTH="90%">
            Бұл – үшінші сырғымалы жол
            </MARQUEE>
         </CENTER> </BODY>
</HTML>
```

1 Сұрақтарға жауап берейік

1. Мәтінді форматтауға нелер жатады?
2. Сырғымалы жолдар деген не?
3. Сырғымалы жолдарды қандай тегтермен ұйымдастырады?

2 Ойланайық, талқылайық

1. Неліктен мәтінді форматтау тегтерден тұрады?
2. Неліктен сырғымалы жолдарды қолданамыз?

3 Талдап, салыстырайық

1. Нөмірленген тізім мен нөмірленбеген тізімнің айырмашылығы неде?
2. Сырғымалы жолдарды салыстырып, талдау жасаңдар.

4 Дәптерде орындайық

1. Тізімді ұйымдастыру тегтерінің қолданылуына мысал келтіріңдер.
2. Тегтердің қолдануына қарай кестеге толтырыңдар.

HTML жалпы құрылымы	Азатжолды форматтау	Қаріптерді форматтау
–	–	–
–	–	–
–	–	–

5 Компьютерде орындайық

Блокнот мәтіндік редакторын ашып, бос құжатты *Тапсырма2.html* деген атпен сақтаңдар.

1. Құжаттағы алғашқы тег <HTML> тегі болу керек. Терезенің тақырыбын енгізу үшін мына жолдарды теріңдер:
```
<HEAD> <TITLE> Менің бірінші бетім </TITLE>
</HEAD>
```
2. Енді құжаттың денесін, яғни HTML құжатта не көрсетілетінін енгіземіз. Ол үшін мына жолдарды теріңдер:
```
<BODY BGCOLOR = yellow TEXT = blue>
```

Мұндағы BGCOLOR = yellow құжаттың фоны – сары, ал TEXT = blue шығатын мәтіннің түсі көк болатынын анықтайды.

3. Енді нөмірленген және нөмірленбеген тізім жасауды үйренейік. Ол үшін мына жолдарды енгізіңдер:

```
<O<UL> Бұл бірінші нөмірленген тізімді білдіреді.
<LI> оқушы
<LI> мұғалім </LI>
</O<UL>
<UL TYPE = DISC> Бұл нөмірленбеген тізбекті білдіреді.
<LI> Бірінші оқушы
<LI> Екінші оқушы
<LI> Үшінші оқушы
</U<UL>
```

4. Келесі жолға

 деп, теріңдер. Бұл екі рет бос жол жасағанды білдіреді («Enter» пернесін басқанды білдіреді).
5. Келесі жолға теріңдер: <MARQUEE> СЫРҒЫМАЛЫ ЖОЛ </MARQUEE>. Нәтижесінде сырғымалы жол жасалынады.
6. HTML құжатта көрсетілетін ақпарат таусылғандықтан <BODY> тегін жабу керек, ол үшін </BODY> деп жазыңдар. HTML құжаттағы жазу аяқталды, сондықтан <HTML> тегін де жабу керек, ол үшін </HTML> деп жазыңдар.

   Блокнотты жауып, жасаған құжатты браузерде көріңдер.

6 Ой бөлісейік

1. Мәтінді форматтау тегтерін пайдаланудың тиімділігі неде?
2. Сырғымалы жолдарды қайда пайдаланады?

§ 40. Практикум. Мәтінді форматтау

Тапсырмаларды орындауға арналған нұсқаулық

1. «Оқушы» бумасында «Менің алғашқы web-сайтым» атты бума жасаңдар.
2. *Блокнот* программасын ашыңдар.
3. Блокнот редакторында қарапайым HTML файл мәтінін жазыңдар.
4. HTML файлды сақтаңдар: Файл ⇒ ... ретінде сақтау ⇒ Жұмыс үстелі ⇒ «Менің алғашқы web-сайтым» бумасы ⇒ файлдың атауы «Алғашқы бет.html» ⇒ Сақтау командаларын орындаңдар.
5. Web-бетті көру үшін желі браузерін қосу керек.

1-тапсырма. Жаңа жолға көшу (абзац) тегін пайдаланып өлең жолын енгізу.

Web-сайттың бетіне кез келген өлең жолын жазыңдар.

Тапсырманы орындауға арналған нұқсаулық

1. Сайттарыңның барлық файлдарын сақтайтын жеке папка құрыңдар.
2. *Блокнот* (*Notepad*) программасын іске қосыңдар (Бастау ⇒ Барлық программалар ⇒ Стандартты).

```
<html> <head> <title> мысал </title> </head>
<body>
<H1> ӨЛЕҢ </H1>
<H2> Абай </H2>
<P> Айттым сәлем,
қаламқас, <BR>
Саған құрбан мал мен бас. <BR>
Сағынғаннан сені ойлап <BR>
Келер көзге ыстық жас. </P>
<P> Көзімнің қарасы, <BR>
Көңілімнің санасы. <BR>
Бітпейді іштегі, <BR>
Ғашықтық жарасы. </P>
</body>
</html>
```

ӨЛЕҢ

Абай

Айттым сәлем, қаламқас,
Саған құрбан мал мен бас.
Сағынғаннан сені ойлап
Келер көзге ыстық жас.

Көзімнің қарасы,
Көңілімнің санасы.
Бітпейді іштегі,
Ғашықтық жарасы.

3. Файлды *Тапсырма2.html* (сақтау кезінде міндетті түрде HTML файлдың типін көрсетіңдер) деген атпен жеке бумада сақтаңдар.

Web-беттерді қарау үшін браузердің кез келген программасын қолданыңдар (*Internet Explorer*, *Opera*, *Mozilla Firefox* немесе басқасын). Ол үшін Блокнот программасынан шықпай, жеке папканы ашып және *тапсырма2.html* файлын екі рет басу арқылы браузер терезесін ашыңдар.

2-тапсырма. Экрандағы мәтіннің орналасуын бақылау

Наурыз мерекесіне байланысты, Қазақстан халқына құттықтау жазыңдар. Нәтижесін экранға шығарып, өз қалауларыңша өзгерістер енгізіп, қайталап сақтаңдар. Өзгерістердің браузерде енгенін тексеріңдер.

3-тапсырма. Сыныптастарыңның тізімін орналастыру

Сыныптағы оқушылардың тізімін жазыңдар. Нәтижесіне кейін өз қалауларыңша өзгерістер енгізіп, қайталап сақтаңдар.

Тапсырманы орындауға арналған нұсқаулық

1. Енгізілген тізімдегі мәтіндерді қажеттеріңе қарай таңдап, түстерін, өлшемін, қарпін орнатыңдар.
2. Құжат бетіне нөмірленген, нөмірленбеген тізімдерін шығарыңдар.

4-тапсырма. Төмендегі мәтінді шығаратын web-бетті құрыңдар.

Шар бетінің ауданы $S = 4\pi r^2$
Судың формуласы H_2O
~~Мәтін~~
HTML редакторы

5-тапсырма. HTML тілінің тегтерін пайдаланып, төмендегі тапсырманы орындаңдар.

1. **Жұмыс күндері:**
   a. дүйсенбі
   b. сейсенбі
   c. сәрсенбі
   d. бейсенбі
   e. жұма
2. **Демалыс күндері:**
   a. сенбі
   b. жексенбі

§ 41–42. Кестелер

Естеріңе түсіріңдер:

- *мәтінді форматтау тегтері;*
- *тізімді шығару тегтері;*
- *сырғымалы жолдарды шығару тегтері.*

Меңгерілетін білім:

- *HTML-де кестелермен жұмыс жасау;*
- *кестелерді шығару тегтерін білу.*

Кесте – web-сайттың негізгі бөліктерінің бірі. Кесте мынадай бөліктерден тұрады:

- кесте тақырыбы;
- бағандар тақырыптары;
- ұяшықтар.

Кесте жолдар тізбегі бойынша біртіндеп толтырылады (солдан оңға қарай жол соңына дейін, содан соң келесі жолға көшу). Әрбір ұяшыққа мәліметтер енгізіледі. Кесте тұрғызу <TABLE> және </TABLE> тегтері көмегімен орындалады, оның әрбір жолын анықтау – <TR> және </TR> тегтері арқылы, ал сол жолдардағы бағандар – <TD> және </TD> немесе <TH> және </TH> тегтері арқылы анықталады. <TD> және <TH> тегтерінің жұмысы ұқсас, бірақ <TH> тегтерімен қоршалған мәтін қарайтылған баған тақырыптары болып табылады да, <TD> тегтерімен одан кейінгі қарапайым бағандар жазылады.

Кесте тақырыбы <CAPTION> және </CAPTION> тегтерімен қоршалып жазылады. Жалпы кестені толық анықтау ережесі төмендегі үлгімен беріледі:

```
<TABLE ALIGN = "center" BGCOLOR = "#rrggbb"
BORDER = "integer"
   BORDERCOLOR = "#rrggbb" WIDTH = "integer">
......
</TABLE>
```

Бірақ кесте тұрғызу кезінде олардың кейбірі қолданылмауы да мүмкін. Енді осы кесте тегі атрибуттарының атқаратын жұмысына тоқталайық:

ALIGN атрибуты кестенің шет жақтарға туралануын анықтайды (көрсетілмесе, келісім бойынша сол жақ шетке). ALIGN мәні – қос тырнақша ішіндегі сөз – мына сөздердің біріне сәйкес келуі тиіс: LEFT (сол жақ шетке), CENTER (ортаға), RIGHT (оң жақ шетке).

BGCOLOR кесте торының ішкі фон түсін тағайындайды (он алтылық RGB форматындағы сан немесе ағылшын тіліндегі белгіленген түс атауы).

BORDER – бүтін сан, кесте жақтаулары сызығының пиксельмен берілген қалыңдығы. Егер BORDER берілмесе, жақтау сызықтары көрсетілмейді.

BORDERCOLOR жақтау сызықтарының түсін тағайындайды (он алтылық RGB форматындағы сан немесе ағылшын тіліндегі белгіленген түс атауы), BORDER атрибутымен бірге қолданылады.

WIDTH – кесте енін анықтайтын бүтін сан, оның мәні пиксельмен немесе процентпен (%) беріледі.

Кесте тақырыбы <CAPTION> тегімен төмендегі ережеге сәйкес беріледі:

```
<CAPTION ALIGN = "top"> ..... </CAPTION>
```

Мұндағы атрибуттардың атқаратын қызметі мынадай болады.

ALIGN атрибуты кесте тақырыбын шет жақтарға туралау кезінде оның мәні LEFT, CENTER (көрсетілмесе, келісім бойынша осы мән қабылданады), RIGHT сөздерінің біріне сәйкес келуі тиіс. Ал егер ол тақырыпты вертикаль бағытта кестенің жоғарғы және төменгі жақтарына орналастыруы қажет болса, онда BOTTOM – жоғарыда (келісім бойынша осы мән қабылданады), TOP – төменде сөздердің бірін мән ретінде қабылдай алады.

Кесте жолы **<TR>** және </TR> тегтерімен қоршалып тұрады, бұлардың алғашқысының мынадай бірсыпыра атрибуттары болуы мүмкін:

```
<TR ALIGN = "center" BGCOLOR = "#rrggbb"
BORDERCOLOR = "#rrggbb"> Кесте жолы… </TR>
```

Енді **<TR>** тегінің осы атрибуттарына тоқталайық.

ALIGN – жол шеттерін туралау. Оның мүмкін мәндері LEFT (келісім бойынша), CENTER, RIGHT.

BGCOLOR жолдың ішкі фон түсін анықтайды (он алтылық RGB форматындағы сан немесе ағылшын тіліндегі белгіленген түс атауы).

BORDERCOLOR – жол жақтауларының түсі (он алтылық RGB форматындағы сан немесе ағылшын тіліндегі белгіленген түс атауы). Бұл атрибут <TABLE> тегінің BORDER атрибутының мәні нөлге тең болмағанда ғана қолданылады.

ROWSPAN – бірнеше жолды біріктіріп, бір-ақ жол етіп жазу кезінде берілетін біріктірілетін жолдар саны.

Кесте жолындағы бағандар (ұяшықтар) **<TD> ... </TD>** және **<TH> ... </TH>** тегтерімен төмендегі ережеге сәйкес анықталады:

```
<TD немесе TH ALIGN = "right" BACKGROUND = "url"
[BGCOLOR = "#rrggbb" BORDERCOLOR = "#rrggbb"]>
Баған… </TD немесе /TH>
```

<TD> және <TH> тегтерінің атрибуттары қызметтері:

ALIGN – мәтінді горизонталь бағытта туралау атрибуты. Оның мүмкін мәндері: LEFT, CENTER (келісім бойынша), RIGHT.

BGCOLOR – фон түсін анықтау атрибуты (он алтылық RGB форматындағы сан немесе ағылшын тіліндегі белгіленген түс атауы).

BORDERCOLOR – ұяшық жақтауларының түсі атрибуты (он алтылық RGB форматындағы сан немесе ағылшын тіліндегі түс атауы). Бұл атрибут <TABLE> тегінің BORDER атрибутының мәні нөлге тең болмағанда ғана қолданылады.

COLSPAN – бағандар тақырыбына арналған біріктірілген жол (ұя) ішінде орналасатын бағандар саны.

Оқушылар тобы жайлы кесте құрып, мысал келтірейік. Блокнот парақшасын ашып, құратын кестенің кодын блокнот бетіне орналастырамыз.

```
<html> <body text = blue> <table border = 10
bordercolor = green
        width = 100% align = center bgcolo =
"yellow">
  <caption align = bottom> <H2> Оқушылар жайлы
  дерек </H2>
  </caption>
  <tr> <th colspan = 4> 6-сынып оқушыларының тізімі
  </th> </tr>
  <tr> <th>   </th> <th> Аты-жөні </th> <th>
  Мекенжайы </th>
```

```
        </tr>
      <tr> <th rowspan = 3> Ұлдар </th> <td> Жаңабаев
      Жандос </td> <td>
        Сәтбаев көшесі, 10-үй, 126 пәтер </td> </tr>
      <tr> <td> Мұқан Қуаныш </td> <td> Әл-Фараби
      даңғылы, 15-үй, 225-пәтер </td>
   </tr>
      <tr> <td> Төреғали Данияр </td> <td> Сейфуллин
      көшесі, 101-үй, 6-пәтер </td>
        </tr>
      <tr> <th rowspan = 3> Қыздар </th>
        <td> Сәкен Сандуғаш </td> <td> Сейфуллин
        көшесі, 101-үй, 25-пәтер
        </tr>
      <tr> <td> Бақытжанқызы Айман </td> <td> Абай
      даңғылы, 115-үй, 121-пәтер
        </td> </tr>
      <tr><td>Төлебай Жанар</td><td>Мұратбаев көшесі,
      36-үй, 95-пәтер
        </td> </tr>
      </table>
   </body>
</html>
```

6-сынып оқушыларының тізімі		
	Аты-жөні	**Мекенжайы**
Ұлдар	Жаңабаев Жандос	Сәтбаев көшесі, 10-үй, 126-пәтер
	Мұқан Қуаныш	Әл-Фараби даңғылы, 15-үй, 225-пәтер
	Төреғали Данияр	Сейфуллин көшесі, 101-үй, 6-пәтер
Қыздар	Сәкен Сандуғаш	Сейфуллин көшесі, 101-үй, 25-пәтер
	Бақытжанқызы Айман	Абай даңғылы, 115-үй, 121-пәтер
	Төлебай Жанар	Мұратбаев көшесі, 36-үй, 95-пәтер

Оқушылар жайлы мәлімет

1 **Сұрақтарға жауап берейік**

1. Web-сайттағы кесте деген не?
2. Кесте қалай толтырылады?

2 Ойланайық, талқылайық

1. Web-бетте кесте тұрғызу не себепті маңызды?
2. Неліктен кестені құруда тегтердің дұрыс орналасу ережелерін ұстанамыз?

3 Талдап, салыстырайық

1. Төменде келтірілген кесте құру ережелерін талдаңдар.

Кесте құру ережесі	Себебі
1. Деректер түгел көрсетіліп тұрғаны дұрыс	
2. Экранда көрсетілген кестенің мағынасына назар аудару керек	

2. Кесте құру тегтерінің мәтінді форматтау тегтерінен қандай айырмашылығы бар?

4 Дәптерде орындайық

Төмендегі сызбаны өздеріңнің достарың жайлы ақпарат жазу арқылы толтырыңдар.

Достарым жайлы ақпарат

- 1
- 2
- 3
- 4
- 5
- 6
- ...

5 Компьютерде орындайық

1. *Блокнот* программасын ашыңдар.
2. Блокнот редакторында төменде берілген HTML файл мәтінін жазыңдар.
3. HTML файлды сақтаңдар: Файл ⇒ ... ретінде сақтау ⇒ Жұмыс үстелі ⇒ «Менің алғашқы web-сайтым» бумасы ⇒

файлдың атауы «Сабақ кестесі.html» ⇒ Сақтау командаларын орындаңдар.

4. Web-бетті көру үшін желі браузерін қосу керек.

```
<HTML> <HEAD>
<TITLE> Сабақ кестесі </TITLE> </HEAD>
<BODY BGCOLOR = "#FFFFFF">
<P ALIGN = CENTER>
<FONT COLOR = "RED" SIZE = "6" FACE = "KZ
ARIAL">
<B> Сабақ кестесі </B> </FONT> </P> <BR>
<FONT COLOR = "BLUE" SIZE = "4" FACE = "Times
New Roman">
<TABLE BORDER = "1" WIDTH = 100% BGCOLOR = "#99
CCCC">
<TR BGCOLOR = "#CCCCFF" ALIGN = CENTER>
<TD> Уақыты </TD>
<TD> 10 а сыныбы </TD>
<TD> 10 б сыныбы </TD>
<TD> 10 в сыныбы </TD>
</TR> <TR> <TD> 8-30 - 9-50 </TD>
<TD> Орыс тілі </TD>
<TD> Информатика </TD>
<TD> Қазақстан тарихы </TD>
</TR> <TR> <TD> 10-00 - 11-20 </TD>
<TD> Математика </TD>
<TD> Қазақстан тарихы </TD>
<TD> Ағылшын тілі </TD>
<TR> <TD> 11-30 - 12-30 </TD>
<TD> Қазақстан тарихы </TD>
<TD> Алгебра </TD>
<TD> Физика </TD> <TR>
</TABLE>
</BODY>
</ HTML>
```

6 **Ой бөлісейік**

Кесте құратын тегтерді есте сақтай отырып, HTML-да кесте құру жолдары туралы сыныптастарыңмен ой бөлісіңдер.

§ 43. Практикум. Кесте құру

Кесте құру үшін <TABLE> және </TABLE> тегтерін қолданылатынын білесіңдер. Мысалы:

```
<table> <tr> <td> Бірінші ұяшық </td> <td> Екінші
ұяшық </td> </tr> </table>
Бірінші ұяшық                    Екінші ұяшық
```

мұнда кесте жиегі болмайды. Кесте жиегі көріну үшін алдыңғы бөлімде өткен «border» атрибуты қолданылады.

```
<table border = "1">
   <tr>
      <td> Бірінші ұяшық </td>
      <td> Екінші ұяшық </td>
   </tr>
</table>
```

Web-беттерінде төмендегідей көрінеді.

Бірінші ұяшық	Екінші ұяшық

1-тапсырма. Жаңа құжат құрыңдар. Құжатқа төмендегі көрсетілген кесте енгізіп, оның түсін қызылға бояңдар және жиегінің өлшемін 5 деп алыңдар.

Бірінші ұяшық	Екінші ұяшық

2-тапсырма. Төмендегі тегтерді құжатқа енгізіңдер. Браузерде нәтижесін көріп, түсіндіріп беріңдер.

```
<table border = "3" cellpadding = "10" bgcolor =
"#999999">
   <tr>
      <td> Бірінші ұяшық </td> <td> Екінші ұяшық
      </td> <td> Үшінші ұяшық </td>
   </tr>
   <tr>
```

```
        <td> Төртінші ұяшық </td>
        <td bgcolor = "#FF0000"> Бесінші ұяшық
        </td> <td> Алтыншы ұяшық </td>
    </tr>
</table>
```

3-тапсырма. Құжатқа төмендегі кесте үлгісі бойынша енгізіңдер.

Сабақ кестесі					
Дүйсенбі	**Сейсенбі**	**Сәрсенбі**	**Бейсенбі**	**Жұма**	**Сенбі**
Математика	**Қазақ тілі**	Химия	Қазақ тілі	Математика	*Әдебиет*
Информатика	**Химия**	Денешынықтыру	Математика	Информатика	*Физика*
Ағылшын тілі	**Физика**	Физика	Химия	Ағылшын тілі	*Информатика*
Қазақ тілі	**Орыс тілі**	Орыс тілі	Денешынықтыру	Қазақ тілі	*Химия*
Денешынықтыру	**Әдебиет**	Математика	Информатика	Денешынықтыру	*Денешынықтыру*

4-тапсырма. Құрылған кестенің фонына сурет енгізіңдер.

Сабақ кестесі					
Дүйсенбі	**Сейсенбі**	**Сәрсенбі**	**Бейсенбі**	**Жұма**	**Сенбі**
Математика	**Қазақ тілі**	Химия	Қазақ тілі	Математика	*Әдебиет*
Информатика	**Химия**	Денешынықтыру	Математика	Информатика	*Физика*
Ағылшын тілі	**Физика**	Физика	Химия	Ағылшын тілі	*Информатика*
Қазақ тілі	**Орыс тілі**	Орыс тілі	Денешынықтыру	Қазақ тілі	*Химия*
Денешынықтыру	**Әдебиет**	Математика	Информатика	Денешынықтыру	*Денешынықтыру*

§ 44–45. CSS

Естеріңе түсіріңдер:

- *web-құжат дегеніміз не?*
- *гипермәтін дегеніміз не?*
- *гипермедиа дегеніміз не?*

Меңгерілетін білім:

- *CSS туралы түсінік;*
- *CSS-тің HTML-де қолданылуы;*
- *CSS пен HTML-дің айырмашылығы;*
- *стиль түрлері.*

Терминдер:

- CSS;
- стиль;
- қолданушы стилі;
- браузер стилі;
- кіріктірілген стиль;
- байланысқан стиль;
- импортталған стиль.

CSS (Cascading Style Sheets – каскадты стильдер парақтары) – браузердегі деректердің көрсетілуін анықтайтын стандарт. HTML құжат құрылымы туралы ақпарат берсе, ал CSS стильдер парақтары оны қалай бейнелеу керектігін қарастырады. Әдетте HTML мәтіннің түсі мен өлшемін беру үшін пішімдеу тегтері арқылы жүзеге асырады. Сайттағы бірдей элементтердің параметрлерін өзгерту қажет болса, тегтерді табу және өзгерту үшін барлық беттерді шолуға тура келеді. CSS тілі HTML тілінің мүмкіндігін кеңейтеді. HTML құжаттарына қажетті бейнені береді.

Көбінесе HTML және XHTML web-беттерді жобалау құралы ретінде пайдаланылады, сонымен қатар SVG, XUL және XML форматында құжаттардың барлық түрлерімен пайдалануға болады.

Қызықты ақпарат

CSS стильдің каскадты кестелері HTML-ден кейін, 1997 жылы пайда болды. CSS-тің HTML-мен жұмыс жасағанына қарамастан, ол – HTML емес. Сонымен қоса CSS HTML тегтерді анықтай отырып, HTML мүмкіндіктерін кеңейтетін жеке кодты ұсынады. Бұл WWW мәтіндік құжаттарды алмасу тәсілі ретінде, ал HTML осы құжаттар жасалатын тіл болғандығымен байланысты. Әскери қызметкерлер мен ғылымдарды құжаттың көріктілігі емес, оның құрамы қызықтырды. Сондықтан HTML-дің алғашқы жобаларында web-беттерге бейнені қосу әдістері болмаған. Бірақ уақыт өте келе Интернетке басқа да пайдаланушылар келіп, олар өздеріне әсем етіп жасауды талап етті. Осылайша web-дизайнерлердің жұмысын жеңілдету мақсатымен жасалған стильдердің каскадты кестелері пайда болды.

CSS-тің HTML-ден айырмашылығы – HTML сайттың контентінің құрылымын жасау үшін қолданылуында. Ал CSS бұл контентті пішімдеу, көркемдеу үшін, дизайн, стиль беру үшін қолданылады.

CSS-тің пайда болуы, оның артықшылықтары web-дизайн әлемінде төңкеріс болды.

CSS-ті қолдану артықшылықтары:

- Анық бақылау кодтың көлемін едәуір кемітеді және оны оқуға ыңғайлайды;
- CSS тілінің көмегімен HTML тілінде беруге болмайтын параметрлерді ашамыз. Мысалға алатын болсақ, сілтемелердің астындағы сызықты алып тастауға болады;
- CSS арқылы web-беттің сыртқы көрінісін оңай өзгертеміз. Көп құжаттардың сыртқы көрінісін бір кесте арқылы көрсетуге болады. Мысалы, біз 30 бет кодтағы қаріптерді жасыл түс еттік делік. Бірақ уақыт өткен соң, көк не қызыл түске өзгерткіміз келіп, барлық 20 бетке кіріп, керекті атрибуттағы қаріпті өзгертіп шығамыз. Ал CSS тілі арқылы сол 20 беттің барлығын бір ғана стильдер кестесінде өзгертуге болады;
- Құрамалы және жинақталған дизайн техникасы. CSS тілінде сайт беттеуі деген ұғым бар.

Егер сайттағы бірдей элементтердің параметрлерін өзгерту қажет болса, тегтерді табу және өзгерту үшін барлық беттерді шолуға тура келеді. Ал CSS бізге түс, мәтін өлшемдерді және стильдің басқа параметрлерін сақтауға мүмкіндік береді.

Стиль – құжаттың сыртқы келбетін жылдам өзгертуде қолданылатын пішімдеу ережелер жиынтығы.

Стильдер бір әрекетпен пішімдеудің барлық атрибуттар тобын қолдануға мүмкіндік береді. Олардың көмегімен барлық тақырыптардың көрінісін өзгертуге болады. Мысалы, тақырыпты пішімдеу үшін үш әрекетті, яғни бірінші оның өлшемі, содан кейін қарпі және соңында ортаға туралау әрекеті орындалса, стильді қолдану арқылы осы әрекеттерді бір уақытта <H1> тегіне пайдалануға болады.

Егер стильдердің бірімен жасалған мәтін көрінісін жылдам өзгерту керек болса, онда ол қолданылған барлық құжаттардағы

стиль параметрлерін өзгертіп, мәтінді автоматты түрде өзгертуге болады (*12-кесте*).

12-кесте. *CSS тиімділігі*

Құжаттың өлшемін азайтады. Бұл жүктелуді азайтуға мүмкіндік береді
CSS-ті пайдалану стиль сипаттамасымен берілген бір ғана файлды өңдеу арқылы көптеген құжаттарды басқаруға мүмкіндік береді
Стильдер қарапайым HTML-ге қарағанда әлдеқайда көп пішімдеу нұсқаларын ұсынады
CSS құжаттарын кэштеу

Стиль кестелерін қолдану түрлері (*13-кесте*):

13-кесте. *Стиль кестелерін қолдану түрлері*

Браузер стилі	Бұл – стандартты стиль парағы. Егер белгілі стиль анықталып, берілмесе, онда осы стандарттар қолданылады
Қолданушы стилі	Кез келген қолданушы браузер параметрін өзгерту арқылы өзінің стиль кестесін жаза алады және қолдана алады
Ендірілген стиль	Style атрибутын қолдану арқылы анықталады
Кіріктірілген стиль	HTML құжатының ішінде орналасады
Байланысқан стиль	Құжатпен link элементінің көмегімен байланысады
Импортталған стиль	Стильдерді импорттайды

HTML-ге қарағанда CSS-тің синтаксисі күрделірек. Себебі HTML-дің бірнеше тегін қалай пайдалану керектігін ұғып алған соң, еш қиналмастан парақша бетін жаза беруімізге болады. Ал CSS – стильдің анықтамасы, оның әр сипаттамасын ғана емес, оны қай жерде қалай қолдануға болатынын да үйреніп алу керек.

Бұл стильді белгі тілі екі бөліктен құралады:

1) селектор-белгілерден;
2) осы селекторларға қолданылатын ережелерден.

HTML тілімен бірге CSS-ті қолданудың үш түрі бар. CSS стильдер кодын осы парақ басында <link> арқылы сипатталып, жүктелетін бөлек файлға жазып қою:

1-мысал. Қарапайым HTML-бет жасап, оған мынадай код салыңдар:

```
body{
background: blue;
color: white;
}
h1{
color:red;
}
h2{
color:yellow;
}
```

Енді блокноттан жаңа парақша ашып, оның атын ***style.css*** деп жазып, **HTML**-бет орнатылған бумаға сақтаңдар (*23-сурет*):

***23-сурет. style.css** файлын сақтау*

Ол біздің стильдеріміздің парағы болады. Енді біз *style.css* парағын HTML-бетке жалғауымыз керек. Ол үшін HTML-де сыртқы файлдарды жалғауға жауап беруші <link> тегі бар. Осы тегті біздің HTML-бетімізге қосамыз:

```
<html>
<head>
<title> HTML-ге CSS жалғау </title>
<link rel = "stylesheet" type = "text/css" href =
"style.css">
```

```
</head>
<body>
<h1> Мынау – бірінші деңгейлі тақырып </h1>
Бұл жер – мәтіннің орны
<h2> Мынау – екінші деңгейлі тақырып </h2>
Бұл жер – мәтіннің орны
</body>
</html>
```

2-мысал. Стильдік класты тегпен сәйкестендіру үшін class атрибуты қолданылады:

```
<P class = def> текст </P>
Бұл кодтың қалай жұмыс істейтінін қарастырайық.
<html>
<head>
<title> Элемент бойынша селекторлар </title>
<link rel = "stylesheet" type = "text/css" href =
"style.css">
</head>
<body>
<P class = def>
Класс &#151; Кластың атауы қойылар алдында
нүкте қойылады
</body>
</html>
style.css
.def
   {font-family:Helvetica;font-size:14pt;
   border: solid 4pt red;padding: 6pt;
   margin-left:5%;margin-right:5% }
```

Қазір біздің HTML-бет мынадай көрініс**те:**

Класс – Кластың атауы қойылар алдында нүкте қойылады

3-мысал. <DIV> және <SPAN> тегтері:

Бұл тегтер CSS үшін маңызды болып табылады. Олар құжаттағы жеке бір бөліктерді ерекшелеп алып, оларға арнайы

қасиеттер беру ісін атқарады. Ол үшін керекті элементтерді <DIV> … </DIV> немесе <SPAN> … </SPAN> тегтері ішіне орналастыру керек.

Бұлардың айырмашылығы мынада: <DIV> блогынан соң браузерді жаңа жолға көшіреміз, ал <SPAN> блогынан кейін бұрынғы жолда қала береміз. Сонымен, <SPAN> тегін пайдалану – бір жолдағы сөздерге не символдарға жеке стильдік қасиеттер тағайындай алады. Осыларға мысал келтірейік.

a) <DIV> тегін пайдалану:

```
<html>
<head>
<title> Элемент бойынша селекторлар </title>
<link rel = "stylesheet" type = "text/css" href =
"style.css">
</head>
<BODY bgcolor = white text = black>
   <DIV class = area1> Еңбектің наны тәтті </DIV>
   <DIV class = area2> Жалқаудың жаны тәтті </DIV>
   </DIV>
</BODY>
</html>
```

style.css

```
.area1
{color:red; font-weight:bolder;
font-size:40pt; background:aqua}
.area2
{color:black; background:#CFB597}
.area3
{color:blue;background:#C0C0C0}
```

Қазір біздің HTML-бет мынадай көріністе:

Еңбектің наны тәтті

Жалқаудың жаны тәтті

b)

```
<html>
<head>
<title> Элемент бойынша селекторлар </title>
```

```
   <link rel = "stylesheet" type = "text/css" href =
   "style.css">
   </head>
   <BODY bgcolor = white text = black>
      <SPAN class = area1> Еңбектің наны тәтті
      </SPAN>
      <SPAN class = area2> Жалқаудың жаны тәтті </
      SPAN>
      </SPAN>
   </BODY>
   </html>
style.css
   .area1
      {color:red; font-weight:bolder;
      font-size:40pt; background:aqua}
   .area2
   {color:maroon; background:#CFB597;
   padding:6pt}
```

Қазір біздің HTML-бет мынадай көріністе:

Еңбектің наны тәтті **Жалқаудың жаны тәтті**

4-мысал. Идентификаторлар мен кластарды HTML-дің кез келген элементімен беруге болады. Бірақ көбінесе былай болады: әдетте түрлі элементтерді бір ғана стильмен, мысалы, жасыл түспен белгілегіміз келеді. Осындай жағдайда бірыңғайланған селекторды қолдануға болады. Ондай селекторларда элемент аты көрсетілмейді, класс немесе идентификатор және атау белгісі ретінде нүкте немесе тор көрсетіледі.

Мысалы:

```
.red{
color:red;
}
#yellow{
color:yellow;
}
```

Сөйтіп, қандай элементке (тақырып, абзац не сілтемеге) `class = "red"` берсек те, мәтіннің түсі қызыл болады. Тек бір элементке `id = "yellow"` бере аламыз және бұл элемент мәтінінің түсі сары болады.

Тікелей HTML тегінің ішінде «style» атрибутын орналастыру:

5-мысал:

```
<html>
<head>
<title> CSS id </title>
</head>
<body>
<p style = "background:#00ff00;color:red;">
Сәлем, CSS ережесі бойынша менің әріптерім қызылға,
айналам жасыл түске боялады </p>
</body>
</html>
```

Қазір біздің HTML-бет мынадай көріністе:

Сәлем, CSS ережесі бойынша менің әріптерім қызылға, айналам жасыл түске боялады

Web-беттің басында, яғни <head> арасына </head> «style» элементін жазу:

6-мысал:

```
<html>
<head>
<title> CSS id </title>
<style type = "text/css"> p{background:#00ff00;co
lor:red;} </style>
</head>
<body>
<p> <b> <u> Егер жеке файлда әртүрлі бетті бір-
дей стильдермен рәсімдейтін болсақ, стильдер CSS
кеңейтілімімен берілген мәтіндік файлға жазы-
лады <u> </b> </p>
```

```
</body>
</html>
```

Қазір біздің HTML-бет мынадай көріністе:

7-мысал. HTML-де элементтің идентификаторы id параметрі арқылы беріліп, оның мәнісі ретінде бірегей (қайталанбайтын) атау беріледі. Мысалы:

```
<p id = "pink"> Идентификаторлық мәтін абзацы (id). </p>
```

Атауға HTML және CSS-тегі тег, параметр және элемент аттарынан басқа кез келген сөзді қоюға болады. Мысалы, идентификаторға body деген ат қоюға болмайды. Енді HTML-бетімізге екі абзац қосып, олардың біріне идентификатор қосамыз:

```
<html>
<head>
<title> CSS id </title>
<link rel = "stylesheet" type = "text/css" href =
"style.css">
</head>
<body>
<h1> Бірінші деңгейлі тақырып </h1>
Бұл жерде жай тақырып
<h2> Екінші деңгейлі тақырып </h2>
Бұл жерде жай тақырып
<p> Жай абзац </p>
<p id = "pink"> Идентификаторлы (id) абзац
</p>
</body>
</html>
```

Қазір браузердегі бетімізге қарайтын болсақ, олардың екеуі де ақ түсте. Стильдер кестесіне (style.css) абзацтар үшін стильдер қосайық:

```
body{
background: blue;
color: white;
}
h1{
color:red;
}
h2{
color:yellow;
}
p{
color:black;
}
p#pink{
color:pink;
}
```

Басында мәтіндегі барлық абзацты қара түсте көрсеттік, бірақ `id = "pink"` пен жазылған абзац мәтіні қызғылт түсте болды. Бұл жағдайда селекторымыз (p) элементінен, бөлгіш (#) және идентификатор аты (pink) элементтерінен тұрады.

Айта кететін жайт, бір бетте тек қана бір (id) идентификатор бола алады. Демек, біздің мысалымызда `id = "pink"` деген екі абзац жасай алмаймыз, id деген абзац тек қана біреу болу керек (себебі id *бірегей, қайталанбайтын* дегенді білдіріп тұр). Дегенмен әрбір абзацтың өз идентификаторы болады, `id = "green"` деген абзац жасап, оған стильді кестелер ішінде стиль бере аламыз.

8-мысал. Жоғарыда айтып өткен мысалымызда қызғылт түсті мәтінде абзац жасап, ондай id тек қана біреу болады деп көрсеттік. Ал егер біз екі не одан да көп абзацта қызғылт түсті мәтін болуын қаласақ, не болады? Ол үшін HTML-де class деген параметр бар, class-тың мәнісі ретінде соның атауы көрсетіледі.

HTML-бетке тағы да екі абзац қосып, оларға class = "pink" қосамыз:

```
<html>
<head>
<title> CSS class </title>
<link rel = "stylesheet" type = "text/css" href =
"style.css">
</head>
<body>
<h1> Бұл – бірінші деңгейлі тақырып </h1>
Бұл жерде жай мәтін
<h2> Бұл – екінші деңгейлі тақырып </h2>
Бұл жерде жай мәтін
<p> Жай абзац </p>
<pid = "pink"> Идентификаторлы абзац </p>
<p class = "pink"> (class) pink класы мен абзац
</p>
<p class = "pink"> (class) pink класы мен абзац
</p>
</body>
</html>
```

Бұл класқа стиль көрсету үшін стильдер кестесінде ереже қосамыз. Ережеде селектор ретінде элемент және атау ретінде pink қолданылады. Бірақ бұл жағдайда pink кластың атауы болғандықтан, бөлгіш ретінде нүкте (.) белгісі қолданылады:

```
body{
background: blue;
color: white;
}
h1{
color:red;
}
h2{
color:yellow;
}
p{
color:black;
```

```
}
p#pink{
color:pink;
}
p.pink{
color:pink;
}
```

Бұндай класпен абзацтарды қалағанымызша жасауға болады.

Осы аралықтағы қорытындыны шығарсақ:

- Егер барлық бірдей (ұқсас) элементтерге (мысалы, барлық h1 тақырыптарға) бір стиль қою керек болса, онда селектор тек қана мына элементтен тұрады.Мысалы: p{color:black;})
- Егер де элемент (кез келген: абзац, тақырып және т.б.) басқаларынан ерекшеленуі қажет болса, онда оған (id) идентификаторы қосылып, стильдер кестесіндегі бөлгіш ретінде тор (#) белгісі қолданылады. Мысалы: p#pink{color:pink;}
- Егер бетте бірдей стильдегі бірнеше элемент болса, онда оларға (class) класы қосылып, стильдер кестесіндегі бөлгіш нүкте (.) белгісі болады. Мысалы: p.pink{color:pink;}
- Идентификатордың класқа қарағанда біраз артықшылығы бар. Сондықтан қандай да бір элемент үшін класс та, идентификатор да көрсетілсе, онда идентификатор стилі қолданылады.

9-мысал. Бізде мынадай кодпен HTML-бет бар делік:

```
<html>
<head>
<title> Элемент бойынша селекторлар </title>
<link rel = "stylesheet" type = "text/css" href =
"style.css">
</head>
<body>
<p> Бұл мәтін абзацта орналасқан. </p>
Бұл – жай мәтін.
<i> Бұл мәтін көлбеу әріппен белгіленген. </i>
<p> Бұл мәтін абзацта орналасқан, бірақ <i> бұл
бөлігі көлбеу әріппен белгіленген. </i> </p>
</body>
</html>
```

Мәтінде көлбеу әріптің белгіленуін қалап, оны сары түске бояймыз. Сонда біз стильдер кестесіне элемент бойынша селектор жазамыз:

```
i{
color:green;
}
```

Қазір біздің HTML-бет мынадай көріністе:

Бұл мәтін абзацта орналасқан.
Бұл – жай мәтін. *Бұл мәтін көлбеу әріппен белгіленген.*
Бұл мәтін азат жолда орналасқан, *бірақ бұл бөлігі көлбеу әріппен белгіленген.*

1 Сұрақтарға жауап берейік

1. CSS деген не?
2. CSS қашан пайда болды?
3. Стиль деген не?
4. CSS-ті қолданудың неше түрі бар?

2 Ойланайық, талқылайық

1. HTML-де CSS қолданудың маңыздылығы неде?
2. Неліктен web-дизайн әлемінде CSS-ті қолдану тиімді?
3. Не себепті HTML-ге қарағанда CSS-тің синтаксисі күрделірек?

3 Талдап, салыстырайық

1. CSS пен HTML-дің айырмашылықтары қандай?
2. Ассоциациялық аймақ жасаңдар.

4 Дәптерде орындайық

Берілген кестені толтырыңдар.

HTML тиімділігі	CSS тиімділігі

5 Компьютерде орындайық

а) «Менің хоббиім» туралы web-бет құрыңдар. Web-бет ортаға түзетілген «Менің хоббиім» тақырыбынан, өздерің туралы қысқаша сипаттамадан және өздеріңнің қызығушылықтарың туралы (спорт, ән айту, билеу және т.б.) тізімнен тұруы керек.

ә) Сандық нөмірленуді әріппен және рим цифрларымен нөмірленуге өзгертіңдер.

б) Әртүрлі маркерді пайдаланып, маркерлену тізімін өзгертіңдер.

6 Ой бөлісейік

1. Қалай ойлайсыңдар, web-беттерді жасауда CSS қолдану және онымен үйлесімді жұмыс істеу қаншалықты маңызды? Өз ойларыңмен достарыңмен бөлісіңдер.
2. Интернетті пайдаланып, CSS-ті қолдану артықшылықтары мен кемшіліктеріне байланысты постер құрып, түсіндіріңдер.

§ 46–47. Мультимедианы енгізу

Естеріңе түсіріңдер:

- *CSS дегеніміз не?*
- *CSS кестелері не үшін қолданылады?*

Меңгерілетін білім:

- *мультимедиа;*
- *мультимедиалық тегтердің түрлері;*
- *web-бетке аудио және бейне файлдарды орналастыру.*

Терминдер:

- мультимедиа;
- фрейм;
- аудио және бейне.

Мультимедиа – мәтін, графика, анимация, сандық бейнелерді, бейне, дыбыс сияқты деректер типін енгізуге, өңдеуге, сақтауға, беруге және бейнелеуге мүмкіндік беретін технологиялар жиынтығы.

Web-беттердің басқа web-беттерге қатысты сілтеменің болуы – World Wide Web жүйелерінің ең тартымды ерекшеліктерінің бірі. HTML құжатында гипермәтіндік сілтемелерді құру өте жеңіл. Ол үшін атрибуты, яғни параметрі бар ашылатын <A...> және қарапайым жазылатын </A> тегтері пайдаланылады. Жалпы сілтемелер жасаған кезде мынадай ережелерді есте сақтаған жөн.

Фреймдер. Фрейм құрылымы. HTML тілі браузер программасы терезелерін бірнеше бөлікке бөліп тастау мүмкіндігін береді және олардың әрқайсысында жеке құжаттар бейнеленеді. Осындай бөліктерді ***фрейм*** деп атаймыз.

1) Фреймдерді құру үшін HTML-дің ерекше құжаты пайдаланылады, оның құрылымы кәдімгі құжаттардан бөлек болады. Осындай құжаттарда құжат «денесінің» бөлімдері болмайды, ол шын мәнінде қандай да болсын мәтінде мүлде болмайды. Оның орнына бұл құжаттарда **<FRAMESET>** және **</FRAMESET>** тегтерінің арасында орналасқан фреймдер болады.

2) <FRAMESET> тегі, терезелерді бөлу әдістерін анықтайтын атрибуттар:

- COLS = атрибуттарын пайдаланған уақытта терезелер вертикаль сызықтармен;
- ROWS = атрибутын пайдаланған уақытта горизонталь сызықтармен бөліктерге бөлінеді.

Осы атрибуттардың мәндері терезе бөліктерінің биіктігін (немесе енін) анықтайды. Әр бағанға (жолға) арналған параметрлер пиксель өлшем бірлігі бойынша үтірлер арқылы немесе процентпен (% белгісі) беріледі.

3) <FRAMESET> және </FRAMESET> тегтерінің арасында қалыптасқан бөліктердің қажеттілігін көрсететін қосымша тегтер орналастырылады. Осы мақсаттар үшін терезені қосымша бөлу мүмкіндігін беретін, ендірілген <FRAMESET> тегін немесе экрандағы жеке бөліктеріне шақырылатын құжаттарды анықтайтын, жеке-даралық <FRAME> тегтерін пайдалануға болады.

<FRAMESET> және </FRAMESET> тегтерінің араларына орналастырылған элементтердің саны қалыптастырылған бөлік санына сәйкес болуы керек.

4) <FRAME> тегінде аталмыш бөлікке шақырылатын құжаттарды анықтайтын SRC = міндетті атрибуттары болуы керек. Қосымша атрибуттар және фреймдер арасындағы қоршауларды және оның басқа кейбір қасиеттерін реттеу мүмкіндігін береді.

1-мысал:

```
 <HTML>
<HEAD>
<TITLE> жаңалықтар
   </TITLE>
      </HEAD>
         <FRAMESET>
<COLS = "25%">
<FRAME SRC = panel.htm>
<FRAME SRC = home1.htm>
         </FRAMESET>
               </HTML>
```

Web-беттерге графиканы қосу. Қазіргі кезде web-браузерлер барлық кескін пішімдерін қолдамайды, сондықтан әр суретті web-бетке орналастыруға болмайды. Растрлық және векторлық суреттер пішімдеріне қарай бөлінеді. Bitmap кескіндері jpg, gif, bmp, tiff, png, psd кеңейтілімдері бар файлда ғана сақталады. Осылардың ішінде web-беттерін өңдеуде растрлік графика үшін *JPEG, PNG, GIF* және векторлық графика үшін *SWF* қолданылады.

Мультимедиалық ақпарат құрылымы мәтін құрылымынан түбегейлі ерекшеленеді, сондықтан мультимедиа тікелей HTML-кодында сипатталмайды. Әзірлеушіге мультимедиаға

қажеттінің барлығы жеке файлдарда сақталады. Оларға сәйкес сілтемелері *HTML*-кодында жазылады.

HTML көмегімен сурет қою үшін міндетті src атрибутынан тұратын жұпсыз тег **<*img*>** қолданылады.

<img>*-дің міндетті емес атрибуттар қатарына жататындар:*

- align – суретті түзету типін береді;
- alt – егер сурет жүктелмеген жағдайда мәтін шығарады;
- border – сурет жақтауларының қалыңдығын анықтайды;
- height – суреттің биіктігін береді;
- hspace – суреттен көлденең шегіну арақашықтығын береді;
- ismap – суреттің карта екенін анықтайды;
- vspace – суреттен тігінен шегіну арақашықтығын береді;
- width – суреттің енін береді;
- usemap – клиенттік карта – суреттердің координаттары бар <map> сілтемесін анықтайды;
- <img> – кіріктірілген тег болып табылады, яғни оны блоктың сыртында пайдалануға болмайды.

**Синтаксисі: <img src = "sample.jpg"> **

Бейне және дыбыспен жұмыс істеу негіздері. HTML спецификациясы тиісінше аудио және бейнемен жұмыс істеу үшін екі тегті қамтамасыз етеді: <audio> және <video>.

Бұл тегтер браузердің өз ортасының құрамдас бөлігі болып табылады. Демек, мультимедиа ақпараты көбінесе қауіпсіздікті арттыру үшін қолданылады. Екіншіден, тығыз интеграцияға байланысты мультимедиа ойнату үшін аздаған аппараттық ресурстарға мүмкіндік береді және үшіншіден, ақпараттық дисплейдің көптеген проблемаларын болдырмайды. Сонымен қатар <audio> және <video> пайдалану арқылы web-сценарийлерден басқаруды ұйымдастыруға мүмкіндік береді. Тегтердің сонымен бірге кемшіліктері де бар.

Кодтау мәселесінің жартылай шешімі ретінде браузер таңдаған бірнеше медиа дереккөздерін жариялауға мүмкіндік беретін <source> элементі пайдаланылады.

2-мысал:

```
<audio>
<source src = "sound1.ogg">
<source src = "sound1.mp3">
</audio>
```

Аудио және бейнежазбаларды қою. HTML-де аудиожазбаларды қою үшін <audio> жұпты тегі қолданылады. Осы тегтің арасында орналасқандар <audio> қолдамайтын браузерде көрінбейтін болады.

Аудионы қоюдың негізгі коды:

```
<audio src = "sound1.mp3"> </audio>
```

Немесе әртүрлі браузерлердің әмбебап өңдеуін қамтамасыз ету үшін:

```
<audio>
<source src = "sound1.ogg">
<source src = "sound1.mp3">
</audio>
```

<audio> тегі мынадай атрибуттардан тұрады:

- autoplay – оны қосқанда бет жүктелгеннен кейін файл бірден ойнай бастайды;
- controls – аудиоға басқару тақтасын қосады;
- loop – бейнені басынан бастап қайталау, ол аяқталғаннан кейін орындалады;
- preload – беттің өзін жүктеумен бірге файлды жүктеу үшін қолданылады, ал autoplay қолданылса, тоқтатылады;
- src – ойнату үшін файлдың жолын анықтайды.

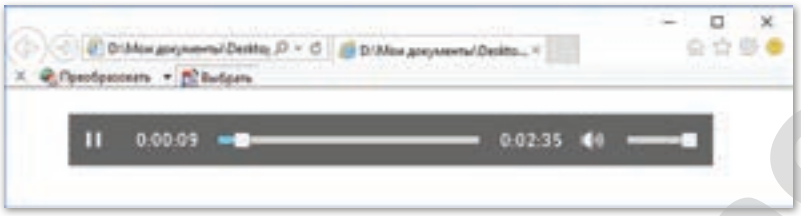
3-мысал:

```
<audio autoplay controls src = "1.mp3">
Тег <audio> қолдамайды
</audio>
```

Нәтиже (*24-сурет*):

24-сурет. *Программа нәтижесі*

Егер браузер көрсетілген тег болмай шықса, <audio> тегті қосу нәтижесі (*25-сурет*).



25-сурет. <audio> *тегін қосу нәтижесі*

Бейне былайша қосылады:

```
<video src = "video1.avi"> </video>
```

<video> *тегінің атрибуттары:*

- autoplay – оны қосқанда бет жүктелгеннен кейін файл бірден ойнай бастайды;
- controls – бейнеге басқару панелін қосады;
- height – бейнені ойнату үшін аймақтың биіктігін береді;
- loop – аудионы басынан бастап ойнатып, аяқталғаннан кейін бірден қайталанады;
- poster – бейнежазба ойнатылмаған кезде немесе қолжетімді емес жағдайда суретке жол көрсетеді;
- src – ойнату үшін файлға жол көрсетеді;
- width – бейнежазбаны ойнату аймағының көлемін көрсетеді.

1 Сұрақтарға жауап берейік

1. Мультимедиа деген не?
2. Фрейм қайда орналасады?
3. <audio> тегінің қандай атрибуттары бар?
4. <video> тегінің атрибуттары қандай?

2 Ойланайық, талқылайық

1. Не себепті HTML құжатында гипермәтіндік сілтемелерді құру жеңіл?
2. Неліктен әр суретті web-бетке орналастыруға болмайды?
3. Не себепті мультимедиаға қажетті материалдар жеке файлда сақталады?
4. Мультимедиа не үшін қолданылады?

3 **Талдап, салыстырайық**

Web-бетке мультимедиа орналастыру тегтерін салыстырып, талдап, кестеге толтырыңдар.

Аудио	Бейне
1.	1.
2.	2.
…	…

4 **Дәптерде орындайық**

Сурет, аудио, бейне файлдарды web-бетке орналастыруы бойынша төмендегі сызбаны толтырыңдар.

5 **Компьютерде орындайық**

Өз өмірбаяндарың жазылған web-бет жасаңдар. Web-беттегі деректер орындалатын талаптар:

1. Мәтінді өңдеу тегтерін пайдаланып, өмірбаян тақырыбы құжаттың ортасында, ал мәтін ені бойынша туралансын.
2. Web-бетке өздеріңнің суреттеріңді қойыңдар және сурет мәтіннің астында, ортада орналассын.
3. Өздеріңнің сүйікті музыкаларыңды web-бетке қосыңдар.
4. Өз өмірлеріңдегі қызықты сәт бейнесін web-бетке қосыңдар.

6 **Ой бөлісейік**

Компьютерде жасаған web-беттеріңді талқылап, ой бөлісіңдер. Қандай қиыншылықтар болды?

§ 48. Практикум. Мультимедианы енгізу

HTML-құжат ішіне сурет енгізіп көрейік. HTML-құжаттың сурет қойылатын жеріне тегті жазып, сол жердегі *файл аты* сурет атына (URL-ына) алмастырылуы керек.

```
<IMG SRC = "файл аты"> нақты файл үшін <IMG SRC =
"dog1.gif">
```

Бұл жердегі SRC – міндетті түрде жазылатын атрибут. Оның мәні ретінде суретке баратын жол мен файл аты көрсетіледі. Егер файл осы бумада болса, жол көрсетілмейді. Сурет үшін GIF, JPEG немесе PNG форматты файлдар қолданылады.

Осының нәтижесінде браузер суретті көрсетілген жерге, оның алдындағы мәтіннің немесе басқа нысанның оң жағына орналастырады.

Төмендегі мысалға назар аударыңдар. Бұл мысалда бір сурет көрсетіледі. Үш ретте де сурет жолда көрсетіледі, сондықтан браузер оны алдындағы мәтіннің оң жағына орналастырады.

```
<HTML> <HEAD> <TITLE> IMG тегін қолдану </TITLE>
</HEAD>
<BODY>
<P> Мұнда сурет мәтіннен кейін пайда болады. <IMG
SRC = "kz.jpg"> </P>
</BODY>
</HTML>
```

1-тапсырма.

Жаңа құжат құрыңдар. Құжатқа сурет қойып, height – суреттің биіктігін және width – суреттің енін, көлемін өзгертіңдер.

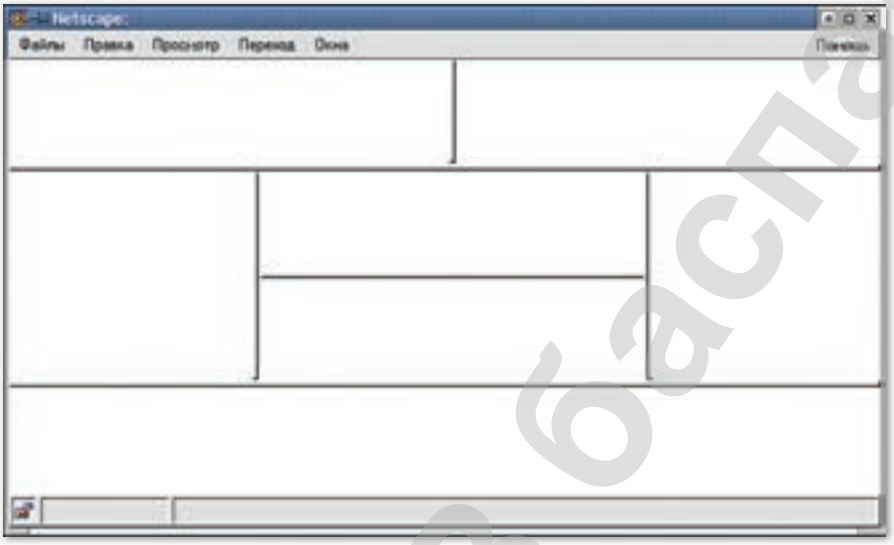
```
<p> <img src = "kz.jpg" width = "..." height =
"..."> </p>
```

2-тапсырма.

Қойылған суретке `Align = left, Align = right` атрибуттарын енгізіңдер. Браузерде нәтижесін көріп, түсіндіріп беріңдер.

3-тапсырма.

FRAMESET тегі арқылы браузер терезесін төмендегі суреттегідей кадрларға бөліңдер. Оларға суреттер (файлдар) орналастырыңдар.



4-тапсырма.

Құжатқа аудиожазбаларды қою үшін `<audio>` жұпты тегі қолданып, оны қосқанда бет жүктелгеннен кейін файл бірден ойнай бастайтын `autoplay` атрибутын пайдаланыңдар.

5-тапсырма.

Құжатқа таңдалған бейне файлды енгізіңдер және бейнежазбаны ойнату аймағының көлемін `width` атрибуты арқылы өзгертіңдер.

§ 49–50. Скрипттер пайдалану

Естеріңе түсіріңдер:

- *мультимедиа дегеніміз не?*
- *<audio> тегінің қандай атрибуттары бар?*
- *<video> тегінің атрибуттары қандай?*

Меңгерілетін білім:

- *JavaScript тілі;*
- *нысан.*

Терминдер:

- скрипттер;
- JavaScript.

«Скрипт» термині – құжатты жүктеу кезінде немесе кейінірек орындалуы мүмкін клиенттік сценарийлерді қамтитын HTML түзету тілі.

JavaScript – нысанды бағытталған, императивті және функционалды стильдерді қабылдай алатын мультипарадигмалық программалау тілі.

Нысан – деректер мен функциялар жиынынан тұратын бірыңғай конструкция немесе JavaScript терминологиясындағы қасиеттер мен тәсілдер жиыны.

JavaScript, әдетте, программалық жасақтаманың нысандарына қол жеткізу үшін енгізілген тіл ретінде пайдаланылады. Бұл тіл браузерлерде web-беттерді интерактивті ету үшін сценарий тілі ретінде кеңінен қолданылады.

JavaScript программасын жасау үшін ешқандай қосымша құралдар қажет емес, тек сәйкес версиядағы JavaScript тілін көтере алатын және HTML құжаттамаларды құруға мүмкіндік беретін мәтіндік редактор ғана қажет. Программа JavaScript-тің тікелей HTML-құжаттама мәтінінде тұрғызылатын болғандықтан, біз өзіміздің жұмысымыздың нәтижесін құжаттаманы браузермен қарау кезінде көре аламыз және қажет болғанда өзгерістер енгізуге мүмкіндігіміз болады.

Негізгі сәулеттік ерекшеліктері:

- динамикалық теру;
- әлсіз теру;
- жадыны автоматты басқару;
- прототипті программалау бірінші санатты нысандар ретінде жұмыс істейді.

JavaScript-ге көптеген тілдер әсер еткендіктен, программалаушы маман болмаса да оны қолдану оңай.

JavaScript – нысанға бағытталған тіл, бірақ тілде қолданылатын прототиптердің дәстүрлі нысанға бағытталған тілге қарағанда төмендегідей айырмашылығы бар:

- бірінші санатты функциялар;
- тізімге ұқсас функциялар;
- анонимді функциялар;
- сілтеме жасаушы.

Бұл қасиеттер тілге қосымша икемділік береді.

JavaScript шамамен жүргізілетін математикалық есептеулерді орындауға мүмкіндік береді. Сондай-ақ бұл тілде күн және уақыттың мәндеріне ие дамыған жұмыс құралдары бар. JavaScript, негізінен, CGI программаларына балама ретінде және Perl сценарийі тіліне баламалар ретінде, сонымен қатар Java тілдеріне қосымша толықтырулар ретінде құрылған.

Web-беттерде орналасуы

Web-беттің ішінде орналасуы

Бетке JavaScript кодын қосу үшін `<script></script>` тегтерін пайдалануға болады, бірақ `<head>` контейнерінің ішіне орналастырылуы міндетті емес. Бір құжатта `<script>` контейнерлері көп болуы мүмкін. «`type = 'text / javascript'`» атрибутын әрқайсысына жеке-жеке пайдалану міндетті емес, бұл мән үнсіздік келісім бойынша қолданылады.

1-мысал.

Браузерде: «Hello, World!» классикалық жазуы бар модульді терезені бейнелейтін сценарий.

```
<html>
<head>
<title> web-беттің ішінде орналасуы </title>
</head>
<body>
<script type = "application/javascript">
alert('Hello, World!');
</script>
</body>
</html>
```

Тегтің ішінде орналасуы

2-мысал.

HTML спецификациясы оқиғалар өңдегіштерін орнату үшін пайдаланылатын атрибуттар жиынтығын сипаттайды. *Мысалы:*

```
<a href = "delete.php" onclick = "return confirm('Сіз сенімдісіз бе?');">
Жою
</a>
```

3-мысал.

JavaScript кодын пайдалану контекстінде түзету бетінде:

```
window.onload = function() {
var linkWithAlert = document.getElementById
("alertLink");
linkWithAlert.onclick = function() {
return confirm('Сіз сенімдісіз бе?');
};
};
```

Жеке файлға өту

4-мысал.

JavaScript-ті қосудың үшінші мүмкіндігі бар – сценарийді бөлек файлға жазыңдар да, оны құрылыстың көмегімен қосыңдар.

```
<head>
<script type = "application/javascript" src =
"http://Файлға_жол!!!">
</script>
</head>
```

1 **Сұрақтарға жауап берейік**

1. Скрипт деген не?
2. Нысанды қалай түсінесіңдер?
3. JavaScript-тің қандай мүмкіндіктері бар?

2 Ойланайық, талқылайық

1. Неліктен web-бетке скрипт қосу маңызды болып табылады?
2. Web-бетке JavaScript файлдарды қосудың қажеттілігі неде?

3 Талдап, салыстырайық

CSS пен JavaScript мүмкіндіктерін салыстырыңдар.

CSS мүмкіндіктері	JavaScript мүмкіндіктері

4 Дәптерде орындайық

Web-бетке орналастыруды талдаңдар және төмендегі кестені толтырыңдар.

Web-бет	Тег	Файл

5 Компьютерде орындайық

Еркін тақырыпқа хабарлама жасаңдар. Web-беттегі директер төмендегі талаптарды орындау керек:

1. Мәтінді өңдеу тегтерін пайдаланып, тақырыбы құжаттың ортасында, ал мәтін ені бойынша туралансын.
2. Web-бетке өздеріңнің суреттеріңді қойыңдар және сурет мәтіннің астындағы орында тұрсын.
3. Өздеріңнің хабарламаларыңа JavaScript қосыңдар.

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Web-бетке JavaScript қосудың қай әдісін қолданар едіңдер? Неліктен?

§ 51–52. Практикум. Скрипттер пайдалану

Программаларды JavaScript тіліне туралау үшін, <script> және </script> тегтері пайдаланылады. Мұнда, негізінен, жұмыс нәтижесін бірден көруге болады және қажет болған жағдайда өзгерістер енгізе аламыз.

```
<html>
<head>
<script language = "JavaScript">
document.write("Hello,world! <p>")
</script>
</head>
<body>
<H3> Менің алғашқы скрипт жасауым!!! <H3>
</body>
</html>
```

Иттің суреті dog.jpg орналасқан жеке терезе ашу

```
<HTML>
   <HEAD>
      <TITLE> Браузерді тексеру </TITLE>
   </HEAD>
   <BODY bgcolor = white text = black>
      <H2> Браузерді тексеру </H2> <HR>
      <SCRIPT language = JavaScript>
      <!--
      var win = open("slon.jpg"," ",
         "width = 320,height = 260" +
         "resizable = 0,scrollbars = 1" +
            "menubar = 0,location = 1" +
            "status = 0, toolbar = no");
      //-->
      </SCRIPT>
      <P>
```

Негізгі мәтінге оралу үшін браузердің саймандар тақтасында орналасқан <EM> Артқа қарай (Назад) </EM> батырмасын басыңдар.

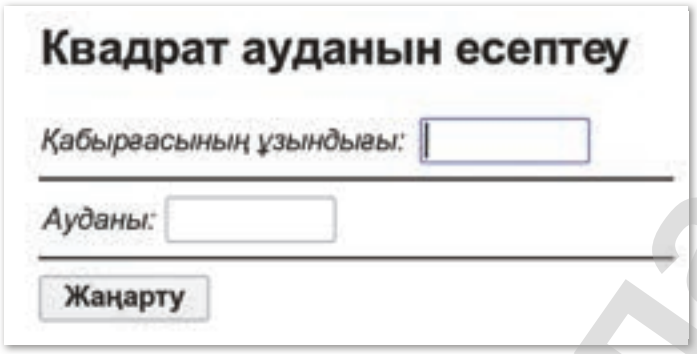
```
</BODY>
</HTML>
```

HTML-код арқылы браузер салған батырма жасау

```
<FORM>
<INPUT type = button
value = "Кәдімгі батырма">
</FORM>
```

Квадрат ауданын есептеу

```
<HTML>
<HEAD>
<title> Change - элемент мәнін өзгерту оқиғасы </title>
<script>
function srec(obj)
{obj.res.value = obj.numl.value* obj.numl.value}
</script>
</HEAD>
<BODY>
<h2> Квадрат ауданын есептеу </h2>
<FORM name = "forml">
Қабырғасының ұзындығы: <input type = "text" size =
7 name = "numl"
onChange = "srec(forml)">
<hr>
Аудан: <input type = "text" size = 7 name = "res"> <hr>
<input type = "reset" value = Жаңарту>
</FORM>
</BODY>
</HTML>
```



Сандардың қосындысын шығаратын скрипт жазу

```
<HTML>
   <HEAD>
      <TITLE> while командасымен тәжірибе </TITLE>
   </HEAD>
   <BODY bgcolor = white text = black>
      <H2> \ while командасымен тәжірибе </H2>
      <HR>
   <SCRIPT language = JavaScript>
   <!--
         var i = 1;
         var sum = 0;
         while(i < = 10)
         {sum + = i; i + + ;}
         alert("Сомасы 1 + 2 + ... + 10 =" + sum);
      //-->
      </SCRIPT>
   </BODY>
</HTML>
```

1-тапсырма. HTML коды арқылы браузер салған батырма жасаңдар.

2-тапсырма. Төрт түлік малдың суреті бар жеке терезе ашыңдар.

3-тапсырма. JavaScript скриптерінің көмегімен, форма құрып, квадраттың периметріне есептеу жүргізіңдер.

4-тапсырма. Экранға «Сәлем!» сөзі бар alert терезесін шығарыңдар.

5-тапсырма. Onclick атрибуты браузерге батырма басылғанда бүгінгі күн, ай, жылды және ағымдағы уақытты көрсетіңдер.

6-тапсырма. «Сенің браузерің JavaScript қабылдайды!!!!!!!» деген хабарламасы бар скриптерді экран бетіне шығарыңдар.

7-тапсырма. 5 пен 13 аралығындағы сандардың қосындысын есептейтін программа жазыңдар.

8-тапсырма. 7 пен 21 аралығындағы сандардың қосындысын есептейтін программа жазыңдар.

9-тапсырма. 10 мен 100 аралығындағы сандардың қосындысын есептейтін программа жазыңдар.

10-тапсырма. 1 мен 100 аралығындағы сандардың қосындысын есептейтін программа жазыңдар.

11-тапсырма. Мына программалық кодты жазып, нәтижесін есептеңдер.

```
var x = 5;
var s = 0;
while(x)
{s + = x; x --;}
alert(s);
```

12-тапсырма. Мына программалық кодты жазып, нәтижесін есептеңдер.

```
var x = 5;
var s = 0;
while (--x) s + = x; s + +;
alert(s);
```

13-тапсырма. Мына программалық кодты жазып, нәтижесін есептеңдер.

```
var x = 5;
var s = 0;
while(-- x || s <10) s + = x;
alert(s);
```

14-тапсырма. Мына программалық кодты жазып, нәтижесін есептеңдер.

```
var x = 5;
var s = 0;
while(-- x && s) s + = x;
alert(s);
```

15-тапсырма. Мына программалық кодты жазып, нәтижесін есептеңдер.

```
var x = 5;
var s = 0;
while(-- x && s <10) s + = x;
alert(s);
```

5-БӨЛІМ

АҚПАРАТТЫҚ ЖҮЙЕЛЕР

Күтілетін нәтижелер:

- Big Data-ны пайдаланудың оң және теріс әсерлерін бағалау;
- программалық жасақтаманы суреттерді өңдеу үшін пайдалану,
- деректер қоры жолдарының деректер типін анықтау;
- біркестелі және көпкестелі деректер қорын құру;
- деректерді енгізуге арналған форма жасау;
- алынған деректерді қолдана отырып, есептерді жасау;
- конструкторды пайдаланып, іріктеу сұрауын құру;
- кестеден деректерді таңдау үшін құрылымдық сұраныстар тілін (SQL) қолдану;
- web-беттердің деректер базасымен байланысын орнату.

§ 53–54. Big Data

Естеріңе түсіріңдер:

- *сайт дегеніміз не?*
- *web-бет құрудың әдістері қандай?*
- *стильдің каскадтау кестесі не үшін қолданылады?*
- *мультимедианы web-бетке енгізуге бола ма?*
- *«Big Data» түсінігімен таныссыңдар ма?*

Меңгерілетін білім:

- *«Big Data» түсінігі;*
- *Big Data пайдаланудың оң және теріс әсерлерін бағалау.*

Big Data (тура аударма – «үлкен деректер») дегеніміз не? «Үлкен деректер» терминін деректердің үлкен көлемін басқаруға және талдауға қатысты деп болжауға болады. **Деректер** – компьютерді басқаратын және электр сигналдары түрінде сақталатын және берілетін магниттік, оптикалық, механикалық тасығыштарға жазылатын шамалар, белгілер немесе символдар.

«Big Data» термині қысқа уақыт ішінде алуан түрлі, ауқымды, құнды деректерді жинау дегенді білдіреді. Мұндай деректерді өңдеу машинасыз жүзеге аспайды.

Соңғы уақытқа дейін деректер электрондық кестелермен немесе деректер қорымен шектелді әрі барлығы өте ретті және ұқыпты болды. Жолдар мен бағандарға орналастыруға келмейтіндердің барлығы жұмыс істеу үшін тым қиын деп есептеліп, еленбеді. Алайда ақпарат сақтау саласындағы прогресс әртүрлі деректің көп санын тіркеп, сақтап, өңдеуге қол жеткізді. Нәтижесінде қазіргі таңда «деректер»ұғымына деректер қорынан бастап, фотосуреттер, бейнелер, дыбыс жазбалары, жазбаша мәтіндер және датчиктердің деректеріне дейін кіреді. Үнемі өсіп келе жатқан ақпарат ағынын бірнеше жыл бұрын елестету де мүмкін болмаған жолмен пайдалана алатынымызды білдіреді. Бүгінде компаниялар клиенттердің нақты санаттарының қандай затты, қашан сатып алуды қалайтынын болжай алады. Big Data сондай-ақ компанияларға өз қызметін әлдеқайда тиімді орындауға көмектеседі.

Термин экономикада, банк қызметінде, өндірісте, маркетингте, телекоммуникацияда, web-талдауда, медицинада және т.б. деректер ағынының жылдамдығының артуы тұрақты болатын, сапалы үлкен көлемдегі деректермен жұмыс істейтін ірі салаларда қолданылады.

Мысалы, **Нью-Йорк қор биржасы** күн сайын өткен сессия бойынша сауда-саттық туралы *1 терабайт* деректер жасайды.

Әлеуметтік медиа: статистика Facebook деректер қорына күн сайын 500 терабайт жаңа деректер жүктелетінін көрсетеді, негізінен әлеуметтік желі серверлеріне фото мен бейнені жүктеуден, хабар алмасу, постылар астындағы түсініктемелер және т.б. салдарынан пайда болады.

Реактивті қозғалтқыш ұшу кезінде әрбір 30 минут сайын *10 терабайт* деректерді тудырады. Күн сайын мыңдаған ұшу болғандықтан, деректер көлемі петабайтқа жетеді.

Ақпаратты тез жинақтаумен бірге деректерді талдау технологиясы да жылдам қарқынмен дамуда. Егер бірнеше жыл бұрын клиенттерді ұқсас қалаулары бар топтарға саралау ғана мүмкін болса, енді нақты уақыт режимінде әрбір клиент үшін модельдер құруға болады, мысалы, нақты тауарды іздеу Интернет желісі арқылы жүзеге асырылады және салынған үлгіге сәйкес лайықты жарнама немесе нақты ұсыныстар шығарылады. Модель сондай-ақ бірнеше жыл бұрын ойластырылған нақты уақыт режимінде реттелуі және қайта құрылуы мүмкін.

Үлкен деректер көлемі, жасалу жылдамдығы, түрленуі және өзгергіштігі бойынша ерекшеленеді. Бұл сипаттамаларды толығырақ қарастырайық.

1. **Көлем**. *Big Data* термині үлкен өлшеммен байланысты. Деректер мөлшері – қажетті нәрселерді анықтаудағы маңызды көрсеткіш. Күн сайын 6 миллион адам сандық медианы пайдаланады, бұл – алдын ала бағалау бойынша 2.5 квинтиллион байт деректер. Сондықтан назар аударатын бірінші сипаттама – көлемі.
2. **Алуан түрлілік** – келесі аспект. Ол құрылымдалған және құрылымдалмаған гетерогенді көздер мен деректер табиғатына сілтеме жасайды. Бұрын электрондық кестелер мен деректер қорлары қосымшалардың көпшілігінде қарастырылатын ақпараттың жалғыз көзі болды. Қазіргі таңда электрондық хаттар, фото, бейне, PDF файлдар, аудио формасындағы деректер де аналитикалық қосымшаларда қаралады. Құрылымдалмаған деректердің осындай алуан түрлілігі сақтау, өндіру және талдау проблемаларына алып келеді: компаниялардың 27%-і лайықты деректермен жұмыс істейтініне сенімді емес.
3. **Жасалу жылдамдығы.** Деректердің талаптарды қанағаттандыру үшін қаншалықты тез жинақталатыны және өңделетіндігі әлеуетті анықтайды. Жылдамдық ақпарат

көздерінен – бизнес үдерістерден, қосымшалардың логоларынан, әлеуметтік желілер мен медиа сайттарынан, сенсорлардан, мобильді құрылғылардан ақпарат ағынының жылдамдығын анықтайды. Деректер ағыны уақыт өте үлкен және үздіксіз болып келеді.

4. **Өзгергіштік** – өңдеу мен басқаруды қиындататын уақыттың кейбір сәттерінде деректердің тұрақсыздығын сипаттайды. Мысалы, деректердің басым бөлігі өз ерекшелігіне сай құрылымдалмаған.

Big Data ұсынатын артықшылықтар:

1. Әртүрлі көздерден деректерді жинау.
2. Нақты уақытта талдау арқылы – бизнес процестерін жақсарту.
3. Үлкен көлемді деректерді сақтау.
4. Инсайттар. Big Data құрылымдалған және жартылай құрылымдалған деректердің көмегімен жасырын ақпаратқа аса бай.
5. Үлкен деректер тәуекелді азайтуға және қолайлы тәуекел-талдаушының арқасында дұрыс шешімдер қабылдауға көмектеседі.

Big Data-ның қиыншылықтары:

1. Деректердің құпиялылығы – Big Data біздің жеке өміріміз туралы көптеген ақпаратты қамтиды, оның құпиялылығын сақтауға толық құқымыз бар.
2. Деректерді қорғау – егер белгілі бір мақсат үшін біздің деректеріміз бөгде жандардың қолында болса, осы деректеріміздің сақталуы мен қауіпсіздігіне сенімді бола аламыз ба?

1 Сұрақтарға жауап берейік

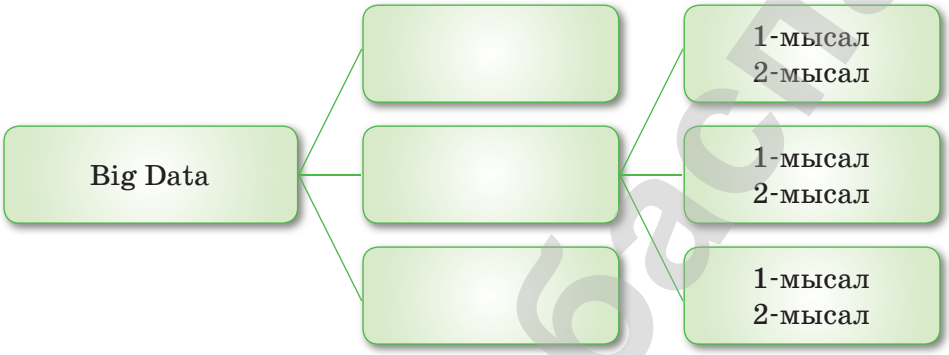
1. Big Data деген не?
2. Big Data-ның негізгі сипатаммалары қандай?
3. Big Data қайда қолданылады?

2 Ойланайық, талқылайық

1. Біздің өмірде Big Data мәні қаншалықты маңызды?
2. Неліктен Big Data-ны қолдану тез дамып келеді?
3. Не себепті өзіміз туралы деректердің сақталуы мен қауіпсіздігіне толық сенімді емеспіз?

3 Талдап, салыстырайық

1. Big Data-ны қолданудың артықшылықтарын атаңдар.
2. Big Data-ны қолданудың кемшіліктерін атаңдар.
3. Қосымша ақпарат көздерін пайдалана отырып, өмірде Big Data-ны қолданудың нақты мысалдарын келтіріңдер және деректер бойынша маман болу үшін қандай білім мен біліктер қажеттігін талдаңдар.



4 Дәптерде орындайық

Төменде көрсетілген кестеге үлкен деректер мен дәстүрлі деректер сипаттамаларын толтырыңдар.

	Дәстүрлі деректер қоры	Үлкен деректер
Қолдану саласы		
Деректер сипаттамасы		
Деректер сақтау тәсілдері		
Деректерді сақтау және өңдеу моделі		
Өңдеуге арналған ақпарат саны		

5 Компьютерде орындайық

Компьютеріңде орнатылған браузердің бірін қолданып:

1. ДҚБЖ-ны табыңдар.
   - SQL;
   - dBase;
   - Access;
   - Fox Pro;
   - Paradox.
2. Деректер қорын құрыңдар, онда мыналар көрсетіледі: ДҚБЖ атауы, ДҚБЖ пайдалану сипаты, функциялар, модель түрі.
3. Ең үлкен функция ДҚБЖ-ның қай түрінде бар?
4. Жұмысты сақтаңдар.

6 Ой бөлісейік

Интернеттен «Цифрлы Қазақстан және Smart City» бағдарламалары жайлы деректерді тауып, олардың болашағы туралы ой бөлісіңдер.

§ 55–56. Деректер қорының негізгі ұғымдары

Естеріңе түсіріңдер:

- *Big Data түсінігі;*
- *Big Data-ны пайдалану.*

Меңгерілетін білім:

- *реляциялық деректер қоры түсінігі;*
- *өріс, жазба, индекс түсініктерінің анықтамалары;*
- *деректер қоры кестелерінің арасындағы реляциялық байланыс.*

Кез келген кәсіби қызмет түрі ақпаратты жинау, сақтау және оны іріктеуді ұйымдастырумен айналысады. Бүгінгі күні белгілі бір ұйымдастыру тәсілін немесе механизмді қажет ететін деректер қоры күнделікті өмірдің ажырамас бөлігі болды. Мұндай механизмді деректер қорын басқару жүйесі (ДҚБЖ) деп атайды. Осы негізгі ұғымдарды қарастырайық.

Деректер қоры (ДҚ) – мекеменің ақпараттық қажеттіліктерін қанағаттандыруға арналған логикалық байланысқан деректердің жиынтығы (және олардың сипаттамасы).

Деректер қорын басқару жүйесі (ДҚБЖ) – қолданушыларға деректер қорын анықтауға, жасауға, қолдауға және оны бақылауға мүмкіндік беретін программалық жасақтама.

Деректер қорын басқару жүйесі бұрыннан бері қолданыста, олардың көбісінің пайда болуы үлкен есептеуіш машиналарындағы құрылымдалмаған файлдар жүйелерінің пайда болуына негізделген.

Деректер қорын басқару жүйелері саласында жалпыға бірдей қазіргі заманғы технологиялармен қатар, дамып келе жатқан бизнестің талаптары, корпоративті деректердің үнемі ұлғаюы және Интернет технологиялардың әсерінен жаңа бағыттары пайда болды.

Реляциялық (ағылш. *relation* – байланыс**) деректер қоры**

Негізгі ақпараттық ағындарды басқару бастауын дәстүрлі деректер қоры жүйесінен алатын реляциялық басқару жүйесі көмегімен жүзеге асырылады. Реляциялық деректер қоры мен клиент-сервер технологиясын біріктіру заманауи кәсіпорынға тауарлар мен қызметтер нарығында бәсекеге қабілеттілігін сақтай отырып, өз деректерін табысты басқаруға мүмкіндік береді.

Реляциялық деректер қорында математикалық қатынастар теориясына негізделген қуатты теориялық негіз бар. *Реляциялық деректер қоры* теориясының пайда болуы, екі класқа бөлінетін сұраныстар тілін жасауға негіз болды:

– қатынастарға қолданылатын арнайы операторлар арқылы сұраныстарды сипаттауға мүмкіндік беретін **алгебралық тілдер;**
– қолданыстағы қарым-қатынастардың белгілі бір жиынтығынан жаңа қатынасты анықтайтын өрнекті жазу ережелерінің жинағы болып табылатын **предикаттық есептеу тілдері.**

Ендеше предикаттық есептеу тілі деректер қорындағы қолданыстағы қатынастардан *сұранысқа* жауап ретінде алынатын жаңа қатынасты анықтау әдісі болып табылады.

Реляциялық ДҚБЖ мысалдары: *MySql, PostgreSql.*

Реляциялық модельде шынайы өмір нысандары және нысандар арасындағы байланыс, өзара байланысқан *кестелер* (қатынастар) көмегімен ұсынылады.

ДҚБЖ функциялары бір немесе бірнеше кестеден ақпарат таңдау үшін пайдаланған жағдайда да (яғни сұраныс жасалғанда) нәтиже кесте түрінде ұсынылады. Сонымен қатар *сұраныстың* басқа *сұраныстың* нәтижелерін пайдалану арқылы жасауға болады.

Деректер қорының әр **кестесі** *жолдар* мен *бағандар* жиынтығынан тұрады, мұнда **жолдар** (жазба) – нысан, нақты оқиға немесе құбылыс саны, ал **бағандар** (өрістер) – нысанның, оқиға немесе құбылыстың атрибуттары (белгілер, сипаттамалар, параметрлер).

Деректер қорымен жұмыс жасау барысында пайда болатын ең негізгі мәселе – іздеуді ұйымдастыру. Сонымен қатар деректер қорында әдетте ақпарат көп болғандықтан, программалаушыларға тек іздеуді ғана емес, тиімді іздеуді жүзеге асыру міндеті қойылады, яғни іздеуді салыстырмалы түрде аз уақытта және жеткілікті дәлдікпен ұйымдастыру. Ол үшін (сұраныстардың өнімділігін тиімділеу үшін) кестенің кейбір өрістеріне **индекстеу** жүргізіледі. Индекстер бір бағанның көрсетілген мәнімен жолдарды жылдам іздеу үшін ыңғайлы. Индекссіз кесте бірінші жазбадан бастап тиісті жолдар табылмайынша, бүкіл кесте бойынша оқылады. Кесте неғұрлым үлкен болса, шығындар да көп болады. Егер кестеде қарастырылатын бағандар бойынша индекс болатын болса, онда *деректер қоры* деректердің барлығын қарастырмай, деректер файлының ортасынан іздеу үшін позицияны жылдам анықтай алады.

Деректер қоры кестелерінің арасындағы реляциялық байланыс

Шынайы өмір нысандары арасындағы *байланыстар*, деректер құрылымында көрініс табуы мүмкін немесе формальды емес деңгейде болуы мүмкін.

Деректер қорының екі немесе одан да көп кестелерінің арасында *бағынышты қатынастар* болуы мүмкін. Олар негізгі (ата-ана) кестенің әрбір жазбасы үшін бағынышты (бала) кестенің бір немесе бірнеше жазбасы болуы мүмкін. Деректер қоры кестелерінің арасындағы байланыстың 3 түрі бар:

– «біреуден – көпке»;
– «біреуден – біреуге»;
– «көптен – көпке».

«Біреуден – көпке» қатынасы

«Біреуден – көпке» қатынасында негізгі кестенің бір жазбасына бағынышты кестенің бірнеше жазбасы сай келеді.

«Біреуден – көпке» байланысын кейде «көптен – біреуге» деп те атайды. Екі жағдайда да кестелер арасындағы байланыс өзгеріссіз қалады. Байланыстың бұл түрі реляциялық деректер қоры үшін кең таралған. Сонымен қатар ол деректердің иерархиялық құрылымын модельдеуге мүмкіндік береді.

«Біреуден – біреуге» қатынасы

«Біреуден – біреуге» қатынасында негізгі кестенің бір жазбасына бағынышты кестенің бір жазбасы сәйкес келеді. Қатынастың бұл түрі «біреуден – көпке» қатынасына қарағанда аз қолданылады. Егер деректер қорының кестесі қосымша ақпараттардан ұлғайып кетпеуін қаласақ, осы қатынас түрін қолданамыз. Алайда бірнеше кестедегі өзара байланысқан ақпараттарды оқу үшін, бір кестеде сақталған деректерден бір ақпаратты оқудың орнына бірнеше операция орындау керек болады.

«Көптен – көпке» қатынасы

«Көптен – көпке» қатынасы мынадай жағдайларда қолданылады:

– негізгі кестедегі бір жазбаға бағынышты кестенің бірден көп жазбалары сәйкес келеді;

– бағынышты кестенің бір жазбасына негізгі кестенің бірнеше жазбасы сәйкес келеді.

Реляциялық деректер қорындағы кез келген «көптен – көпке» қатынасын, қосымша кестелерді енгізу арқылы «біреуден – көпке» қатынасына ауыстыру керек.

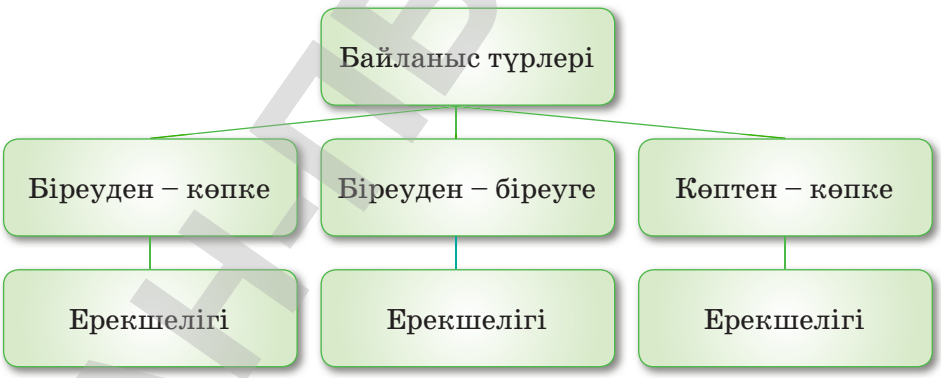
1 Сұрақтарға жауап берейік

1. Деректер қоры деген не?
2. Деректер қорын басқару қажет пе?
3. Реляциялық байланыс деген не?
4. Деректер қоры кестелері байланыстарының қандай түрлері бар?

2 Ойланайық, талқылайық

1. Деректер қорын басқару жүйесі не үшін қажет?
2. Неліктен реляциялық деректер қоры кестелермен тығыз байланысты?

3 Талдап, салыстырайық



4 Дәптерде орындайық

Реляциялық деректер қорындағы нысандарға анықтама беріп, дәптерге жазыңдар.

5 Компьютерде орындайық

MySQL серверін орнату:

1. *http://dev.mysql.com* сайтынан MySQL Database Server жүктеп алыңдар.

2. Жүктелген файлды тінтуірдің сол жақ батырмасымен 2 рет шертіп, орнатуды бастаңдар.
3. «typical install» (қарапайым орнату) нұсқасын қолданып, серверді орнатуды бастаңдар. Серверді орнату жылдам жүру керек.
4. Серверді орнату аяқталғаннан кейін, аяқтау батырмасын баспастан бұрын Configure the MySQL Server now жалаушасының қосылып тұрғанына көз жеткізу керек, бұл Configuration Wizard (конфигурация шебері) жүргізу үшін қажет.
5. Configuration Wizard-ты жүргізу барысында жалаушалар қосу үшін Standard Configuration (Стандартты конфигурация) ⇒ Install as Windows Service (Windows қызметі ретінде орнату) ⇒ Include Bin Directory in Windows Path (Windows іздеу жолына Bin каталогін қосу) командасын орындаймыз.
6. Конфигурациялау барысында сендерге root артықшылығы жоғары қолданушы үшін құпиясөз орнату ұсынылады. Құпия сөзді орнатып, оны жаттап алғандарың дұрыс, себебі оны кейін жұмыс барысында қолданасыңдар.
7. Консольді ашыңдар (Start ⇒ Run ⇒ Command (Бастау ⇒ Орындау ⇒ Command)), mysql және root p командалары арқылы артықшылығы жоғары қолданушы ретінде тіркеліңдер. Осы кезде сендерге парольді енгізу ұсынылады, әрі қарай mysql> көмекші команда пайда болады.
8. grant all privileges on *.* to 'lrng sql'@'localhost' identified by 'xxxxx'; («xxxxx» қолданушы үшін таңдап алған парольмен ауыстырыңдар) командасы арқылы жаңа артықшылығы жоғары қолданушы құрыңдар.
9. quit; (шығу) командасы арқылы сеансты аяқтаңдар және mysql командасы көмегімен консольде жаңа қолданушы ретінде тіркеліңдер.

6 Ой бөлісейік

Күнделікті өмірде қандай сұраныс тілдерімен кездестіңдер?

§ 57–58. Деректер қорындағы бастапқы кілт

Естеріңе түсіріңдер:

- *реляциялық деректер қоры;*
- *өріс, жазба, индекс түсініктері;*
- *деректер қоры кестелерінің арасындағы байланыс.*

Меңгерілетін білім:

- *деректер қорындағы бастапқы кілтті анықтау.*

Алдымен мына сұрақ жөнінде ойланып көрейік: әңгімелесушіге, сөйлесіп отырған адамының нақты сол адам және басқа мұндай адам жоқ екеніне сенімді болатындай қандай ақпарат бере аламыз? Тегін айту жеткіліксіз, себебі тегі бірдей адамдар кездеседі. Егер қажетті нысан адам болса, онда біз жобамен оның қандай да бір жағдайда қалай әрекет еткендігін еске түсіру арқылы кім екенін түсіндіре аламыз. Ал компьютер мұндайды түсінбейді, оған кім екенін түсіндіру үшін нақты ережелер қажет. Деректер қорын басқару жүйесінде мұндай мәселені шешу үшін *бастапқы кілт* ұғымы енгізілді.

Деректер қорындағы әр кестенің бастапқы кілті болу керек – нысанның немесе жазбаның әр данасын бірегей түрде анықтайтын өріс немесе өрістер жиынтығы. ДҚ кестесінің бастапқы кілт мәні бірегей болу керек, яғни кестеде бастапқы кілт мәндері бірдей екі немесе одан да көп жазбалары болмау қажет. Ол өте аз мөлшерде болу керек, яғни өшірілсе, оның бірегейлігіне әсер ететін өрістер болмауы тиісті.

Бастапқы кілт (`primary key, PK`) – кестедегі жазбаны бірегей түрде анықтайтын өрістердің ең аз жиынтығы. Демек, **бастапқы кілт** – ең алдымен, кесте өрістерінің жиынтығы, екіншіден, осы өрістердің әр мәндерінің жиынтығы кестедегі бір жазбаны (жолды) анықтайды және үшіншіден, өрістердің бұл жиынтығы бірдей сипатқа ие ең азы болуы керек. Бастапқы кілт тек бір ғана бірегей жазбаны анықтағандықтан, кестеде ешбір екі жазбаның бірдей бастапқы кілт мәндері бола алмайды.

Деректер қорының мысалы (*14-кесте*):

***14-кесте.** Телефон кітапшасы*

Аты-жөні	Телефон нөмірі	Мекенжайы
Асанов Асан Асанұлы	233-44-55	Шәріпов көшесі, 12
Үсенов Үсен Үсенұлы	233-66-77	Абылайхан даңғ., 32

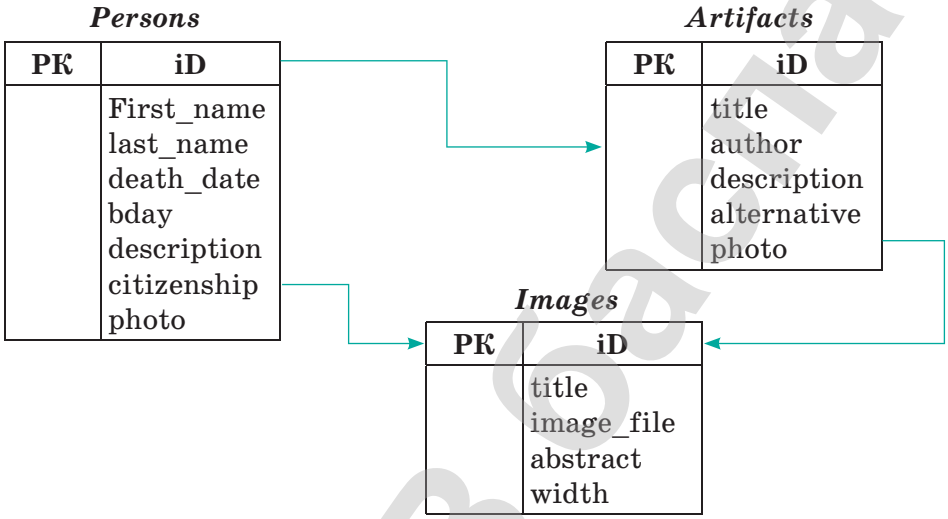
14-кестеде аты-жөні мен мекенжайы адам туралы жазбаны бірегей түрде бөліп алуға мүмкіндік береді. Егер шешілетін мәселеге байланыссыз айтатын болсақ, мұндай ақпараттар ізделіп отырған адамды дәл табуға мүмкіндік бермейді, себебі дәл осындай аты-жөнімен басқа қалаларда да тұратын адамдар бар. Егер біздің мақсатымыз үшін қаланың атауынсыз, аты-жөні, телефон нөмірі және мекенжайы туралы ақпарат жеткілікті болса, онда аты-жөні және мекенжай өрістері бастапқы кілт бола алады. Кез келген жағдайда бастапқы кілтті жасау мәселесі деректер қорын жобалаушы маманның құзырында болады. Бұл мәселенің шешімі ретінде кестедегі жазбаны анықтайтын сипаттамаларды ерекшелеу (логикалық немесе табиғи бастапқы кілт үшін тапсырмалар) немесе кестедегі жазбаларды бірегей анықтауға мүмкіндік беретін қосымша өрісті (суррогатты немесе жасанды бастапқы кілт үшін тапсырмалар) жасау болып табылады. *Логикалық кілт* мысалы ретінде тұрғындар туралы деректер қорындағы төлқұжат нөмірі немесе телефон кітапшасындағы аты-жөні мен мекенжайды қарастыруға болады. *Суррогатты кілтті* жасау үшін кестеге, кестенің әр жолына бірегей және мәні бүтін сан болатын id өрісін қосуға болады. Мұндай суррогатты кілттер табиғи бастапқы кілт өрістердің үлкен жиынтығынан тұрса немесе оны ерекшелеу күрделі болған жағдайда пайдаланылады.

Бастапқы кілттер жазбаны бірегей анықтаудан бөлек, басқа кестелермен байланысты ұйымдастыру үшін де қолданылады.

Мысалы, бізде 3 кесте бар делік: тарихи тұлғалар туралы ақпараттан тұратын (Persons), олардың өнертабыстары туралы ақпарат жазылған(Artifacts), тұлғалардан және артефакторлардан тұратын кесте (Images) (*26-сурет*).

Бұл кестелердің бәрінде де *бастапқы кілт* ретінде id (идентификатор) өрісі болып табылады. *Artifacts* кестесінде Persons кестесіндегі өнертабыс авторына тағайындалған идентификатор жазылған author өрісі бар. Бұл өрістің әрбір мәні Persons кестесінің бастапқы кілті үшін **сыртқы кілт** болып табылады. Сонымен қатар Persons және *Artifacts* кестелерінде, Images кестесіндегі суретке сілтеме болатын photo өрісі бар. Бұл өрістер Images кестесінің бастапқы кілті үшін сыртқы кілті болып табылады және *Persons-Images* бен *Artifacts-Images* бірегей логикалық байланысты орнатады. Егер тұлғалар кестесіндегі photo *сыртқы кілтінің* мәні 10-ға тең болса, онда

бұл тұлға фотосуретінің `id` идентификаторы суреттер кестесінде 10-ға тең. Осылайша, **сыртқы кілттер** деректер қорының кестелері (негізгі және бағынышты) арасындағы байланысты ұйымдастыру және деректердің тұтастығы туралы шектеулерді сақтау үшін пайдаланылады.



26-сурет. *Басқа кестелермен байланыс ұйымдастыру үшін бастапқы кілттерді пайдалану мысалы*

1 Сұрақтарға жауап берейік

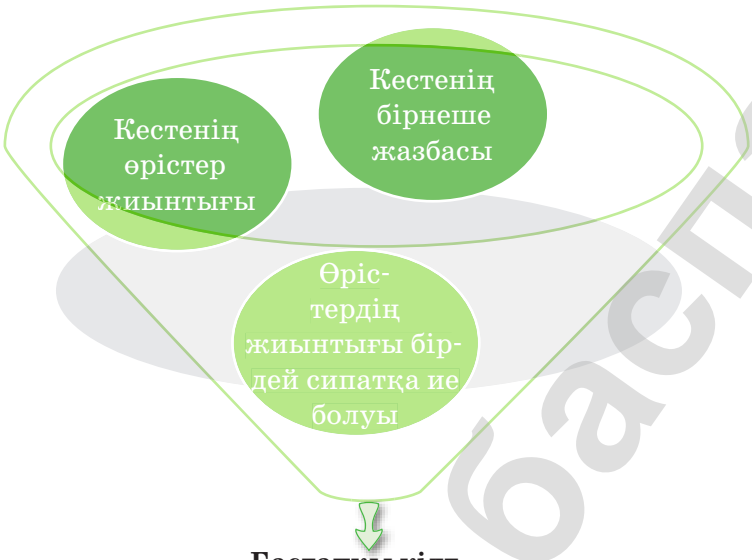
1. Бастапқы кілт деген не?
2. Күнделікті өмірде бастапқы кілтке не балама бола алады?
3. Логикалық кілт деген не?
4. Суррогатты кілт қалай жасалады?
5. Сыртқы кілт деген не?

2 Ойланайық, талқылайық

1. Бастапқы кілттер не үшін қажет?
2. Бастапқы кілттерге сыртқы кілттер не үшін пайдаланылады?
3. Не себепті кестенің екі жазбасында бірдей бастапқы кілттер болмайды?
4. Сыртқы кілт не үшін қажет?

3 Талдап, салыстырайық

Бастапқы кілттің дұрыс сипаттамаларын анықтаңдар.



4 Дәптерде орындайық

Дәптерде «Телефон кітапшасы» деректер қорын жасап, оны сыныптастарыңның аты-жөні мен телефон нөмірлерімен толтырыңдар.

«Телефон кітапшасы» деректер қоры		
Аты-жөні	Телефон нөмірі	Мекенжайы

5 Компьютерде орындайық

Адамдар және телефон нөмірлері деген екі кесте құрыңдар. Кестелерде бастапқы және сыртқы кілттерді орнатыңдар.

Кесте. *Адамдар*

?	Аты-жөні
1	Асанов Асан Асанұлы
2	Үсенов Үсен Үсенұлы
3	Дүйсенов Дүйсен Дүйсенұлы

Кесте. *Телефон нөмірлері*

?	Телефон	?
1	87011234567	1
2	87059876543	1
3	87084561223	2
4	87001231234	3
5	87784564568	3

Төмендегі сұрақтарға жауап беріңдер:

1. Телефон нөмірлері кестесінде қай өріс бірегей болып табылады?
2. Осы кестенің сыртқы кілті «Адамдар» кестесінің бастапқы кілті болып табыла ма?
3. «Телефон нөмірлері» мен «Адамдар» арасындағы байланыс ненің көмегімен жүзеге асырылады?
4. Әр адамда неше телефон нөмірі бар екенін анықтаңдар.

6 Ой бөлісейік

Мектеп кітапханасындағы оқулықтар деректер қорының бастапқы кілті қандай болуы мүмкін?

§ 59–60. Деректер қорын әзірлеу

Естеріңе түсіріңдер:

- *деректер қорындағы бастапқы кілт деген не?*

Меңгерілетін білім:

- *деректер қорындағы деректер түрлері (SQL);*
- *біркестелі деректер қорын құру (SQL);*
- *көпкестелі деректер қорын құру (SQL).*

MySQL – дүниежүзінде ең көп қолданылатын, тегін және ашық, реляцияланған деректер қоры жүйесі (RDBMS). Серверлік программа ретінде, бірнеше қолданушыға бірнеше дерек қорын қолдануды қамтамасыз етеді. MySQL сөзіндегі «My» сөзі, программа жасаушысы Майкл Видньюстың (Michael Widenius) қызының аты – «My» сөзінен алынған. Ал SQL фразасы – Құрылымдасқан Тапсырыс Тілі (Structured Query Language) дегенді білдіреді.

Жалпы барлық танымал деректер қорлары, оның ішінде MySQL де жолдар, мерзімдер және сандар сияқты деректер типтерін сақтау қабілетіне ие.

Әдетте, олардың арасындағы айырмашылық деректердің арнайы типтерін, мысалы, XML құжаттарды, үлкен көлемді мәтіндерді немесе екілік құжаттарды сақтау мүмкіндіктерінде болып тұр.

Деректер – деректер қорында бірнеше типті деректердің бір типінде сақтайтын ақпараттар жиыны. Олардың көмегімен кестенің нақты бағанының құрамындағы деректер үшін негізгі ережелерді, сонымен қатар оларға берілетін жады көлемін орнатады.

Біз деректердің тек таңбалық, сандық және уақытша типтерін қарастырамыз.

Таңбалық деректер

Таңбалық деректер бекітілген немесе айнымалы ұзындықтың жолдары ретінде сақталуы мүмкін. Айырмашылық: бекітілген ұзындық жолдары оң жағынан бос орындармен толтырылады, ал айнымалы ұзындық толтырылмайды. Таңбалық жолды анықтау барысында онда сақталатын жолдың максималды өлшемін беру қажет. Мысалы, егер 20 таңбадан тұратын жолды сақтау керек болса, онда төмендегі сипаттамалардың бірін қолдануға болады:

```
CHAR(20) /* бекітілген ұзындық жолы */
VARCHAR(20) /* айнымалы ұзындық жолы */
```

Деректердің бұл типінің максималды ұзындығы 255 таңбаны құрайды. Әлдеқайда ұзын жолдарды сақтау үшін `tinytext`

(кішігірім мәтін), text (мәтін), mediumtext (орташа мәтін), longtext (ұзын мәтін) сияқты мәтіндік деректерді пайдаланған жөн. Жалпы char типін бағанда тек ұзындықтары бірдей жолдарды сақтау ұйғарылған жағдайда (мысалы, мемлекеттердің қысқартылған атаулары) қолдану ыңғайлы. Ал varchar типі ұзындықтары әртүрлі жолдар үшін қолданылады. Char және varchar типтері деректер қорының барлық негізгі серверлерінде бірдей пайдаланылады.

Латын әліпбиін қолданатын тілдердегі таңбалар жиынтығында аз таңба болады, яғни әр таңба 1 байт ретінде сақталады. Ал, корей немесе жапон тілдерінде, керісінше, таңбалар өте көп. Осылайша, ондағы бір таңбаны сақтау үшін бірнеше байт қажет болады. Сондықтан мұндай таңбалар жиынтығын **көпбайтты таңбалар жиынтығы** (multibyte character sets) деп атайды. MySQL түрлі таңбалар жиынтығын (бір және көпбайтты таңбалар жиынтығын) пайдаланып, деректерді сақтай алады. Сервердің жұмыс жасайтын таңбалар жиынтығын көру үшін show (көрсету) командасын пайдалану керек:

```
mysql> SHOW CHARACTER SET;
```

Charset	Description	Default collation	Maxlen
big5	Big5 Traditional Chinese	big5_chinese_ci	2
dec8	DEC West European	dec8_swedish_ci	1
cp850	DOS West European	cp850_general_ci	1
hp8	HP West European	hp8_english_ci	1
koi18r	KOI8-R Relcom Russian	koi18r_general_ci	1
latin1	ISO 8859-1 West European	latin1_swedish_ci	1
latin2	ISO 8859-2 Sentral European	latin2_general_ci	1
swe7	7bit Swedish	swe7_swedish_ci	1
ascii	US ASCII	ascii_chinese_ci	1

Мәтіндік деректер

Егер char немесе varchar типті бағанның 255 таңбасы жетпейтін деректерді сақтау қажет болса, онда мәтіндік типтің бірін пайдалану керек болады.

15-кестеде қолжетімді мәтіндік типтері көрсетілген.

***15-кесте.** MySQL-дің мәтіндік типті деректері*

Тип	Таңбалардың максималды саны
Tinytext	255
Text	65 535
Mediumtext	16 777 215
Longtext	4 294 967 295

Мәтіндік типтің түрін таңдау барысында мынадай жағдайларды есте сақтау керек:

- Егер мәтіндік бағанға жүктелетін деректердің өлшемі осы типтің максималды өлшемінен асып кетсе, онда сыймай қалған деректер қиылып тасталады.
- `varchar` типті бағанға қарағанда, мұндай бағанға деректерді жүктеу барысында жол соңындағы бос орындар өшірілмейді.
- `text` типті бағандарды сұрыптау және топтастыруда қолдану барысында алғашқы 1024 байт ғана қолданылады, алайда қажет болған жағдайда бұл мәнді ұлғайтуға болады.
- Әртүрлі мәтіндік типтер тек MySQL-ге тән. SQL Server-де үлкен өлшемді таңбалық деректер үшін бір ғана `text` типі бар, ал DB2 және Oracle үшін `clob (Character Large Object)` деректер типі пайдаланылады. Еркін форматтағы деректер үшін баған құру барысында, мысалы, 255 таңбамен шектегілерің келмеген notes бағанының тұтынушылар мен қызмет бөлімі арасындағы өзара байланысы туралы ақпаратты сақтау үшін `text` немесе `mediumtext` типін таңдаймыз.

Сандық деректер

Сандық деректерді сақтау үшін жалғыз `numeric` типі жеткілікті деп пайымдасақ та, төмендегідей сандарды пайдаланудың әртүрлі тәсілдерін көрсететін түрлі типтер бар: тапсырыс берушіге тапсырыс берудің индикаторы болып табылатын баған *логикалық* деп аталады, ол 0 (жалған) және 1 (ақиқат) болуы мүмкін.

Жүйе арқылы жасалатын транзакция кестесінің бастапқы кілті, әдетте, 1-ден басталады және 1-ден ең үлкен мәндерге дейін артады. Клиенттің электрондық сауда қоржынындағы позиция нөмірі – осы түрдегі бағанның мәні 1-ден (максимум) 200-ге дейін оң бүтін сандар. Баспа сызбаларына арналған бұрғылау қондырғысының орналасу деректері, жоғары дәлдіктегі ғылыми немесе технологиялық деректер сегіз таңбалы санға дейінгі дәлдікті талап етеді. MySQL осы және басқа да ақпарат типтерімен жұмыс істеу үшін бірнеше сандық түрлеріне ие. Көптеген сандық түрлер бүтін сандарды сақтау үшін қолданылады. Осы типтердің бірін беру барысында деректер таңбасыз деп көрсетуге болады, онда сервер бағандағы барлық деректер теріс емес екенін біліп отырады. *16-кестеде* бүтін сандарды сақтауға арналған деректердің 5 типі көрсетілген.

16-кесте. MySQL-дің бүтін типті деректері

Тип	Таңбалы мәндердің диапазоны	Таңбасыз мәндердің диапазоны
Tinytext	-128-ден 127-ге дейін	0-ден 255-ке дейін
Smallint	-32768-ден 32767-ге дейін	0-ден 65535-ке дейін
Mediumint	-8388608-ден -8388607-ге дейін	0-ден 16777215-ке дейін
Int	-2147483648-ден 2147483647-ге дейін	0-ден 4294967295-ке дейін
Bigint	-9223372036854775808-ден 9223372036854775807-ге дейін	0-ден 18446744073709551615-ке дейін

Бүтін типтің біреуінің бағанын құру барысында MySQL деректерді сақтау үшін сәйкесінше 1 байттан `tinyint` типі 8 байтқа `bigint` дейінгі жады бөліп береді. Сондықтан жадыны қажетсіз пайдаланбас үшін, ең үлкен сандарды сақтау үшін жеткілікті мөлшердегі типті таңдап көріңдер.

Өзгермелі нүктелер үшін (мысалы, 3,1415927) *17-кестеде* көрсетілген типтердің бірін таңдауға болады.

17-кесте. MySQL-дің өзгермелі нүктелер үшін деректердің типтері

Тип	Таңбалардың максималды саны
Float (p.s)	-3,402823466E+38-ден -1,175494351E-38-ге дейін және 1,175494351E-38-ден 3,402823466E+38-ге дейін
Double (p.s)	1,7976931348623157E+38-ден -2,225073858072014E-308-ге дейін және 2,2250738585072014E-38-ден 1,7976931348623157E+308-ге дейін

Өзгермелі нүктелі типке дәлділік (`precision`) (ондық нүктенің оң және сол жағынан рұқсат етілген жалпы разрядтар саны) және масштаб (`scale`) (ондық нүктенің оң жағынан рұқсат етілген разрядтар саны) беруге болады, бірақ бұл параметрлер міндетті болып табылмайды. *17-кестеде* олар *p* және *s* түрінде көрсетілген.

Егер сақталған разрядтар саны көрсетілген масштаб және/немесе дәлдіктен асып кетсе, өзгермелі нүктелік бағанның дәлдігі мен масштабын берген жағдайда, бағанда сақталатын деректердің барлығы дөңгелектенетін болады.

Уақыт (даталық) деректер

Жолдар мен сандардан өзге, біз мерзім мен уақыт туралы ақпаратпен де жиі жұмыс жасаймыз. Деректердің бұл типін уақытша (temporal) деп атайды. Деректер қорындағы уақыт (даталық) деректер мысалдары:

- болашақ оқиғаның мерзімі, мысалы, сатып алушыға жеткізу күні;
- сатып алушы тапсырысын жеткізудің нақты мерзімі;
- қолданушының белгілі бір жолды өзгертетін күні мен уақыты;
- қызметкердің туған күні;
- деректер қоймасында yearly_sales (жылына сату) кестесінің қатарына сәйкес келетін жыл;
- конвейерлік таспада автокөліктің электр сымдарын монтаждауға қажетті уақыт. MySQL-де осыған ұқсас жағдайлардың барлығын өңдеуге қажетті деректер типтері бар.

18-кестеде MySQL-де қолданылатын уақыт (даталық) деректер типтері көрсетілген.

***18-кесте.** MySQL-дің уақыт (даталық) деректер типтері*

Тип	Бекітілген формат	Мүмкін мәндері
Date	YYYY-MM-DD	1000-01-01-ден 9999-12-31-ге дейін
Datetime	YYYY-MM-DD HH:MI:SS	1000-01-01 00:00-ден 9999-12-31 23:59:59-ға дейін
Timestamp	YYYY-MM-DD HH:MI:SS	1970-01-01 00:00-ден 2037-12-31 23:59:59-ға дейін
Year	YYYY	1901-ден 2155-ке дейін
Time	HHH:MI:SSS	-838:59:59-дан 838:59:59-ға дейін

Кестелер құру

MySQL деректер қорында қандай деректер типтерін сақтауға болатыны туралы нақты түсінік пайда болғаннан кейін, кестелерді сипаттағанда осы типтерді қалай қолданатынымызды көрейік. Тұлға туралы ақпаратты сақтауға арналған кестенің сипаттамасынан бастайық.

1-қадам: Жобалау

Ең алдымен, тұлға туралы деректерді бермес бұрын қандай сипаттамалардан тұру керек екенін анықтап алайық:

- Аты, тегі (name, surname);
- жынысы (gender);

- туған күні (birth date);
- мекенжайы (address);
- сүйікті асы (favorite foods).

Әрине бұл толық тізім емес, бірақ әзірге жеткілікті. Келесі қадам – бағандарға атау беру және деректер типін тағайындау. *19-кестеде* алғашқы нұсқасы келтірілген.

19-кесте. Person *(тұлға) кестесі*

Баған	Тип	Рұқсат етілген мәндер
Name	Varchar (40)	
Gender	Char (1)	
Birth_date	Date	
address	Varchar (100)	M, F
favorite foods	Varchar (200)	

Varchar типті name, address және favorite_foods бағандары ақпаратты еркін формада жазуға мүмкіндік береді. **Gender** (пол) бағанында тек бір символ рұқсат етіледі: M (ер) немесе F (әйел). birth_date (туған күні) бағанына date типі тағайындалған, себебі нақты уақыт қажет емес.

2-қадам: Нақтылау

Кесте бағандарын қайта талдау барысында мынадай пікірлер пайда болады:

- name бағаны шын мәнінде, аты және тегінен тұратын құрамды нысан.
- Бірнеше адамның есімдері, жынысы, туған күндері және т.б. ақпараттары бірдей болуы мүмкін болғандықтан, person кестесінде бірегей баған жоқ.
- Address бағаны да құрамды нысан, себебі оның құрамына көше, қала, облыс, мемлекет және пошта индексі туралы ақпараттар кіреді.
- favorite_foods бағаны – 0,1 немесе одан да көп тәуелсіз элементтерден тұратын тізім. Бұл деректерді бөлек кестеге енгізген дұрыс болар еді, ол кестеде нақты ас түрі қай адамға тиісті екенін белгілеу үшін person кестесіне сыртқы кілт жасау керек.

20-кестеден жоғарыда келтірілген барлық ескертулерді ескергеннен кейінгі person кестесінің түрін көре аласыңдар. Енді person кестесінің бірегейлігіне кепілдік беретін бастапқы кілті (person_id) бар болғандықтан, келесі қадам person кестесіне сыртқы кілті бар favorite_food кестесін құру керек (*21-кесте*).

person_id және food (ac) бағандары favorite_food кестесінің бастапқы кілтін құрайды. Сонымен қатар person_id бағаны person кестесіне сыртқы кілт болып табылады.

***20-кесте.** Өзгерістер енгізілген* Person *кестесі*

Баған	Тип	Рұқсат етілген мәндер
person_id	Smallint	
First_name	Varchar (20)	
Lastt_name	Varchar (20)	
Gender	Char (1)	
Birth_date	Date	
Street	Varchar (30)	
City	Varchar (20)	*M, F*
State	Varchar (20)	
Country	Varchar (20)	
Postal_code	Varchar (20)	

21-кесте. Favorite_food *(сүйікті асы) кестесі*

Баған	Тип
person_id	Smallint (unsignet)
Food	Varchar (20)

3-қадам: Деректер сызбасын басқарудың SQL өрнектерін құру

Жеке ақпаратты енгізу жөніндегі 2-кестені жобалау аяқталғаннан кейін, келесі қадам деректер қорында кесте құру үшін **SQL өрнектерді** қалыптастыру болып табылады. Person кестесін құруға арналған өрнек:

```
CREATE TABLE person (person_id SMALLINT UNSIGNED,
fname VARCHAR(20),
lname VARCHAR(20),
gender CHAR(1),
birth_date DATE,
address VARCHAR(30),
city VARCHAR(20),
state VARCHAR(20),
country VARCHAR(20),
postal_code VARCHAR(20),
CONSTRAINT pk_person PRIMARY KEY (person_id) ).
```

Бұл өрнекте соңғы элементтен басқасының барлығы түсінікті болу керек. Кестені сипаттау барысында деректер қорының серверіне, кестенің бастапқы кілті рөлінде қай бағандар

болатынын хабарлайды. Ол кесте үшін шектеулер (constraint) жасау арқылы жүзеге асырылады. Кестенің сипаттамасына шектеулердің бірнеше типінен біреуін қоямыз. Бұл шектеу бастапқы кілттің шектеуі (primary key constraint) болып табылады. Ол person_id кестесіне қойылады және pk атауы pk_person болады. Әдетте, бастапқы кілт үшін шектеулер атауын pk префиксінен бастаған жөн, кейін шектеулер тізімін қарағанда, қайсысы қай кестенің шектеуі екенін білу үшін кесте атауы беріледі.

1 **Сұрақтарға жауап берейік**

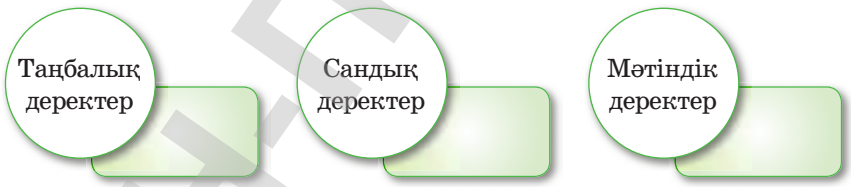
1. Деректердің қандай типтері бар?
2. Деректер типі арқылы кесте құрудың қадамдары қандай?

2 **Ойланайық, талқылайық**

1. Деректердің түрлі типтері не үшін қажет?
2. Деректер қорында таңбалық, мәтіндік және сандық ақпаратты қалай жалпылауға болады?

3 **Талдап, салыстырайық**

1. Деректердің ақпараттан айырмашылығы неде?
2. Деректер типтерінің белгілерін салыстырып, сызбаны толтырыңдар.

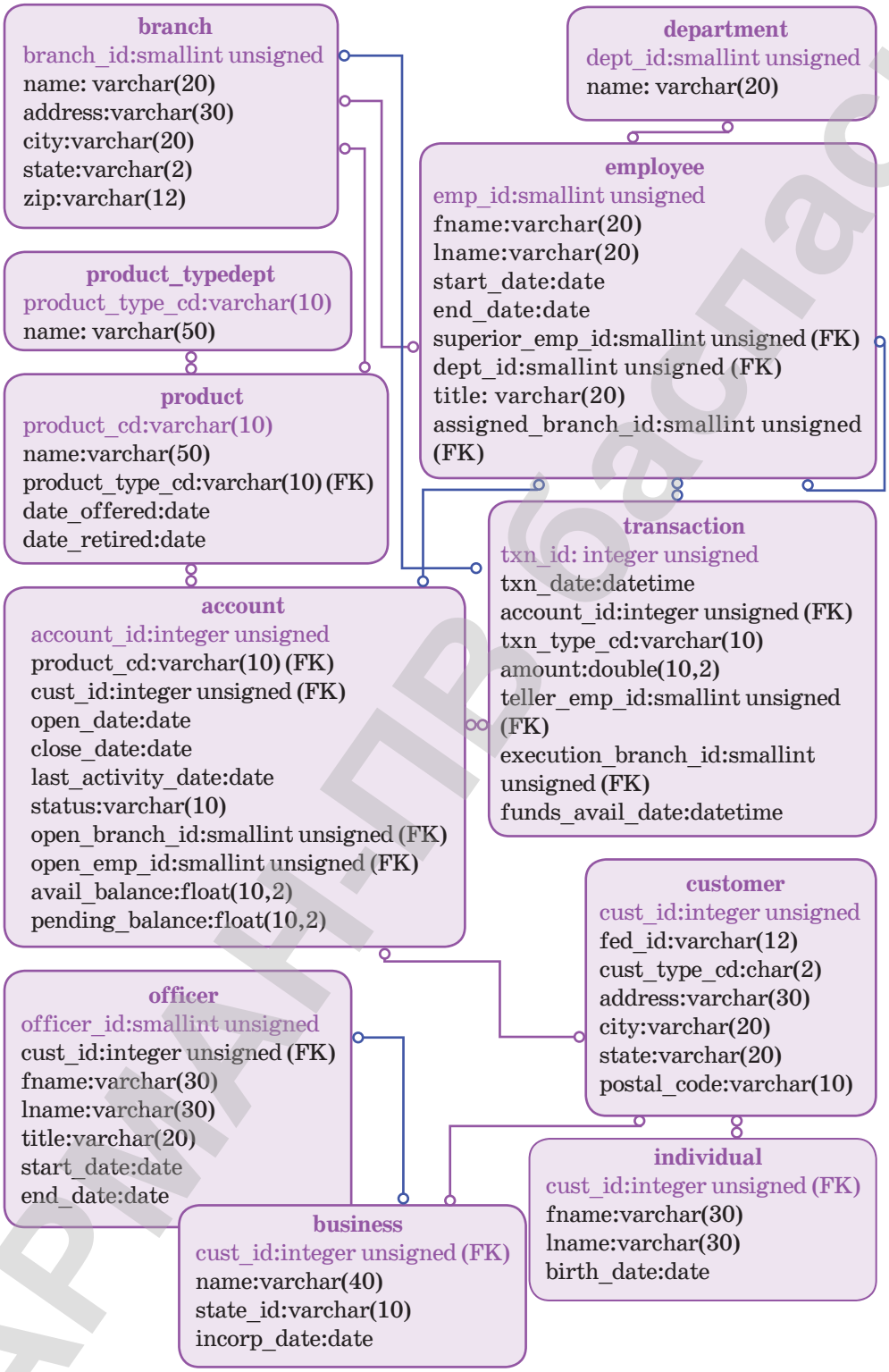


4 **Дәптерде орындайық**

Дәптерге өз сөздеріңмен деректер және олардың типтері туралы түсінікті жазыңдар.

5 **Компьютерде орындайық**

Кішігірім қоныстанған аймаққа қызмет көрсететін банкті модельдеу кестесін жасаңдар. Ол Employee (қызметкер), Branch (бөлім), Account (шот), Customer (клиент), Product (қызмет), Transaction (транзакция) және Loan (қарыз) кестелерінен тұрады делік. Кестелер, олардың бағандары мен олардың арасындағы байланыстар көрсетілген диаграмма мынадай түрде берілген:



Кестеде банк сызбасында пайдаланылатын барлық кестелер және оларға қысқаша сипаттамалар келтірілген.

Кесте	Сипаттама
Account	Нақты клиент үшін ашылған нақты шот
Business	Клиент – заңды тұлға (Customer кестесінің ішкі типі)
Customer	Банкке мәлім жеке немесе заңды тұлға
Department	Белгілі бір банк функциясын атқаратын қызметкерлер тобы
Employee	Банкте жұмыс істейтін адам
Individual	Клиент – жеке тұлға (Customer кестесінің ішкі типі)
Officer	Клиент – заңды тұлға атынан әрекет жасай алатын адам
Product	Клиенттерге ұсынылатын банк қызметі
Product, type	Функционалды ұқсас қызметтер тобы
Transaction	Шот балансының өзгеруі

6 **Ой бөлісейік**

Күнделікті өмірде өздерің деректер қорының қай типімен жұмыс істейсіңдер?

§ 61–62. Формалар

Естеріңе түсіріңдер:

- *деректер қоры жолдарының деректер типі қандай?*
- *біркестелі деректер қорын құру қалай орындалады?*
- *көпкестелі деректер қорын құру қалай орындалады?*

Меңгерілетін білім:

- *деректерді енгізуге арналған форма жасау.*

Форма – кесте мен сұраныс деректерін енгізу, бейнелеу және түзету әрекеттерін орындау үшін қолданушы интерфейсін ұйымдастыруға мүмкіндік беретін нысан. Деректерді енгізуге арналған формалар құжаттар формасын ұйымдастыруда қабылданған жобалау кезеңіндегі шаблондарға сәйкес келеді.

Формалар – ақпараттарды кестеге тасымалдаудағы «қолданушы канал» немесе деректерге қолжетімдікті қамтамасыз ететін қолданушы орта. Осылайша қолданушы өзіне таныс емес кесте, деректер сызбасы секілді нысандарды қарастырмастан, үйреншікті ортасынан қажетті ақпаратты ала алады.

Форма қажеттілігіне қарай қолданушы деректері бір немесе бірнеше қосылыстан тұрады. Сұранысқа байланысты форма арқылы сыртқы деректер қорына, мысалы, Microsoft SQL Server деректер қорына немесе деректерге қосылатын web-қызметтерге ақпарат жөнелтіледі.

Microsoft SQL Server деректер қорына негізделген форма шаблонын жасауда сұраныс өрісі мен деректер өрісінен тұратын топтастырылған деректер қоры құрылады. Осы өрістер арқылы деректер қорына негізгі қосылыс жасалып, форма шаблонына қайтарылады. Бұл өрістер мен топтар деректер қорында сақталған ақпараттар әдісіне сәйкес келеді. Қолданушы енгізген сұраныс өрісі жазбаларға сұраныс нәтижесін шектейді, ал ол өз кезегінде сұраныс өрісіндегі ақпаратпен сәйкес келеді.

SQL Server сұранысты деректер қосылысы арқылы негізгі деректер қорына жібереді. Деректер қоры сұраныс нәтижесін формаға қайтарады. Сұраныс нәтижесі өрістермен байланыстырылған басқару элементтері арқылы өзгертілетін деректер өрісіне орналасады. Сұраныс деректері мен деректер өрісінің құрылымдары деректер қорындағы ақпаратпен сәйкес келуі міндетті болғандықтан, бұл өрістерді немесе топтарды өзгертуге

болмайды. Өрістер немесе топтарды негізгі деректер қорының түпкі тобына қосуға болады.

Формаларды деректер қорына негізгі қосылыс арқылы жіберуге болады, ол үшін форма негізіндегі форма шаблоны мына талаптарды қанағаттандыруы қажет:

- Microsoft SQL Server деректерді негізгі дерек қосылысына жөнелту үшін қосылыс құрмайды. Форма арқылы қолданушыларға дерек жіберуге рұқсат беру үшін деректер қорында жұмыс істейтін web-қызметтер көмегімен қолданушыларға ақпарат жөнелтетін форма шаблонын құру керек.
- Байланысқан кесте жұбының кем дегенде бір байланысы үшін кестенің сол жағындағы бірінші баған алғашқы кілт болады.
- Егер сұраныс үлкен екілік типті деректерді сақтайтын болса, мысалы, сурет, OLE нысандары, кіріктірілген файлдар, онда Microsoft SQL Server деректер жіберу қосылысын өшіріп тастайды.

Microsoft SQL Server деректерді жіберу қосылысына рұқсат берсе, онда формалар үшін жөнелту параметрлерін баптай аламыз.

Деректер қорына негізделген формалар шаблонын жасауда web-браузерді қолдайтын форма шаблонын жасауға болады. Microsoft SQL Server сұраныс үшін форма шаблонына жалғану мақсатында қосылыс түзеді. Қолданушылар деректер қорына ақпарат жібере алуы үшін браузер қолдайтын форма шаблонын баптау мүмкін емес.

Форма шаблоны негізінде SQL Server арқылы деректер қорын құру үшін деректер қорының әкімшілігіне мынадай ақпараттар қажет:

- форма шаблонына қосылатын деректер қорынан тұратын сервер аты;
- форманың шаблонын қолданатын деректер қорының аты;
- деректер қорында тіркеуді тексеру. Деректер қорына қолданушылардың қолжетімдік әдісін тексеру үшін Microsoft Windows немесе SQL Server қызметтерін қолдануға болады.
- ақпарат жібергелі отырған кесте аты немесе формадан ақпарат алатын кесте. Бұл басты кесте болып табылады. Егер деректер қорында бірнеше кесте қолданатын болса, онда басқа қолданушылардың аты, ішкі кестелер қажет

болады. Сонымен қатар ішкі кестедегі өріс аттары, басты кестемен байланысы керек.

Форма шаблонын құру

Сұраныс ақпаратына сәйкес форма шаблонын құру үшін мынадай әрекеттерді орындау қажет:

1. Форма шаблонын құру. Microsoft SQL Server шаблон формасы негізінде деректер қорын құруда форма шаблоны мен деректер қоры арасында ақпарат қосылысы құрылады. Бұл үрдіс автоматты түрде негізгі деректер қорының форма шаблонын құрады.
2. Форманы ашқан кезде қолданушыларға өрістерді қарап, оларға өзгерістер енгізуге рұқсат беру үшін форма шаблонына басқару элементін қосып, оны негізгі деректер қорымен байланыстыру қажет.

1 **Сұрақтарға жауап берейік**

1. Форма деген не?
2. Форма арқылы сұраныс жасау қалай орындалады?
3. Деректер қорын құру үшін қандай ақпарат қажет?
4. Форма қандай қосылыстан тұрады?
5. Форма шаблонын құру үшін не істеу керек?

2 **Ойланайық, талқылайық**

1. Форма құрудың басты ерекшелігі неде?
2. Деректерді енгізуге арналған форма жасаудың қажеттілігі неде?
3. Форма шаблонын құруда қосылыс түзудің маңыздылығы қандай?

3 **Талдап, салыстырайық**

Форма шаблонын құру ерекшеліктерін төмендегі кестеге толтырыңдар. Оларды талдап, салыстырыңдар.

Орындалатын әрекет	Ерекшелігі
Форма шаблонын құру	
Сұраныс нәтижесін бейнелеу үшін бір немесе бірнеше басқару элементтерін қосу	

4 Дәптерде орындайық

Форманың шаблонына сәйкес келетін талаптарды жазыңдар.

1-талап: ...
2-талап: ...
3-талап: ...
4-талап: ...

5 Компьютерде орындайық

Форма шаблонын құрыңдар.

1. Файл мәзірінен Форма шаблонын құру (Создание шаблона формы) жолын таңдаңдар.
2. Жаңасын құру (Разработка нового) бөлімінде, Форма шаблонын құру (Создание шаблона формы) сұхбат терезесінде Форма шаблонын шертіңдер.
3. Негізінде тізімінен Деректер қорын таңдаңдар.
4. Егер браузер қолдауымен форма шаблонын құратын болсақ, онда Қараушымен үйлесетін мүмкіндіктерді ғана қосу (Включить только возможности, совместимые с обозревателем) өрісіне жалауша қойыңдар.
5. ОК батырмасын басыңдар.
6. Деректерді қосу шеберінде Деректер қорын таңдау (Выбор базы данных) батырмасын басыңдар.
7. Ақпарат көзін таңдау (Выбор источника данных) өрісінен Қор құру (Создать источник) жолын таңдаңдар.
8. Қосылу керек деректер қорының типін таңдау (Выберите тип источника данных, к которым нужно подключиться) тізімінен Microsoft SQL Server жолын таңдап, Әрі қарай (Далее) батырмасын басыңдар.
9. Сервер аты өрісіне SQL Server деректер қорындағы сервер атын енгізіңдер.
10. Есептік деректер кіріс (Вход учетные данные) жолында төмендегі әрекеттердің бірін орындаңдар:
    a. Егер деректер қоры Microsoft Windows желісінде есептік ақпараттар негізінде қолжетімдікке кім ие екендігін анықтаса, онда Windows тіркелуді тексеруді қолдану (Использовать проверку подлинности Windows) батырмасын басыңдар.
    b. Егер деректер қоры берілген қолданушы аты мен құпиясөз арқылы қолжетімдікке кім ие екендігін

анықтаса, онда Қолданушы аты мен құпиясөзді қолдану (Использовать следующее имя пользователя и пароль) жолына жалауша қойыңдар, содан кейін сәйкес өрістерге қолданушы аты мен құпиясөзді енгізіңдер.

11. Әрі қарай (Далее) батырмасын басыңдар.
12. Керекті ақпараттан тұратын деректер қорын таңдаңдар (Выберите базу данных, которая содержит нужные данные) жолынан деректер қоры атын таңдап, белгілі кестеге қосылу өрісіне жалауша қойыңдар, басты кесте атын шертіп, Әрі қарай (Далее) батырмасын басыңдар.
13. Шебердің келесі бетінде деректер қосылысы туралы өрісіне файл атын енгізіңдер де, осы параметрлерді сақтау үшін Дайын батырмасын шертіңдер.
14. Әрі қарай (Далее) батырмасын басыңдар.
15. Шебердің соңғы бетінде деректердің негізгі қосылысына файл атауын енгізіңдер. Бұл атау деректер қоры тізімінде тұрады.

6 Ой бөлісейік

Деректерді енгізуге арналған формалар ереже бойынша құжаттар формасын ұйымдастыруда қабылданған жобалау кезеңіндегі шаблондарға сәйкес келмейтін жағдай орын алса қандай әрекеттер орындалады? Өз ойларыңды сыныптастарыңмен бөлісіңдер.

§ 63–64. Сұраныстар

Естеріңе түсіріңдер:

- *форма деген не?*
- *деректерді енгізуге арналған форманы қалай құруға болады?*

Меңгерілетін білім:

- *алынған деректерді пайдаланып, сұраныс жасау.*

MySQL сервері сұраныстарды қалай орындайды?

Сұраныс – деректер қорынан белгілі бір ақпаратты алу талабы. Mysql командалық жолының клиенттік программасы деректер қорына кіру кезінде қолданушы аты мен құпиясөз дұрыстығы тексеріледі, содан кейін деректер қорымен қосылу орындалады.

Сервер қосылысты іске қосқаннан кейін сұраныс жұмыстарын жасауға болады. Әрбір сұранысты орындамас бұрын сервер мыналарды тексереді:

- Өрнекті орындауға рұқсатыңыз бар ма?
- Қажетті деректерге қол жеткізуге рұқсатыңыз бар ма?
- Өрнек синтаксисі дұрыс па?

Егер өрнек осы үш тесті өтетін болса, онда ол *сұраныс оптимизаторына* беріледі. Сұраныс оптимизаторының жұмысы сұранысты орындаудың тиімді әдісін анықтау болып табылады. Оптимизатор сұраныста келтірілген кесте байланысын реті мен қолжетімді индекстерді қарастырады, ал сосын осы сұранысты орындауда сервер қолданатын *орындау жоспарын* анықтайды.

Сұранысты орындап болғаннан кейін, сервер шақыратын қосымшаны нәтиже жиынтығына қайтарады.

Сұраныс блогы

`select` өрнегі бірнеше құраушыдан немесе блоктардан (`clauses`) тұрады. MySQL-мен жұмыс кезінде олардың біреуі ғана міндетті болып табылады. Сұраныс кезінде қолжетімді блоктардың алтауының ішіндегі кем дегенде екі-үшеуі қосылады. *22-кестеде* түрлі блоктар мен олардың атқаратын қызметі келтірілген.

***22-кесте.** Сұраныс блоктары*

Блок	Қызметі
`Select`	Сұраныстың нәтиже жиынтығына қосылатын бағандарды анықтайды

Блок	Қызметі
From	Ақпарат алынатын кестелерді, олар қалай байланысу керектігін көрсетеді
Where	Соңғы нәтиже жиынтығындағы жолдар санын шектейді
Group by	Бағандардың бірдей мәні бойынша жолдарды топтау үшін қолданылады
Having	Соңғы нәтижелі жиынтықта деректерді топтау арқылы жолдар санын шектейді
Order by	Бір немесе бірнеше баған бойынша соңғы нәтиже жиынтығының жолдарын сұрыптайды

select блогы

select блогы ең алғашқы блок болғанымен, деректер қорының сервері оны ең соңынан өңдейді. Себебі таңдау жұмысын орындау үшін алдымен жиынтыққа кіретін барлық кестелер мен ондағы дерек түрлерін білу қажет. Сондықтан select блогының қызметімен танысу үшін алдымен from блогының қызметімен танысу керек. Сұраныс жасау түріне мысал келтіретін болсақ:

```
Mysql > SELECT *
   > FROM department;
```

dept_id	name
1	Operations
2	Loans
3	Administration

3 rows in set (0.04 sec)

Бұл сұраныста *from* блогында тек қана бір кесте (*department*) көрсетіліп тұр және *select* блогында *department* кестесіндегі барлық бағанды («*» символымен белгіленген) қарау керектігі сұралған. Бұл сұранысты өз тілімізде былайша айтуға болады:

department кестесіндегі барлық бағанды маған көрсет

Олай болса, *select* блогының қызметі – барлық мүмкін бағандар сұраныс қажеттілігін қанағаттандыратындығын анықтау.

from блогы

from блогы сұраныс қолданатын кестені, сонымен қатар кестелер байланысы құралын анықтайды.

table термині деректер қорында сақталатын кестедегі өзара байланысқан жолдар жиынтығын береді. Бұл еркін анықтамаға кестенің үш түрлі типі кіреді:

- тұрақты кестелер (*create table* өрнегінің көмегімен құрылған кесте);
- уақытша кестелер (сұраныспен қайтарылған жолдар);
- виртуалды кестелер (көріністер) (*create view* өрнегінің көмегімен құрылған кестелер).

Кестелердің әрбір типі сұраныстың *from* блогына қосылады.

where блогы

where блогы – нәтижелі жиынтық үшін қажет емес жолдарды қарастырмау механизмі.

Мысалы, *employee* кестесінен ақпарат алу керек болсын, ол дерек тек қана аға қызметкерлер ретінде жұмыс істейтін қызметкерлерге қатысты. Келесі сұраныста *where* блогы тек қана төрт аға қызметкер туралы ақпарат алу үшін жасалған сұраныс:

```
mysql> SELECT emp_id, fname, lname, start_date,
title
   > FROM employee
   > WHERE title = 'Head Teller';
```

emp_id	fname	lname	start_date	title
6	Helen	Fleming	03 10 2019	Head Teller
10	Paula	Roberts	03 10 2019	Head Teller
13	John	Blake	03 10 2019	Head Teller
16	Theresa	Markham	03 10 2019	Head Teller

4 rows in set (0.00 sec)

group by және having блоктары

Бұған дейінгі қарастырылған сұраныстар ешқандай әрекет орындамаған, өңделмеген жолдарды шығарды. Алайда бізге жиынтық деректі бермес бұрын жалпы бағытты анықтап алу қажет болады. Сондай жағдайда баған мәні бойынша деректерді топтау үшін *group by* блогы қолданылады. Мысалы, қызметкерлер мен олар қызмет ететін бөлім орнына әрбір бөлімде жұмыс жасайтын қызметкерлер тізімі болсын. *group by* блогымен қатар *having* блогын да қолдануға болады, олар *where* блогы секілді өңделмеген ақпаратты сұрыптауға мүмкіндік береді.

order by блогы

Жалпы жағдайда сұраныс нәтижесін беретін жолдар түрлі ретпен шығарылады. Егер сұраныс нәтижесін белгілі бір жолмен реттеу қажет болса, онда серверге *order by* блогы көмегімен сұрыптау өрнегін қосу керек.

order by блогы – берілген баған, бағанда қолданылатын өрнек бойынша нәтижені сұрыптау механизмі.

open_emp_id	product_cd
10	CHK
10	SAV
10	CD
10	CHK
10	SAV
13	CHK
13	MM
1	CHK
1	SAV
1	MM
16	CHK
1	CHK
1	CD
10	CD
16	CHK
16	SAV
1	CHK
1	MM
1	CD
16	CHK
16	BUS
10	BUS
16	CHK
13	SBL

24 rows in set (0.00 sec)

open_emp_id	product_cd
1	CHK
1	SAV
1	MM
1	CHK
1	CD
1	CHK
1	MM
1	CD
10	CHK
10	SAV
10	CD
10	CHK
10	SAV
10	CD
10	BUS
13	CHK
13	MM
13	SBL
16	CHK
16	CHK
16	SAV
16	CHK
16	BUS
16	CHK

24 rows in set (0.00 sec)

Өсу ретімен және кему ретімен сұрыптау

Сұрыптау кезінде өсу реті бойынша (*ascending*) немесе кему реті бойынша (*descending*) сұрыптау қызметтерін *asc* және *desc* қызметші сөздері арқылы қосуға болады.

Үнсіз келісім бойынша өсу реті бойынша сұрыптау орындалады, сондықтан егер кему реті бойынша сұрыптау қажет болса, онда *desc* кілттік сөзін қосуға болады. Мысалы, төмендегі сұраныс бойынша есепшоттағы қалдық бойынша кему ретімен көбінен бастап азына қарай сұрыпталады:

```
mysql > SELECT account_id, product_cd, opendate,
avail_balance
  > FROM account
            > ORDER BY avail_balance DESC;
```

account_id	product_cd	open_date	avail_balance
24	SBL	03 10 2019	50000.00
23	CHK	03 10 2019	38552.05
20	CHK	03 10 2019	23575.12
13	CD	03 10 2019	10000.00
22	BUS	03 10 2019	9345.55
18	MM	03 10 2019	9345.55
10	MM	03 10 2019	5487.09

1 Сұрақтарға жауап берейік

1. Сұраныс деген не?
2. Сұраныс жасау үшін қандай шарттар қанағаттандырылады?
3. Қандай сұраныс блоктары бар?
4. `Select` блогының қызметі қандай?
5. `From` блогы қандай қызмет атқарады?
6. `Where` блогы қандай қызмет атқарады?
7. `Group by` мен `Having` блоктары қандай қызмет атқарады?
8. `Order by` блогы қандай әдістер бойынша сұрыптайды?

2 Ойланайық, талқылайық

1. Деректер қорында сұраныс қызметтері не себепті қолданылады?
2. Сұраныс блоктарының түрлі шарттарды қанағаттандыру бойынша қажеттілігі неде?

3 Талдап, салыстырайық

Сұраныс блоктары мен олардың атқаратын қызметін талдап, өзара салыстырыңдар:

Блок	Қызметі
`Select`	
`From`	
`Where`	
`Group by`	
`Having`	
`Order by`	

4 Дәптерде орындайық

Өсу ретімен сұрыптау (ascending) мен кему ретімен сұрыптау (descending) жұмыстарын орындау үшін қолданылатын қызметші сөздер туралы толық ақпарат пен мысал келтіріңдер.

5 Компьютерде орындайық

1-тапсырма. Қызметкерлер кестесінен барлық қызметшілердің ID, атын, тегін шығаратын сұраныс жасаңдар. Тегі бойынша, аты бойынша сұрыптау жұмыстарын орындаңдар.

2-тапсырма. Есепшот ID-ін, клиент ID-ін және есепшоттағы барлық 'ACTIVE' статусындағы 2500 теңгеден жоғары қалдықтарды шығаратын сұраныс жасаңдар.

3-тапсырма. account кестесіне есепшот ашқан қызметкерлер (account.open_emp_id) ID-ін шығаратын сұраныс жасаңдар.

4-тапсырма. Төмендегі сұраныс бойынша бос орындарды толтыруға арналған сұраныс жасаңдар:

```
mysql > SELECT p.product_cd, a.cust_id, a.avail_
         balance
      > FROM product p INNER JOIN account
      > ON p.product_cd = <2>
      > WHERE p. <3> = 'ACCOUNT';
```

6 Ой бөлісейік

Деректер қоры бойынша орындалатын сұраныстардың тиімділігі мен қажеттілігі қаншалықты маңызды екендігі туралы өз сыныптастарыңмен ой бөлісіңдер.

§ 65–66. Есептер

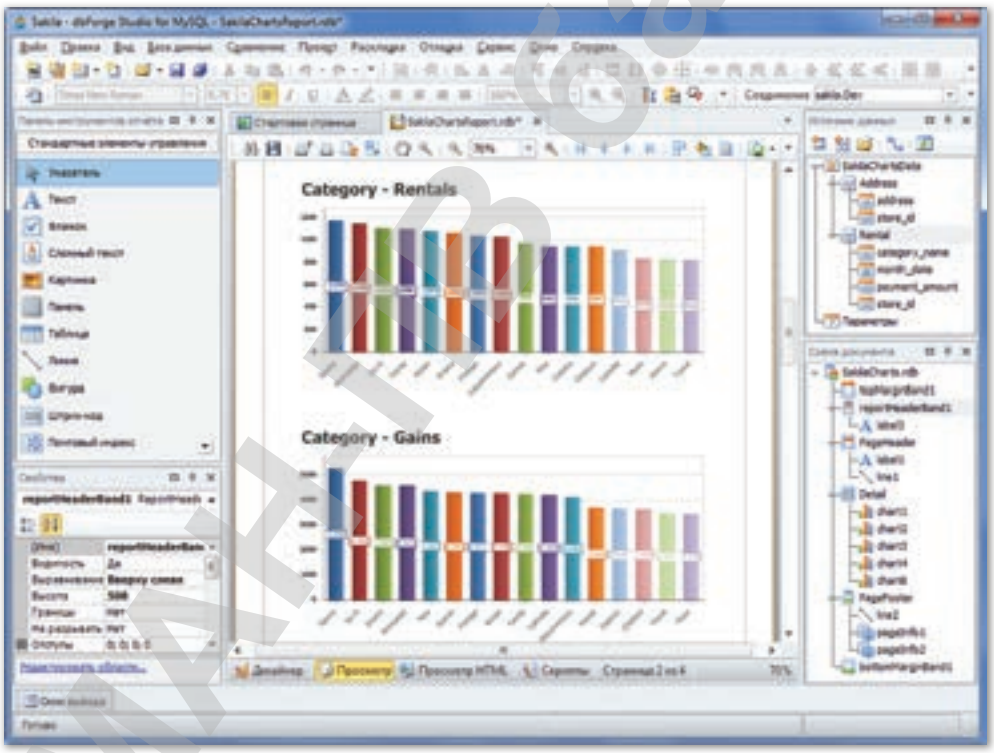
Естеріңе түсіріңдер:

- *алынған деректерді пайдалана отырып, сұраныстарды қалай жасауға болады?*
- *сұраныстың қандай блоктарын білесіңдер?*

Меңгерілетін білім:

- *алынған деректерді қолдана отырып, есептер жасау.*

Есеп – экран беті, принтер секілді құрылғыларға немесе файл ретінде шығарылатын бір немесе бірнеше кесте деректерінің қосындысы. Есеп негізін деректер қоры кестесінің жазбалары мен сұраныстары құрайды. Есепке сызбалар мен диаграммалар, суреттер мен бейнелер, графикалық басқару элементтері, қызметтік ақпараттардан тұратын жоғары және төменгі колонтитулдар жатады (*27-сурет*).



27-сурет. *Есеп түрі*

Деректер қорын басқару жүйесі деректерге қатынасып, бір емес бірнеше қолданушыға бір уақытта өзгерте беруіне рұқсат береді. Егер әрбір қолданушы сұранысты орындайтын болса, бұл көп уақыт алған болар еді, ал деректер қорының сервері

үшін ол ешқандай қиындық тудырмайды. Дегенмен кейбір қолданушылар өзгеріс енгізіп, қосымша дерек қосқысы келсе, онда серверге аралық нәтижелерді қосу керек болады.

Мысалы, есепшоттағы барлық қалдық сомасы туралы ақпарат есебі құрылады. Алайда бұл есеп түрі құрылып жатқан кезде төмендегі әрекеттер орындалып жатыр:

- банк қызметкері бір тұтынушы үшін қор құрып жатыр;
- тұтынушы банкоматтан ақшасын алды;
- банк қосымшасы әрбір айдың соңында есепшоттағы сомасына қарай пайыздық үстемесін аударды.

Осылайша, есеп құрылып жатқан кезде бірнеше қолданушы өз деректерін өзгертіп жатады. Есеп түрі қандай болу керек? Сұрақтың жауабы сервердің бұғаттан қалай шыққанына байланысты. Бұғат (`locking`) – деректер қорын бір мезгілде қолдануды басқару механизмі. Деректер қоры серверінің көпшілігі бұғаттаудың екі жолының бірін қолданады.

1. Деректер қорында жазбаны жазатын қолданушылар жазбаны бұғаттан шығаруды (`write lock`) серверден сұрап, деректерді өзгерту үшін серверден рұқсат алуы керек. Ал деректер қорынан ақпарат алатын қолданушылар сұранысты қанағаттандыру үшін серверден оқуды бұғаттан шығаруға (`read lock`) рұқсат алу керек. Бірнеше қолданушы тарапынан оқу бір мезгілде жүзеге асырылатындықтан, әрбір кесте үшін бір жазба ғана бұғаттан шығарылады және оқу сұранысы жазба бұғаты алынғанға дейін бұғатталып тұрады.

2. Деректер қорына жазба жазатын қолданушылар ақпаратты өзгерту үшін серверден жазба бұғатын алуды сұрап, одан рұқсат алу керек, ал деректерді алатын қолданушылар ақпарат сұранысы үшін бұғаттың ешқандай типін қажет етпейді. Оның орнына сервер оқырманның ақпаратты ұсынуда ешқандай қарама-қайшылық болмайтындығына кепілдік береді (ақпарат өзгеріссіз ұсынылады). Бұл әдіс нұсқаны бақылау (`versioning`) әдісі ретінде белгілі.

Екі әдістің де өз артықшылықтары мен кемшіліктері бар. Бірінші әдіс бойынша егер бір уақытта оқуға да, жазуға да сұраныс түссе, онда күту уақыты ұзақ болып көрінуі мүмкін. Екінші әдіс бойынша ұзақ уақыт алатын сұраныстар мәселе тудыруы мүмкін, себебі деректер өзгертіліп жатыр.

Ресурстарды бұғаттаудың түрлі әдістері бар. *7-сызбада* ұсынылған үш деңгейдің бірімен бұғаттау орындалуы мүмкін.

7-сызба. *Ресурстарды бұғаттаудың әдістері*

Есептің беттеріндегі ақпараттар деректер қорының күйін білдіреді немесе есеп құрғанға дейінгі жағдайын, сервердің оқуды бұғаттау кездегі күйін көрсетеді.

Есеп құруға арналған жүйелерде немесе деректер қорында мынадай сұраныс түрлері көп кездеседі:

```
mysql > SELECT p.name product, b.name branch,
      > CONCAT(e.fname,' ',e.lname) name,
      > SUM(a.avail_balance) tot_deposits
      > FROM account a INNER JOIN employee e
      > ON a.open_emp_id = e.emp_id
      > INNER JOIN branch b
      > ON a.open_branch_id = b.branch_id
      > INNER JOIN product p > ON a.product_cd =
      p.product_cd
      > WHERE p.product_type_cd = 'ACCOUNT'
      > GROUP BY p.name, b.name, e.fname, e.lname;
```

product	branch	name	tot_ deposits
certificate of deposit	Headquarters	Michael Smith	11500.00
certificate of deposit	Woburn Branch	Paula Roberts	8000.00
checking account	Headquarters	Michael Smith	782.16
checking account	Quincy Branch	John Blake	1057.75
checking account	So. NH Branch	Theresa Markham	67852.33
checking account	Woburn Branch	Paula Roberts	3315.77
money market account	Headquarters	Michael Smith	14832.64
money market account	Quincy Branch	John Blake	2212.50
savings account	Headquarters	Michael Smith	767.77
savings account	So. NH Branch	Theresa Markham	387.99
savings account	Woburn Branch	Paula Roberts	700.00

11 rows in set (0.02 sec)

Есеп құру

Есеп құру барысында ескерілетін екі маңызды дағдыға төмендегілер жатады:

- есеп құру механизмінің мүмкіндіктерін білу;
- деректер қорының серверін қолданатын SQL-ді жүзеге асыруды жетілдіру.

Есеп құру құралдарының көпшілігі SQL арқылы құруды қолдайды. Бұл жағдайда деректер қорының серверіне не жөнелтіліп жатқанын білеміз және нәтижесінде алынған есепті қолдап немесе баптай аламыз. SQL-ді толық түсіну, ішкі сұраныстар қызметін, жиындар мен логика шарттарымен орындалатын әрекеттерді білу арқылы толық есеп құруға және алуға болады.

1 **Сұрақтарға жауап берейік**

1. Есеп деген не? Есептің кеңінен қолданылуын қай салада байқадыңдар?
2. Есептің құрамына нелер кіреді?
3. Есеп құруда бұғат қандай рөл атқарады?
4. Ресурстарды бұғаттаудың қандай әдістері бар?

2 **Ойланайық, талқылайық**

1. Деректер қорында есеп қызметтерін қолдану қаншалықты маңызды?
2. Есеп беттеріндегі ақпараттар деректер қорының күйіне тәуелді ме?
3. Неліктен есептің бірнеше түрлері қолданылады?

3 **Талдап, салыстырайық**

Есеп түрінің сервердің бұғаттан қалай шыққанына байланыстылығын көрсететін екі әдісті талдап, артықшылықтары мен кемшіліктерін салыстырыңдар.

4 **Дәптерде орындайық**

Есеп құруға арналған жүйелерде немесе деректер қорында мынадай сұраныс түрін талдап, әрбір жолдың атқаратын қызметі туралы ақпарат келтіріңдер.

```
mysql > SELECT p.name product, b.name branch,
    > CONCAT(e.fname,' ',e.lname) name,
    > WHERE p.product_type_cd = 'ACCOUNT'
    > GROUP BY p.name, b.name, e.fname, e.lname;
```

Есеп үшін есеп моделін құрыңдар.

1. MS SQL Server-ді ашыңдар.
2. Файл мәзірінен Құру жолына өтіп, Жоба өрісін таңдаңдар.
3. Шаблондардан Есеп моделінің жобасы жолын таңдаңдар.
4. «Атауы» өрісіне Есеп моделінің үлгісін енгізіңдер.
5. ОК батырмасын басыңдар.
6. MS SQL Server терезесінің Шешімдер шолушысынан тінтуірдің оң жақ батырмасын басып, Жаңа деректер қорын қосу жолын таңдаңдар. Әрі қарай батырмасын басыңдар.
7. Қосылысты анықтау әдісін таңдау бетінде Деректер қорын құру жолы таңдалып тұрғанына назар аударыңдар. Әрі қарай Құру батырмасын басыңдар.
8. Қосылыстар диспетчері сұхбат терезесінде қосылудың берілген қасиеттерін орнатыңдар:
   - Сервер аты – MS SQL Server деректер қорының атын таңдаңдар немесе ашылған тізімнен таңдаңдар.
   - Windows тіркеуін тексеруді қолдану жолын таңдаңдар.
   - Ашылған тізімнен деректер қорының атын таңдаңдар немесе енгізіңдер.
9. Деректер қорының қосылу жұмыс қабілетін тексеру үшін Қосылуды тексеру жолын белгіленңдер.
10. Егер желіге қосылса, онда ОК батырмасын басыңдар. Ал желіге қосылмаса, онда енгізілген деректердің дұрыстығын, қосылуды қайта тексеріңдер.
11. Қосылымды анықтау әдісін таңдау бетінде бар немесе жаңа қосылым негізінде деректер көзін жасау деп таңдалғанына көз жеткізіңдер, деректер қосылымының тізімінде көрсетілген деректер көзі таңдалғанына көз жеткізіңдер, одан кейін Келесі батырмасын басыңдар.
12. Деректер қорының аты өрісіне Есеп моделінің мысалы жолын енгізіп, Дайын батырмасын басыңдар.
13. Шешімдер шолушысынан тінтуірдің оң жақ батырмасын басып, Жаңа деректер қорын қосу командасын таңдаңдар.
14. Әрі қарай батырмасын басыңдар.
15. Деректер қорын таңдау бетіндегі Деректердің реляциялық қоры терезесінен Есеп моделінің мысалы жолы таңдалып тұрғандығына назар аударыңдар, Әрі қарай батырмасын басыңдар.

16. Кесте таңдау бетінде есеп моделі үшін қолданылатын MS SQL Server деректер қорының кестесін таңдаңдар. Әрі қарай батырмасын басыңдар.
17. Шебердің жұмысын аяқтау бетінде есеп моделін таңдап, Дайын батырмасын басыңдар.
18. Шешімдер шолушысында тінтуірдің оң жақ батырмасын басып, Есептің жаңа моделін қосу өрісін таңдаңдар.
19. Әрі қарай батырмасын басыңдар.
20. Деректер қорын ұсынуды таңдау тізімінен Қолжетімді деректер қоры жолын таңдап, Әрі қарай батырмасын басыңдар.
21. Есеп моделін құру ережесін таңдау бетінде үнсіз келісім бойынша тұрған көрсеткіштерді өзгеріссіз қалдырып, Әрі қарай батырмасын басыңдар.
22. Шебердің жұмысын аяқтау бетінде Аты өрісінде Есеп моделінің мысалы тұрғандығына назар аударыңдар.
23. Жұмысты аяқтап, Есеп моделін құру үшін Орындау батырмасын басыңдар.

6 Ой бөлісейік

Деректер қорын басқару жүйесі деректерге қатынасып, бір немесе бірнеше қолданушағы бір уақытта өзгерте беруіне рұқсат беру қызметі туралы не ойлайсыңдар? Сыныптастарыңмен өз ойларыңды бөлісіңдер.

§ 67–68. Деректер базасымен web-беттердің байланысы

Естеріңе түсіріңдер:

- *сұраныс түсінігі;*
- *сұраныс блоктары;*
- `Select` *блогының қызметі;*
- `From` *блогының қызметі;*
- `Group by` *мен* `Having` *блоктары;*
- `Order by` *блогы.*

Меңгерілетін білім:

- *MySQL деректер қоры;*
- *MySQL-дің артықшылығы мен кемшілігі;*
- *PHP тілі.*

Терминдер:

- MySQL;
- PHP.

PhpMyAdmin – PHP тілінде жазылған web-программа және MySQL жүйесін басқаруға арналған web-интерфейс. Ол арқылы браузерді пайдаланып, MySQL серверін басқаруға, SQL командаларын орындауға, деректер қорындағы кестелердегі жазбаларды өңдеуге болады.

PhpMyAdmin-нің кең қолданылатын себебі – осы интерфейс арқылы SQL операторларын қолмен жазып отырмай-ақ **MySQL** жүйесін оңай басқаруға болады.

PhpMyAdmin көп жұмысты және шеберлікті қажет етпейтін деректер қорын басқару. MySQL – дерекқорды басқарудың еркін жүйесі әрі – шағын және орта бизнеске арналған шешімдер.

MySQL деректер қорының сервері – үлестірілген деректер қорының жылдам және мықты басқару жүйесі. Ол ақпаратты тиімді түрде сақтауға, іздеуге, сұрыптауға және таңдауға мүмкіндік береді.

MySQL – кіші және орта программалар үшін нағыз сәтті жасалған дүние. Ол көп ағынды сүйемелдеуі бар UNIX-серверлерде өз мүмкіндіктерін толық көрсете алу нәтижесінде өте үлкен көрсеткіштерге ие болуда. MySQL сервер коммерциялық емес бағытта қолдануда тегін болып саналады.

MySQL-дің мүмкіндіктері

MySQL ANSI 92 стандартындағы SQL сұраныстарын қолданады. Оның мынандай мүмкіндіктері бар:

1. Саны шектеусіз пайдаланушылар деректер қорымен бір уақытта жұмыс істей алады.

2. Кестедегі жолақтар саны 50 млн-ға дейін жете алады.
3. Командалар өте тез орындалады. Осы күнгі серверлердің ішінде MySQL ең жылдамы болып есептеледі.
4. Қарапайым және нәтижелі қауіпсіз жүйе.

Артықшылықтары:

– Web-қосымшаларының файлдық нұсқаларына қарағанда кодтың әлдеқайда кішігірім болуы (2–3 есе). Оның құрастыру уақытын үнемдеп, өңдеу үрдісін жеңілдетеді;
– С тілінде жазылғандықтан, МҚБЖ сұраныс процедураларының жоғары жылдамдығымен орындалуы.

1-мысал. PHP сценарийлерінде MySQL деректер қорын және кестелерді құру.

PHP сценарийлерінде орындалған MySQL сұрауларын пайдаланып, деректерді басқаруға көшу:

Ол үшін Z локальді дискісінде:/ WebServers/home/localhost /www/my_phone create.php файлын құрамыз.

1. Блокнотқа мына кодты теремiз.

```
   <html> Абай атындағы №165 жббм <body><br>
информатика кафедрасы <br> <br>
<hr>
<b> зертханалық жұмыс </b><br>
<?php
$key1 =0; echo "<b> mysql серверіндегі деректер
қорын құру </b>"."<br>"."<hr>";
$sdb_name = "localhost";
$user_name="root"; $user_pass = " ";
$db_name = "phone";
// сервермен байланыс
$link=mysql_connect($sdb_name,$user_name,$user_
pass); echo "report mysql-server"."<br>";
if (!$link)

{echo "mysql-server-мен байланыс жоқ <br>";exit();}
echo "mysql-server-мен байланыс орнады<br> <hr>";
// деректер қорын құру
```

```
$str_sql_query = "create database $db_name"; echo
"message:".$str_sql_query;
if (!mysql_query($str_sql_query, $link))
{echo "<br> деректер қорын құрмадыңыз!!! <br>";}
// деректер қорын таңдау
if   (!mysql_select_db($db_name,$link))   {echo
"Деректер қорын таңдамадыңыз - табылмады."."<br>";
exit();} echo "<br> Деректер қорын таңдадыңыз
<br>";
// Кесте құру
mysql_query("create table tb1(nom text(20), fam
text(50))") or die ("<br> Кесте құруда қателік
бар <br>". mysql_error()); mysql_close($link);
?> </body> </html>
```

2. Start Denwer іске қосамыз.
3. Браузердің мекенжай жолынан сценарийді шақырамыз. (мысалы: localhost / MY_PHONE / create.php), оның жұмысын тексереміз.
4. Берілген деректер қорына, кестеге және өріс аттарына MySQL деректер қорына кіріп, MY_PHONE1 бумасында жаңа жоба үшін create1.php сценарийін іске қосамыз.
5. Сценарий мәтіндерін Word программасынан көшіргенде, Word программасындағы ұзын жолдың мәтіні автоматты түрде келесі жолға ауыстырылатынын есте сақтаңдар.
6. Басынан бастап [;] белгісіне дейін Сценарийде бұл бір (!) жол:

2-мысал. Дерекқор жазбаларымен өзара әрекеттесуі

```
if (!mysql_select_db($db_name,$link))
{echo "Not find for use data base"."<br>"; exit();
}
echo "Open data base - Yes"."<br>";
// Кестеге сұраныс
$str_sql_query= "SELECT * FROM Tb1";
// Кестеге сұраныс
if (!$result=mysql_query($str_sql_query,$link))
{
```

```
echo "Not RUN query Tb1"."<br>"; exit();
} echo "Query Table data base - Yes"."<br>";

$newnom='203040';
$newfam='Тегі5';
// Деректер енгізу
//
   $str_sql_query="INSERT INTO Tb1 SET
   Nom='Жаңа нөмір',
   Fam='Жаңа тегі'";
//
   $str_sql_query="INSERT INTO Tb1 SET
        Nom='$newnom', Fam='$newfam'";
//
//$str_sql_query="INSERT INTO Tb1(nom, fam)VALUES
('$newnom','$newfam')";
// Деректерді жаңарту- Нөмір өзгерту
   // $str_sql_query="UPDATE Tb1     SET fam =
   'Ауысты' WHERE nom='55555555'";
// Деректерді жою
//   $str_sql_query="DELETE   FROM   Tb1   WHERE
nom='55555555'";
(!mysql_query($str_sql_query,$link))
```

1 **Сұрақтарға жауап берейік**

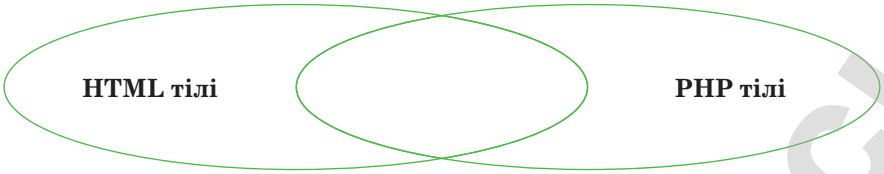
1. MySQL деген не?
2. Php тілі қайда қолданылады?
3. Қандай кең тараған web браузерлерді білесіңдер және өздерің қандай түрін қолданасыңдар?

2 **Ойланайық, талқылайық**

1. MySQL деректер қорының PHP тілімен қандай байланысы бар?
2. Не себепті PhpMyAdmin көп қолданылады?

3 **Талдап, салыстырайық**

HTML тілі мен PHP тілін салыстырыңдар.



4 Дәптерде орындайық

Access, MySQL деректер қоры туралы ақпарат тауып, талдап, төмендегі кестеге олардың сипаттамаларын жазыңдар.

Деректер қорында сұраныс құру	
Access	MySQL

5 Компьютерде орындайық

PHP-ді пайдаланып, Web-беттің деректер қорына қолжеткізу үшін мына әрекеттерді орындау қажет:

1. MySQL серверін қосылу.
2. Деректер қорын таңдау.
3. Деректер қорына сұранысты ұйымдастыру:
   - қосу;
   - өшіру;
   - өзгерту;
   - іздеу;
   - сұрыптау.
4. Сұраныс нәтижесін алу.
5. Деректер қорынан ажырау.

Деректер қорының серверіне қосылу үшін PHP-де `mysql_connect( )` функциясы қолданылады. Функцияның аргументі: компьютер аты, қолданушы аты және құпиясөз. Үнсіз келісім бойынша компьютерді аты `= localhost`, онда қолданушы аты мен құпиясөз қажет болмайды. Егер барлығы табысты өтсе, `mysql_connect()` және `mysql_pconnect()` функциялары қосылу идентификаторын қайтарады.

Мысалы:

```
$link = mysql_pconnect ();
if ( !$link ) die ("MySQL-ге қосылу мүмкін емес");
```

MySQL серверімен байланыс орнатылғаннан кейін, қажетті деректер қорын таңдау керек. Ол үшін `mysql_select_db()` функциясы қолданылады. Оның аргументі: деректер қорының аты. Егер таңдалған деректер қоры бар және қолжетімді болса, функция `true` мәнін қайтарады.

```
$db = "sample";
mysql_select_db ( $db ) or die ("$db ашу мүмкін
   емес");
```

Қосу, өшіру, өзгерту және деректерді таңдау үшін SQL сұраныс құрып, орындау керек. Ол үшін PHP тілінде `mysql_query()` функциясы пайдаланылады. Оның аргументі: сұранысы бар жол. Функция сұраныс идентификаторын қайтарады.

Мысалы:

```
<html>
<head>
<title>Кестеге жазба қосу</title>
</head>
<body>
<?php
$db = "sample";
$link = mysql_pconnect ();
if ( !$link )
    ie ("MySQL-ге қосылу мүмкін емес");
mysql_select_db ( $db ) or die ("$db ашу мүмкін
емес");
$query = "INSERT INTO books
     VALUES ('966-7393-80-1', 'Аллен Вайк',
          PHP. Анықтама, '213', '4')";
mysql_query ( $query );
```

```
mysql_close ( $link );
?>
</body>
</html>
```

Мысалды орындаған сайын кестеге деректері бірдей жаңа жазба жазылып отырады.

Тапсырмалар:

1. Деректер қорына жаңа кітап қосатын HTML-форма құрыңдар.
2. Деректер қорынан белгілі бір кітапты іздейтін HTML-форма құрыңдар.

6 Ой бөлісейік

Сабақта не білдіңдер? Не үйрендіңдер? Өз ойларыңды достарыңмен бөлісіңдер. Қалай ойлайсыңдар, web-беттерді жасауда MySQL-ді қолдану және онымен үйлесімді жұмыс істеуі қаншалықты маңызды? Мысал келтіріңдер.

Глоссарий

Ақпараттық қауіпсіздік – ақпараттың бұрмалануы немесе жоғалуы салдарынан қолданушыға орны толмас зиян келтіретін кез келген әрекеттерден ақпаратты қорғау.

Ақпаратты қорғау – ақпараттық қауіпсіздікті қамтамасыз етуге бағытталған шаралар кешені.

Ақиқат кестесі – логикалық операцияның кестелік түрде ұсынылуы.

Алгоритм беріктігі – шифрдан ашуда өте көп көлемді есептеулерді талап ететін алгоритм.

Ауқымды айнымалылар – программа басында кез келген ішкі программаларды хабарлағанға дейін хабарланған деректер типтері, тұрақтылар мен айнымалылар.

Аутентификация – серверде енгізілген жеке деректерді тексеру және қабылдау.

Бастапқы кілт (`primary key, PK`) – кестедегі жазбаны бірегей түрде анықтайтын өрістердің ең аз жиынтығы.

Граф – екі жиынның жиынтығы: нүктелер жиыны мен сол нүктелердің кейбірін жұптап қосатын сызықтар жиыны.

Граф маршруты – көршілес төбелерді қосатын сол төбелер мен қабырғалардың соңғы кезектелген тізбегі.

Деректер – деректер қорында бірнеше типті деректердің бір типінде сақталатын ақпараттар жиыны.

Деректер қоры (ДҚ) – мекеменің ақпараттық қажеттіліктерін қанағаттандыруға арналған логикалық байланысқан деректердің жиынтығы (және олардың сипаттамасы).

Деректер қорын басқару жүйесі (ДҚБЖ) – қолданушыларға деректер қорын анықтауға, жасауға, қолдауға және оны бақылауға мүмкіндік беретін программалық жасақтама.

Домен – нүктемен бөлінген символдық аттар.

Есеп – экран беті, принтер секілді құрылғыларға немесе файл ретінде шығарылатын бір немесе бірнеше кесте деректерінің қосындысы.

Идентификация – серверге тіркелген қолданушының өзіне ғана тән жеке деректерді енгізуі.

Компьютерлік желі – бір-бірімен дерек алмаса алатын кем дегенде екі компьютердің байланыс құралдары көмегімен қарым-қатынас жасауына арналған ақпарат өңдеудің тармақталған жүйесі.

Концентратор – коммутатор мен маршрутизатормен салыстырғанда желідегі ең қарапайым және арзан құрылғы.

Көпіршікті сұрыптау – массивтер мен тізімдерді тізбектей салыстыру және егер алдыңғы элемент кейінгі тұрған элементтен үлкен болатын болса, көрші элементтерін ауыстыратын сұрыптау әдісі.

Криптоанализ – шифрдан ашу әдістері мен тәсілдері туралы ғылым.
Криптография – ақпаратты шифрлау әдістері туралы ғылым.
Қолжетімдік – уәкілетті қолданушылардың ақпаратына және тиісті активтеріне қолжетімдікті қамтамасыз ету.
Құпиялылық – ақпаратты тек уәкілетті пайдаланушыларға ұсыну.
Логика – адам ойлауының түрлері мен заңдары туралы, оның ішінде дәлелдеуге болатын пікірлердің заңдылықтары туралы ғылым.
Логикалық элемент – логикалық функциялардың біреуін орындайтын электронды құрылғы.
Маршрутизатор (router) – деректер пакетін тарату үшін екі не одан да көп желілерді байланыстыратын «ақылды» құрылғы.
Нысан – деректер мен функциялар жиынынан тұратын бірыңғай конструкция немесе JavaScript терминологиясында, қасиеттер мен тәсілдер жиыны.
Провайдер – ұйым мен жеке тұлғаларға Интернет қызметтерін ұсынатын компания.
Протокол – компьютерлік желіде ақпаратты ұсыну, түрлендіру және жіберу стандарты.
Пікір – жалған немесе ақиқат болуы мүмкін қандай да бір пайымдау.
Санау жүйесі – сандарды жазуға арналған ережелер мен арифметикалық операцияларды орындау мүмкіндігін беретін арнайы сандар жиыны.
Скрипт – құжатты жүктеу кезінде немесе кейінірек орындалуы мүмкін клиенттік сценарийлерді қамтитын HTML түзету тілі.
Стиль – құжаттың сыртқы келбетін жылдам өзгертуде қолданылатын пішімдеу ережелер жиынтығы.
Сұраныс – деректер қорынан белгілі бір ақпаратты алу талабы.
Тұтастық – ақпараттың сенімділігі мен толықтығын және оны өңдеу әдістерін қамтамасыз ету.
Түйін – желіге қосылу мүмкіндігі бар кез келген құрылғы.
Файл – компьютердің тұрақты жадысында жазылған символдар тізбегі.
Форма – кесте мен сұраныс деректерін енгізу, бейнелеу және түзету әрекеттерін орындау үшін қолданушы интерфейсін ұйымдастыруға мүмкіндік беретін нысан.
IP-мекенжай – бір түйіннен екінші түйінге ақпаратты жіберу, алу, табу үшін қажетті бірегей мекенжай.
JavaScript – нысанды-бағытталған, императивті және функционалды стильдерді қабылдай алатын мультипарадигмалық программалау тілі.
MySQL – дүниежүзінде ең көп қолданылатын, тегін және ашық, реляцияланған деректер қоры жүйесі (RDBMS).

Пайдаланылған әдебиеттер

1. Божко А.Н. Adobe FrameMaker. Сложная верстка: Учеб. пособие. – Аскери, М.: 2015. – 255 с.
2. Глушаков С.А., Кнабе Г.А., Компьютерная графика. Учебный курс – М.: Фолио, 2010.
3. Иванов В.П., Батраков А.С. Трехмерная компьютерная графика./ Под ред. Полищука. К.М. - М.: Радио и связь, 2015.
4. Компьютерная графика и анимация: А. Калбег – Санкт-Петербург, АСТ, Астрель, 2014 г. – 72 с.
5. Компьютерная графика. Учебник / М.Н. Петров, В.П. Молочков – СПб.: Питер, 2012.
6. Компьютерная графика: Практикум./ Л.А.Залогова – М.: ЛБЗ, 2009.
7. Корриган Дж. Компьютерная графика – М.: ЭНТРОП, 1995.
8. Культин Н.Б., С/С++ в задачах и примерах. БХВ – Петербург, 2012. – 288 с.
9. Маргулис Д. Препресс-ресурсы: Учеб. пособие. – М.: Попурри, 2010. – 256 с.
10. Матвеева Р.В., Трубникова Г.Г., Шифрина Д.А. Основы полиграфического производства: Учеб. пособие. – М.: Книга, 2014. – 312 с.
11. Мэйрин Д., Шэффер Д. Формат PDF в полиграфии: Учеб. Пособие. – М.: ПРИНТ-МЕДИА центр, 2014. – 234 с.
12. Немцова Т. И., Назарова Ю. В. Компьютерная графика и web-дизайн. Практикум: учебное пособие. ИД «ФОРУМ», ИНФРА-М, 2011.
13. Персональный компьютер: настройка и техническая поддержка: Учебное пособие. – Алматы, 2013. – 224 с.: ил.
14. Тутубалдин Д.К., Ушаков Д.А./ Компьютерная графика. Adobe Photoshop: Учеб. Пособие – изд. 2-е – Томск, 2015. – 131 с.
15. Шишкин Е.В., Боресков А.В. Компьютерная графика. Динамика, реалистичные изображения. – М.: Диалог-МИФИ, 2015.

Электрондық ресурстар

www.web-systems.com.ua
http://nashdesign.org.ua
http://bibliofond.ru/
http://3d.demiart.ru
http://www.f1cd.ru/soft/
http://ru.wikibooks.org/wiki/Blender_3D
http://digital-fantasy.ru
http://blender-school.ru

МАЗМҰНЫ

Назар аудар
Электронды қосымша жүктелген CD қолжетімсіз болған жағдайда, қосымшаны *arman-pv.kz* сайтынан тауып, өз компьютеріңе жүктеп алуыңа болады

Оқулық басылым

Гүлназ Ибрагимқызы Салғараева
Жұлдыз Болатханқызы Базаева
Айгүл Сейсенбайқызы Маханова

ИНФОРМАТИКА

Жалпы білім беретін мектептің жаратылыстану-математика бағытының 10-сыныбына арналған оқулық

Бас редакторы Қ.Қараева
Редакторы А.Умбеткалиева
Техникалық редакторы В.Бондарев
Көркемдеуші редакторы Е.Мельникова
Бильд редакторы Ш.Есенкулова
Суретші-безендіруші О.Подопригора
Мұқабаның дизайны В.Бондарев, О.Подопригора
Дизайнері Е.Молчанова
Беттеген Г.Илишева

Сатып алу үшін мына мекенжайларға хабарласыңыздар:
Нұр-Сұлтан қ., 4 м/а, 2 үй, 55 пәтер.
Тел.: 8 (7172) 92-50-50, 92-50-54. E-mail: astana@arman-pv.kz
Алматы қ., Ақсай-1А м/а, 28Б үй.
Тел./факс: 8 (727) 316-06-30, 316-06-31. E-mail: info@arman-pv.kz

«Арман-ПВ» кітап дүкені
Алматы қ., Алтынсарин к/сі, 87 үй. Тел.: 8 (727) 303-94-43.

Теруге 15.07.2018 берілді. Басуға 28.06.19 қол қойылды. Пішімі 70 x 100 $^{1}/_{16}$.
Қағазы офсеттік. Қаріп түрі «MM Mekteptik». Офсеттік басылыс.
Шартты баспа табағы 19,35. Таралымы 40000 дана.

Артикул 810-005-002к-19