ИНФОРМАТИКА

ЖАНА ЭСЕПТӨӨ ТЕХНИКАСЫНЫН НЕГИЗДЕРИ

Жалпы орто билим берүүчү мектептердин 9-классы үчүн окуу китеби

9

2-басылышы

Өзбекстан Республикасы Элге билим берүү министрлиги тарабынан бекитилген

Cho'lpon атындагы басма-полиграфиялык чыгармачылык үйү
Ташкент — 2015

УЎК: 372.8:004=512.154(075)
КБК 32.81(5Кир)я721
Б 73

Авторлор:
Б.Ж. Балтаев, А.Р. Азаматов, А.Д. Аскаров,
М.К. Садыков, Г.А. Азаматова

Жооптуу редактор:

Н. Тайлаков *– педагогика илимдеринин доктору, профессор.*

Рецензенттер:

М. Арипов *– Өз УУ «Информатика колдонмо программалоо» кафедрасынын профессору, физика-математика илимдеринин доктору,*
М. Ташов *– Наманган области Чуст районундагы 52-жалпы орто билим берүүчү мектептин жогорку категориялуу информатика-математика мугалими.*

Шарттуу белгилер:

Эсте тут

Суроо жана тапшырмалар

Практикалык иш же текшерүү сабагы

«Республикалык максаттуу китеп фонду каражаттары эсебинен басылды».

ISBN 978-9943-05-749-4

I БӨЛҮМ АЛГОРИТМДӨӨНҮН НЕГИЗДЕРИ

1-сабак. Маселелерди компьютерде чечүүнүн баскычтары

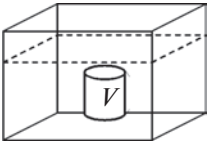
Адам практикалык иш жараянында өтө көп маселелерди чечүүгө туура келет. Маселелердин кээ бирлери жеңил, кээ бирлери татаал эсеп-кысап менен байланышкан болот. Кээ бир маселелерди чечүүдө кандайдыр бир амалдардын тобу болсо миңдеген жолу аткарылуусуна туура келиши мүмкүн. Ошондуктан чын дилден жана өтө тез иштей турган жардамчыбыз болгон компьютер бул ишибизге жардам бере алабы, эгерде жардам бере алса, анда маселелерди компьютерде чечүү кандай уюштурулат деген суроонун болушу табигый.

Бул суроого жооп берүүдөн мурда бир канча маселе жана алардын чыгарылышы көрүп чыгылат.

1-маселе. Көлөмү 20 $см^3$ болгон тело сууга матырылды. Ага аракет этип жаткан көтөрүүчү күчтүн маанисин тап.

Маселени чечмелейбиз: физика курсунан белгилүү болгондой, сууга матырылган тело өз көлөмүнө тең сууну сүрүп чыгарат жана ага сүрүп чыгарылган суунун салмагына тең күч аракет этет, бул күч Архимед күчү деп аталат.

Чиймеси:



Берилди:

$V = 20\ см^3 = 20 \cdot \frac{1}{100} \cdot \frac{1}{100} \cdot \frac{1}{100}\ м^3;$

$r = 1000\ \frac{кг}{м^3};\qquad g = 9{,}81\ \frac{Н}{кг}.$

Табуу керек: $F_A - ?$

Формулалар:

$F_A = r \cdot V \cdot g.$

Чыгаруу. $F_A = 1000\ \frac{кг}{м^3} \cdot \frac{20}{1000000}\ м^3 \cdot 9{,}81\ \frac{Н}{кг} =$

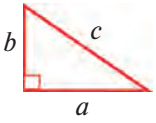
$= 0{,}1962\ \frac{кг}{м^3} \cdot м^3 \cdot \frac{Н}{кг} = 0{,}1962\ Н.$

Жооп: 0,1962 *Н*.

2-маселе. Мухтар чакмак баракка кызыл түстүү калемде негизи 16 чакмак, бийиктиги негизинин 3/4 бөлүгүнө тең тик бурчтуу үч бурчтук сызды. Ушул үч бурчтуктун периметрин тап.

Маселени чечмелейбиз: биринчиден, маселенин чечимин табуу үчүн үч бурчтуктун кандай түстүү калемде сызылганынын мааниси жоктугун, башкача айтканда (б.а.) бул биз үчүн **«керексиз»** маалымат, экинчиден үч бурчтуктун тик бурчтуу болушу негизги маалымат экендигин аныктайбыз. Эгерде эки чакмак 1 см ге тең экени эсепке алынса, анда геометрия курсунда маселенин чечилиши төмөнкү көрүнүштө болот:

Чиймеси:



Берилди:

$a = 16$ чакмак $= 8$ *см;*

$b = 8$ *см* $\cdot\ 3/4 = 6$ *см.*

Табуу керек: $P_{\text{үч б.}} - ?$

Формулалар:

Периметр: $P_{\text{үч б.}} = a+b+c.$

Пифагор теоремасы:

$c^2 = a^2+b^2.$

Чыгаруу. Пифагор теоремасынан:

$$c = \sqrt{a^2 + b^2} = \sqrt{(8см)^2 + (6см)^2} = \sqrt{100см^2} = 10\ см.$$

Анда: $P_{\text{үч б.}} = 8\ см + 6\ см + 10\ см = 24\ см.$

Жооп: 24 см.

3-маселе. Бекзат китептин төрт бетин жана дагы төрт катарын окуду. Китептин бетинде канча катар болсо, ар бир катарда ошончодон белги бар. Эгерде Бекзат окуган маалымат 6560 байт болсо, китептин бир бетинде канча катар бар экендигин аныкта.

Маселени чечмелөөгө өтөбүз.

Маселенин башталгыч баалары:

- Бекзат китептин 4 бетин жана 4 сабын окуган;
- Бекзат окуган маалымат 6560 байт;
- беттеги саптардын саны беттердеги белгилердин санына тең.

Маселенин максаты.

Китептин бетинде канча сап бар экендигин аныктоо.

Маселенин шарттарына ылайык тендеме түзүү.

Маселеде табуу талап кылынган саптардын санын x менен белгилейбиз. Анда шарт боюнча ар бир сапта x белги болот. Демек, китептин бир бетинде x^2 (x белгиден турган x сап) белги бар. Маселенин шартына ылайык Бекзат $4x^2 + 4x$ (4 бет жана 4 сап) белги окуган. Бул белгилердин саны 6560 байт (бир белги – бир байт) ка тең:

$$4x^2 + 4x = 6560.$$

Тендемени $x^2 + x - 1640 = 0$ көрүнүштөгү квадрат тендемеге келтиребиз, башкача айтканда маселенин шарттарына ылайык тендеме пайда кылдык.

Тендемени чыгаруу удаалаштыгы:

Сага белгилүү болгон квадраттык тендемени чыгаруу методу пайдаланылат:

1) дискриминант эсептелет: $D = 1^2 - 4 \cdot 1 \cdot (-1640) = 6561 = 81^2$.

2) $D > 0$ болгондуктан эки чечим табылат:

$$x_1 = \frac{-1-81}{2 \cdot 1} = -41, \quad x_2 = \frac{-1+81}{2 \cdot 1} = 40.$$

Натыйжаны иликтөө:

Тендеменин эки чыгарылышы бар экен. Бирок китептин беттеринин саны терс боло албайт, башкача айтканда тендеменин маселени канааттандыра турган чечими $x = 40$ экен. **Жооп:** 40 сап.

Жогорудагы маселелердин чыгарылышын чечмелеп, алар төмөнкү баскычтардан тургандыгын көрүү мүмкүн:

1. Ар бир маселеде оболу **маселенин коюлушу,** башкача айтканда маселеде берилген башталгыч чондуктар жана маселенин максаты (табылышы керек болгон натыйжалык чондуктар) аныкталат.

2. Маселени чечүү үчүн зарыл болгон формулалар, башкача айтканда **математикалык туюнтмалар** пайда кылынат.

3. Маселенин чыгарылышындагы амалдарды (формулаларды, туюнтмаларды) **аткаруу удаалаштыгы** аныкталат (2–3-маселелерде бул даана байкалат).

4. **Натыйжаны чыгаруу жана иликтөө.**

Жогорудагы сыяктуу башка маселелерди да компьютердин жардамында чечүү мүмкүн жана ал жогорудагы 4 баскычка кошумча **амалдарды компьютер түшүнө турган тилге которуу** жана **компьютердин эстутумуна** киритүү сыяктуу баскычтарды өз ичине алат:

Маселелерди компьютерде чыгаруу баскычтарынан кээ бирлери белгилүү бир билим жана тажрыйбаны талап кылгандыктан, атайын темалар аркылуу жарытып барылат.

Биринчи баскыч: Маселенин коюлушу	Маселеге ылайык башталгыч чондуктар жана натыйжалык чондуктар аныкталат.
Экинчи баскыч: Маселенин моделин куруу	Маселе көрүлүп жаткан тармактын илимий ийгиликтеринен келип чыгып, формулалар менен туюнтулат.
Үчүнчү баскыч: Алгоритм түзүү	Маселенин моделинен пайдаланып, чечүүнүн көрсөтмөлөр удаалаштыгы түзүлөт.

Төртүнчү баскыч: Программа түзүү	Алгоритмдеги көрсөтмөлөр удаалаштыгын компьютер түшүнө ала турган тилге өткөрүү.
Бешинчи баскыч: Программаны компьютердин эстутумуна киритүү	Түзүлгөн программа компьютердин эстутумуна киритилет.
Алтынчы баскыч: Натыйжа алуу жана аны иликтөө	Программа иштетилет жана натыйжасы иликтенгенден соң, ката жана кемчилдиктер жоюлат.

Суроо жана тапшырмалар

1. *Компьютерде маселе чыгаруунун баскычтары канчоо?*
2. *Эмне үчүн алынган натыйжа иликтенет?*
3. *Калькулятордо эсеп-кысап иштери аткарылганда кандай каталыктар келип чыгат?*
4. *$23 + 46 \cdot 3 - 24 : 3$ арифметикалык туюнтманы эсептөө үчүн амалдарды аткаруунун удаалаштыгын аныкта.*
5. *Маселенин шартына ылайык теңдеме түзүү үчүн мисалдар келтир.*

Көнүгүүлөр

Төмөнкү маселелердин шартын иликте жана баскычтарга бөлүп чыгар.

1. Кыймылсыз суудагы ылдамдыгы 15 км/саат болгон кайыктын дарыянын агымы бойлоп 2 сааттагы басып өткөн аралыгы агымга каршы 3 саатта басып өткөн аралыкка тең болсо, дарыя агымынын ылдамдыгын тап (көмөк: ылдамдык = жол/убакыт).

2. Туура төрт бурчтуктун жактары, тиешелүү түрдө, 4 см жана 3 см болсо, анын диагоналынын узундугун тап (көмөк: туура төрт бурчтуктун диагоналы төрт бурчтукту эки туура үч бурчтукка ажыратат, демек, диагонал гипотенуза болот).

2-сабак. Модель жана анын түрлөрү

Бизди кызыктырып жаткан жана үйрөнүлүп жаткан нерсе же жараян **объект** деп аталат. Мисалы, күн системасындагы планеталар, спорт топтору, мектебиндеги компьютерлер объекттерге мисал болот. Бир түрдөгү үйрөнүлүп жаткан объекттер өзүнүн касиеттери – **мүнөздөмөсүнө** ээ болот. Ар бири өзүнчө алынган объект болсо башкасынан ушул мүнөздөмөгө тиешелүү

мүнөздөмө баасы менен айырмаланат. Мисалы, үйрөнүлүп жаткан компьютерлер наамдуу объекттердин мүнөздөмөсү: иштеп чыгарган фирманын аты, негизги платанын маркасы (motherboard), процессордун аты, процессордун ылдамдыгы (CPU), винчестер сыйымдуулугу, ылдам эстутумдун (RAM) сыйымдуулугу, видеоэстутум сыйымдуулугу болсо, анык компьютердин мүнөздөмө баасы: иштеп чыгарган фирманын аты FUJITSU SIEMENS, негизги платанын маркасы D1170, процессордун аты Pentium IV, процессор ылдамдыгы 3,06 Ггц, винчестер сыйымдуулугу 160 гигабайт, ылдам эстутум сыйымдуулугу 1 Гигабайт, видеоэстутум сыйымдуулугу 512 Мбайт.

Эгерде үйрөнүлүп жаткан объекттер планеталар болсо:

Планетанын **мүнөздөмөсү**	формасы	салмагы	радиусу	айлануу ылдамдыгы
Жер үчүн **мүнөздөмө баасы**	Шар сымал	$5976 \cdot 10^{21}$ кг	6378 км	30 км/сек

«Топ» аттуу объект үчүн:

Топтордун **мүнөздөмөсү**	формасы	салмагы	радиусу	материалы
Топ үчүн **мүнөздөмө баасы**	Сфера сымал	2,2 кг	15 см	резина

Көп учурда белгилүү бир тармакка тиешелүү изилдөө алып барганда чыныгы объект эмес, анын кайсы бир маанидеги көчүрмөсү үйрөнүлөт. Буга, бир жааттан, белгилүү бир себептерге көрө (чагылгандын туруктуу эместиги, күндүн алыстыгы, объект менен иштөө чоң каражатты талап кылуусу же адамдын жашоосуна кооп салышы жана башка) чыныгы объектти түздөн-түз үйрөнүүнүн аргасы болбосо, экинчи жактан изилдөө үчүн объекттин кандайдыр маанидеги көчүрмөсүн үйрөнүүнүн өзү да жетиштүү болот. Албетте, бул учурда объекттин көчүрмөсү изилдөөчү тармактын талабына толук жооп бериши керек.

Модель – чыныгы **объект**тин изилдөө алып барылып жаткан тармактын белгилүү бир талаптарына жооп бере турган **көчүрмөсү** болуп саналат.

Модель сөзү (латынче **modulus** – өлчөө, норма) сага самолёт, машина же кеме куруучулук ийримдери аркылуу тааныш болсо керек. Турмушта объект моделине көп мисал келтирүү мүмкүн. Мисалы, жердин модели глобус же карта; самолёт модели болуп кичирейтилген нускасы, автомашинанын модели оюнчук-

тар; чагылгандын модели жогору чыңалуудагы электр булагындагы кыска туташуу же ширетүүдө электроддун күйүшү; адамдын модели болуп анын клеткасы же куурчак, же фотосүрөтү; адам мээсинин эсептөө боюнча модели болуп калькулятор же компьютер кызмат кылат.

Чыныгы объект жана анын модели жүргүзүлүп жаткан тажрыйбаларда бирдей натыйжа берсе гана изилдөө жүргүзүлүп жаткан тармактын талаптарына жооп берет. Мисалы, самолёт жана анын модели бир түрдүү аэродинамикалык мыйзамдарга баш иет. Модель үчүн алынган натыйжалар чыныгы самолёт үчүн да орундуу болот. Пландаштырылган чыныгы самолёт курулгандан соң, аны лабораториядагы атайын курулмалар – самолётко аба агымын жиберүүчү стенддерде сынап көрүлөт. Бул абалда лабораториядагы стенддер атмосферанын модели болуп кызмат кылат. Турмушта, модели катары математикалык туюнтма жана формулалар карала турган жараяндар да болот. Мында тандалган модель чыныгы объекттин касиеттерин өзүндө камтыган болушу зарыл, б. а. үйрөнүлүп жаткан объект менен анын моделинин өзгөчөлүктөрү бирдей катыш жана формулалар аркылуу туюнтулушу керек.

Үйрөнүлүп жаткан объект мүнөздөмөлөрүнүн математикалык туюнтма, белги жана байланыштар аркылуу берилиши ***математикалык модель*** деп аталат.

Үйрөнүлүп жаткан объекттин математикалык туюнтма жана белгилер аркылуу берилүү жараяны математикалык моделдештирүү деп аталат.

Өткөн сабакта көрүп өтүлгөн китептеги саптардын санын табуу маселеси квадраттык теңдеме көрүнүшүндө туюнтулду. Демек, маселени квадраттык теңдеме түрүндө туюнтуу математикалык моделдештирүү, ал эми тиешелүү теңдеме маселенин математикалык модели болот. Ошондой эле, Архимед күчү, Пифагор теоремасы жана периметрдин формуласы да математикалык модель болот.

Математикалык моделдештирүү байыртадан астрономия, химия жана физикада колдонулган. Мисалы, Нептундун ачылышын алалы. 1846-жылда француз астроному У.Леверье Уран планетасынын өзгөчө аракеттенүүсүнө Күн системасынын ошол мезгилге чейин белгисиз болгон планетасы себепчи экенин математикалык жактан далилдеп берген. Ошол жылы Леверьенин

көрсөтмөсүнө негизделип немец астроному Галлей Нептун планетасын телескоп аркылуу байкай алган.

Химиялык реакциялардын математикалык моделине мисалдар:

1) хлор менен натрийдин биригүү реакциясы:

$2Na + Cl_2 = 2NaCl$;

2) табигый газдан күкүрттү бөлүп алуу реакциясы:

$2H_2S + O_2 = 2H_2O + 2S$.

Физикалык кубулуштардын математикалык моделине булар мисал болот:

1) Ньютондун экинчи мыйзамы, б. а. телого аракет этип жаткан күчтүн формуласы: $F = ma$, мында m – телонун массасы, a – ылдамдануу;

2) Ньютондун бүткүл дүйнөлүк тартылуу мыйзамы: $F = G\frac{m_1 m_2}{R^2}$, мында m_1, m_2 – өз ара аракеттешип жаткан телолордун массалары, R – алардын арасындагы аралык, G – гравитациялык туруктуулук.

Бүгүнкү күндө да моделдештирүү химия, биология, медицина, экономика өндүү илим тармактарында кең колдонулуп, өтө кызыктуу натыйжалар алынууда.

Жалпысынан алганда, моделдер объекттердин туюнтуу каражаттарын тандоого карап төмөнкү схемада сүрөттөлгөндөй, үч негизги түргө бөлүнөт:

1. Абстракттуу моделдер өз кезегинде эки топко бөлүнөт: **математикалык** жана **экономикалык математикалык** моделдер.

Математикалык моделдер объекттин түзүлүшү жана өз ара байланыш мыйзам ченемдүүлүктөрүнүн математикалык туюнтма, формула жана математикалык-логикалык мүнөздөмөсүнөн турат. Мындай моделдерге өткөн сабактарда мисалдар көрүлдү.

Экономикалык математикалык моделдер XVIII кылымдан колдонула баштаган. Ф. Кененин «Экономикалык жадыбалдарында» биринчи жолу бүткүл коомдук өндүрүш жараянынын калыптануусун көрсөтүп берүүгө аракет жасалган. Бүгүнкү күндө экономикалык моделдердин жардамында экономикалык өнүгүүнүн эң жалпы мыйзам ченемдүүлүктөрү текшерилет. Түрдүү экономикалык көрсөткүчтөр, мисалы, улуттук киреше, керектөө, жумуш менен камсыздалуу, фонддор, инвестиция көрсөткүчтөрүнүн өзгөрүүсүн жана катышын иликтөө, аны алдын ала айтып берүү үчүн татаал экономикалык моделдер колдонулат. Эгемендүү Өзбекстандын **5 принцибинин негизинде базар экономикасына өтүү модели** да экономикалык математикалык моделдин негизин түзөт (бул принциптерди эске ал!).

2. Физикалык моделдерде объекттин табияты жана түзүлүшү дээрлик чыныгы көчүрмөсү сыяктуу болот, бирок андан ченем (өлчөмү, ылдамдыгы жана башка) жагынан айырмаланат. Мисал катары самолёт, кеме, автомобиль, поезд жана башкалардын моделдерин алуу мүмкүн.

3. Биологиялык модель болсо түрдүү тирүү объекттер жана алардын бөлүктөрүнө (клетка, организм жана башка) тиешелүү биологиялык түзүлүш, функция жана жараяндарды моделдештирүүдө колдонулат. Биологиялык модель адам жана жаныбарларда кездеше турган белгилүү бир абал же ооруларды лаборатория жараянында сынап көрүү мүмкүнчүлүгүн берет. Мисалы, зыяндуу вирусту өлтүрө турган дарыны текшерүү үчүн адамдын өзүндө эмес, анын аз өлчөмдөгү канынан алып, ошол канында сынап көрүлүшү жетиштүү болот.

Төмөндө физикалык жана биологиялык жараяндардын математикалык моделдерине мисал көрүп чыгылат.

1-маселе. Басымы p, көлөмү V жана температурасы T болгон идеалдык газ абалын туюнтуучу математикалык модель түз. Бул маселенин чечимин Клайперон формуласы берет, б. а. идеалдык газдын басымы, көлөмү жана температурасы өз ара төмөнкүчө байланышкан:

$$\frac{pV}{T} = \text{const}.$$

Бул формула идеалдык газ температурасынын өзгөрүүсү басым же көлөмдүн өзгөрүүсүнө себеп болушун даана туюнтат.

2-маселе. Гүлдүн өсүү жараянынын моделин түз.

Өсүмдүктүн жашашы жана өсүшү үчүн аба, жарык, суу жана азык керек болоору ботаника курсунан маалым. Алардын өлчөмү түрдүү өсүмдүктөр

үчүн ар башка болот. Мисалы, кээ бир гүлдөр караңгы жана кургак шартта жакшы өссө, башкалары жарыкты жана нымдуулукту көп талап кылат. Ошондуктан маселенин модели төмөнкү тендемелер системасы аркылуу туюнтулат:

$$\begin{cases} T = T_0 \cdot (1 + \alpha t); \\ I = I_0 \cdot (1 + \beta t); \\ H = H_0 \cdot (1 + \gamma t), \end{cases}$$

мында t — убакыт; T — абанын температурасы; I — жарыктын өлчөмү; H — гүлдүн денесиндеги нымдуулуктун өлчөмү; α, β, γ — температура, жарык, нымдуулукка мүнөздүү туруктуу чоңдуктар.

Көрүп өтүлгөн маселелердин моделдерине көңүл буруп, кайсы тармакта болбосун, математикалык моделдештирүү үчүн бир гана математикадан эмес, ошондой эле ушул тармактардан да жетиштүү билимге ээ болуу зарыл экендигин айтуу мүмкүн.

Суроо жана тапшырмалар

1. *Объект деп эмнеге айтылат?*
2. *Объекттин мүнөздөмөсү жана мүнөздөмө баасы жөнүндө мисалдар аркылуу айтып бер.*
3. *Модель деп эмнеге айтылат?*
4. *Объект жана ага мүнөздүү моделдерге мисалдар келтир.*
5. *Математикалык модель деп эмнеге айтылат? Математикалык моделдер кандай тармактарда колдонулат?*
6. *Математикалык моделдин башка моделдерден айырмасын түшүндүр.*
7. *Нептун планетасы кандай ачылган?*
8. *Математикалык моделдердин химия жана физикада колдонулушуна мисалдар келтир.*
9. *Моделдер канча түргө бөлүнөт?*
10. *Кандай абстракттуу моделдер бар?*
11. *Экономикалык математикалык моделдер жөнүндө айтып бер.*
12. *Кандай физикалык моделдерди билесиң?*
13. *Биологиялык моделдердин мааниси жөнүндө айтып бер.*

Көнүгүүлөр

Төмөнкү объекттердин мүнөздөмөсүн жана мүнөздөмө баасын жаз.

1. Объект: облустар (көмөк: аталышы, аянты, калкынын саны, негизги экономикалык продукциясы, ...).

2. Объект: классташтар (көмөк: жынысы, бою, чачынын түсү, салмагы, көзүнүн түсү, ...).

3. Объект: китептер (көмөк: аталышы, бетинин саны, түстүүлүгү, салмагы, баасы, ...).

3-сабак. Маселелерди компьютерде чечүүнүн баскычтары жана моделдин түрлөрү темаларын кайталоо

1. Төмөнкү маселелердин шартын чечмеле жана баскычтарга бөлүп чыгар.

А. Катеттери ***a*** жана ***b*** болгон туура үч бурчтуктун гипотенузасын эсепте.

Б. Катеттери ***a*** жана ***b*** болгон туура үч бурчтуктун аянтын эсепте.

В. Жагы ***a*** болгон тең жактуу үч бурчтуктун бийиктигин тап.

2. Төмөнкү объекттердин мүнөздөмөсү жана мүнөздөмө баасын жаз.

а) объект: өзүң жашай турган облус (шаар)тагы коллеждер (көмөк: аты, курулган жылы, багыттары, кабыл алына турган окуучулардын саны, ...).

б) объект: Асака автомобиль заводу өндүрө турган автомобилдер (маркасы, чыга баштаган жылы, саны, түстөрү, ...).

3. Төмөнкү маселелерге мүнөздүү модель түз жана чыгар.

А. Банкка жылына A пайыздуу киреше алуу үчүн коюлган B сум акчанын M жылдан кийинки абалын туюнтуучу модель түз.

Көмөк. 1-жылдын аягында алына турган киреше $\frac{B}{100} \cdot A$ сум болот. Ошондуктан жыл аягында банктагы акча $\frac{B}{100} \cdot A + B = B \cdot (\frac{A}{100} + 1)$ сум болот. Экинчи жылдын аягында алына турган киреше $B \cdot (\frac{A}{100} + 1) \cdot \frac{A}{100}$ сум болот. Экинчи жылдын аягында банктагы акча $B \cdot (\frac{A}{100} + 1) \cdot \frac{A}{100} + B \cdot (\frac{A}{100} + 1) = B \cdot (\frac{A}{100} + 1) \cdot (\frac{A}{100} + 1) = B \cdot (\frac{A}{100} + 1)^2$ сум болот.

Үчүнчү жана төртүнчү жылдын аягында банктагы акча канча болушун эсепте жана алынган формулаларды жалпылаштыр.

Б. Самолёт арасындагы аралык 2100 км болгон A шаардан B шаарга чейин 3 саат, арасындагы аралык 4800 км болгон B шаардан M шаарга чейин 6 саат учту. Самолёт кандай орточо ылдамдыкта учкан? (көмөк: орточо ылдамдык=(1-жол + 2-жол)/(1-убакыт + 2-убакыт)).

4-сабак. Алгоритм түшүнүгү

Инсан турмушта чоң жана кичине милдеттерди же маселелерди чечүүнү өз алдына максат кылып коёт. Адатта, ал өз максатына жетүү үчүн аткарышы керек болгон амал же иштерди

турмуштук тажрыйбасы же өздөштүргөн билимдерине негизделип белгилүү бир тартипке келтирет. Буга ар түрдүү мисалдарды келтирүү мүмкүн.

1-мисал. Чай демдөө максат кылып коюлган болсун. Анда чай демдеп жаткан адам бардыгыбызга адат болуп калган төмөндөгү иштерди аткаруусу керек болот:

1) чайнектин капкагын ач;
2) чайнекти кайнаган суу менен чайка;
3) чайнекке бир чай кашык өлчөмдө кургак чай сал;
4) чайнек толгончо кайнаган суу куй;
5) чайнектин капкагын жап;
6) чайнекти сүлгү менен ороп беш мүнөткө дем бер.

2-мисал. Туурасы N метр жана узуну M метр болгон жайды толтурууга 12×25 сантиметрлүү (туурасы 12, узуну 25 см) кыш (кирпич) тан канча даана сатып алынышын табуу керек болсун. Эсептеп жаткан адам геометриядан алган билимине негизделип төмөнкү удаалаш амалдарды аткарат:

1) жайдын аянты $S_{жай}$ сантиметр чен бирдигинде табылсын;
2) бир даана кыштын аянты $S_{кыш}$ сантиметр чен бирдигинде табылсын;
3) кыштын саны $S_{сан}$ жайдын аянтынын кыштын аянтына катышы деп алынсын.

Бул амалдар удаалаштыгын төмөнкү математикалык формула жардамында туюнтуу мүмкүн:

$$S_{сан} = \frac{S_{жай}}{S_{кыш}} = \frac{N \cdot 100 \cdot M \cdot 100}{12 \cdot 25}.$$

3-мисал. Амал аткарылсын: 19632107 + 19702202. Бул амалды кандай аткарган болот элең? Ооба, туура, бул сандарды мамыча көрүнүшүндө дээрлик төмөнкүчө кошосун:

1) сан разряддары туура келе турган тартипте үстү-үстүнө жазып алынсын;

2) сандардын бирдик разрядынын сандарын кошуп, сумманын бирдик саны бирдиктердин астына жазылып, ондук саны көмүскөдө сакталсын;

3) сандардын ондук разрядынын сандарына көмүскөдөгү сан кошулуп, сумманын бирдик саны ондуктардын астына жазылып, ондук саны көмүскөдө сакталсын; жана 3-пункттагы эреже жүздүктөр, миңдиктер ж. б. үчүн кайталанат. Бул амалдар төмөнкү көрүнүштө сага абдан тааныш:

$$\begin{array}{r} 19632107 \\ +\underline{19702202} \\ 39334309 \end{array}$$

Жогорку мисалдарда келтирилген амалдардын удаалаштыгы, б. а., көрсөтмө же командалардын удаалаштыгы кандайдыр бир адам тарабынан аткарылгандан соң, көздөлгөн максатка жети-

шилет. Турмушта ар күнү жана ар саатта кездеше турган түрдүү эрежелердин ичинде кандайдыр бир зарыл натыйжага жетишүүгө алып келүүчү амалдарды удаалаш аткаруу талап кыла турган эрежелер информатиканын негизги түшүнүктөрүнөн бири **алгоритм** сөзү менен туюнтулат.

Алгоритм сөзү IX кылымда жашап өзүнүн илимий иштери менен дүйнөгө таанылган мекендешибиз улуу астроном, математик жана географ Абу Абдулла Мухаммад ибн Муса **ал-Харезмийдин** (783–850) атынан келип чыккан. Ал-Харезмийдин арифметикага арнаган чыгармасы XII кылымда Испанияда латын тилине которулган. Бул чыгарманын XIV кылымда көчүрүлгөн жалгыз кол жазма үлгүсү Кембриж университетинин китепканасында сакталууда. Чыгарма латын тилинде **«Dixit Algorithmi»**, б. а. «ал-Харезмий мындай деди» сөзү менен башталат.

Алгоритмдеги ар бир көрсөтмө же команда кандайдыр бир **амал**ды аткарууну көздө тутат. Алгоритмдеги амалдарды аткара турган объект **аткаруучу** түшүнүгү менен байланышат. Ар кандай алгоритм – бул амалдарды белгилөөчү эреже болуп, алардын чынжыры натыйжасында берилген маанилерден изделген натыйжага келинет. Мындай амалдардын чынжыры алгоритмдик жараян, ар бир амал **алгоритмдин кадамы** деп аталат.

Алгоритм дегенде кайсы бир максатка жетишүүгө каратылган, **аткаруучу** аткарышы көздөлгөн командалардын удаалаштыгы түшүнүлөт.

Демек, жогорку мисалдардагы команда (же көрсөтмө)лар удаалаштыгы **алгоритм** жана бул алгоритмдерди аткаруучу адам – **аткаруучу** болот. Биринчи мисалдагы көрсөтмөлөр «Чай демдөө алгоритми» деп аталат. Мындан ушундай жыйынтыкка келебиз: адам турмушта көздөгөн максатына жетүү үчүн аткаруучу катары көптөгөн алгоритмдерди аткарат. Көптөгөн алгоритмдер адам үчүн адат болуп калган. Мисалы, тамак даярдоо, тамактануу, тартиптүү кийинүү, бөлмөдөн чыгуу, жазуу, бир жайдан экинчи жайга баруу ж. б.

Көрсөтмөлөр тартибинин бузулушу кандай натыйжага алып келиши мүмкүндүгүн элестетүү кыйын эмес. Мисалы, «Чай демдөө алгоритми»нде биринчи жана үчүнчү көрсөтмөлөрдүн ордун алмаштырып аткаруу жетиштүү. Адатта, алгоритмдеги көрсөтмөлөр аткаруучуга түшүнүктүү болушу үчүн жөнөкөй амал-

дардан турган болушу керек. Экинчи мисалдагы алгоритмдин биринчи көрсөтмөсүн төмөнкү үч көрсөтмөгө бөлүү мүмкүн:

а) жайдын туурасы N метр сантиметр өлчөм бирдигине өткөрүлсүн;

б) жайдын туурасы M метр сантиметр өлчөм бирдигине өткөрүлсүн;

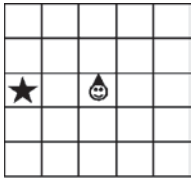
в) жайдын аянты $S_{жай}$ табылсын.

Алгоритмдин аткаруучусу жалаң гана адамбы, деп суроо бериши табигый. Бул суроонун жообу төмөнкүдөй:

Алгоритмдин аткаруучусу – алгоритмде көрсөтүлгөн команда же көрсөтмөлөрдү аткара ала турган абстракттуу же реалдуу (техникалык же биологиялык) система болуп саналат.

Аткаруучу аткара алышы мүмкүн болгон көрсөтмө же командалардын жыйнагы **аткаруучунун көрсөтмөлөр системасы** (кыскача, **АКС**) деп аталат. Мисалы, «16 санынан квадраттык тамыр чыгарылсын» көрсөтмөсү 2-класс окуучусунун көрсөтмөлөр системасына тиешелүү болбойт, бирок 8-класс окуучусунун көрсөтмөлөр системасына тиешелүү болот. Информатикада алгоритмдин негизги аткаруучусу болуп **компьютер** кызмат кыла тургандыгын баса белгилөө зарыл.

Аткаруучунун көрсөтмөлөр системасын төмөнкү маселе аркылуу түшүндүрөбүз.



4-мисал. Боорсок үчүн «алдындагы» чакмак калпагы көрсөтүп жаткан чакмак болуп саналат, мисалы ал оңго бурулганда көрүнүштө болот. Боорсок 1 кадам алдыдагы чакмакка жүрө алат же турган чакмагында оңго же солго бурула алат. Боорсок бир чакмактан бир канча жолу өтүшү да мүмкүн. Боорсок өзү турган чакмактан ★ менен белгиленген чакмакка кандайдыр бир жол менен бара ала турган болсо, зарыл болгон көрсөтмөлөрдүн удаалаштыгын жаз. Маселе шартынан аткаруучу Боорсоктун көрсөткүчтөрүнүн системасын (БКС) жаза алабыз, башкача айтканда **БКС={алдыга; оңго; солго}**. Эми маселенин чечими катары төмөнкү алгоритмдерден бирин алуу мүмкүн:

Кадамдардын саны	1-алгоритм	2-алгоритм	3-алгоритм
1	1) солго;	1) оңго;	1) алдыга;
2	2) алдыга;	2) оңго;	2) солго;
3	3) алдыга.	3) оңго;	3) алдыга;
4		4) алдыга;	4) алдыга;
5		5) алдыга.	5) солго;
6			6) алдыга.

Демек, **маселенин чечими**не алып баруучу алгоритм **жалгыз болбостугу** да **мүмкүн** экен.

Жогоруда көрүп чыгылган мисалдардан же айтып өтүлгөн маселелерден мындай жыйынтыкка келебиз: аткаруучу алгоритмди аткаруу жараянында көздөлгөн максатты билбестиги да мүмкүн.

Мисалы, төмөнкү алгоритмди аткаруудан кандай максат көздөлгөнү алдын ала билинбейт:

1) *N* жана *M* натуралдык сандар алынсын;
2) *S* саны нөлгө тең деп алынсын;
3) *N* жана *M* сандардын чоңу өзү менен кичине сандын айырмасына тең деп алынсын жана *S* ке бир кошулсун;
4) эгерде *N* жана *M* сандарынын экөөсү тең нөлдөн чоң болсо 3-пунктка өтүлсүн, антпесе кийинки катарга өтүлсүн;
5) жооп катары *S* жазылсын.

Бул алгоритм төмөнкү маселенин чечимин табуу мүмкүнчүлүгүн берет:

5-мисал. Жактары *N* жана *M* натуралдык сандарга барабар болгон туура төрт бурчтук берилген. Эгерде ар кадамда аянты эң чоң аянттуу квадрат кесип алына берсе, канча квадрат кесип алынат?

Бул сабак аркылуу маселелерди компьютерде чыгаруунун негизги баскычтарынан бири менен байланыштуу болгон информатиканын **алгоритм, алгоритмдин аткаруучусу, аткаруучунун көрсөтмөлөр системасы** сыяктуу негизги түшүнүктөрү менен таанышып, ушундай жыйынтыкка келинет: **алгоритм аркылуу аткаруучу башкарылат.**

Суроо жана тапшырмалар

1. Алгоритм дегенде эмнени түшүнөсүң?
2. Алгоритм сөзүнүн келип чыгуу тарыхын айтып бер.
3. Алгоритмге мектеп турмушунан мисалдар келтир.
4. Окуу китебинен берилген теманы табуу алгоритмин түз.
5. «Тамак» бышыруу алгоритмин түз.
6. Компьютерди ишке түшүрүү алгоритмин түз.
7. Алгоритм аткаруучусу жөнүндө эмнелерди билесиң?
8. Кандай көрсөтмөлөрдү аткаруучу аткара албайт?
9. Аткаруучунун көрсөтмөлөр системасына мисалдар келтир.
10. Классташың аткара албай турган көрсөтмөлөрдү жаз.
11. Төмөнкү көрсөтмөлөр алгоритм боло алабы жана аларды аткаруудан кандай максат көздөлгөн?
1) дарыядан бир чака суу алынсын;
2) чакадагы суу дарыяга куюлсун;
3) 1-пунктка кайтылсын.

5-сабак. Алгоритмдин негизги касиеттери

Мурдагы сабакта алгоритм жана алгоритмдин аткаруучусу жөнүндө сөз болгон эле. Эми алгоритмдин негизги касиеттери менен кеңири тааныштырылат:

1. Түшүнүктүүлүк. Алгоритмдин аткаруучусуна түшүнүктүү болушу үчүн аткаруучунун мүмкүнчүлүктөрүн билүү керек. Эгерде аткаруучу адам болсо, анда алгоритм адамдын мүмкүнчүлүктөрүнөн келип чыгып түзүлүшү керек. Мында көздөлгөн максат жана алгоритмден келип чыгып адам түшүнө турган тил, адамдын билими, турмуштук жана кесиптик тажрыйбасы, жашы, жадагалса, күчтүк мүмкүнчүлүктөрү эсепке алынышы зарыл. Эгерде аткаруучу техникалык каражат (мисалы, компьютер, электрондук саат, аспаптар) болсо, анда алгоритм ушул техникалык каражаттын мүмкүнчүлүктөрүнөн келип чыгып түзүлүшү зарыл.

Демек, берилип жаткан ар кандай көрсөтмө аткаруучунун көрсөтмөлөр системасынан алынышы, башкача айтканда аткаруучу аны кандай аткарууну билиши керек экен.

2. Тактык. Алгоритмдеги бардык амалдар, көрсөтмөлөр же командалар бир маанилүү жана так болушу керек. Мисалы, «азыраак туз салынсын» (бир аш кашыкпы же бир чай кашыкпы же болбосо бир пиялабы?), «керегинче суу куюлсун» (керек дегенде канча суу көздө тутулду: 1 литрби, 100 литрби, 1 тоннабы?), «дил баян жазып келинсин» (кайсы тема боюнча?) сыяктуу көрсөтмөлөр ар түрдүү (көбүнчө керексиз) натыйжаларга алып келет.

Мындан ушундайча жыйынтыкка келебиз: тактык өзгөчөлүгүнө негизинен алгоритмдин аткаруучусу көрсөтмөлөрдүн удаалаштыгын механикалык түрдө катасыз аткарат жана кошумча түшүндүрүүлөр талап кылбайт.

3. Дискреттүүлүк (үзгүлтүксүз, өзүнчө). Алгоритмде маселени чыгаруу жараяны атайын алынган жөнөкөй көрсөтмөлөрдүн удаалаштыгын иреттүүлүк менен аткаруудан турушу керек. Бул касиет мурдагы сабактагы мисалдарда ачык-айкын көрүнүп турат.

4. Натыйжалуулук (чектүүлүк). Алгоритмдин мүнөздөмөсүндө «кандайдыр бир максатка жетишүүгө каратылган» сөзү колдонулган. Бул максатты жогоруда келтирилген мисалдарда көрүү мүмкүн: чай демдөө, кыштардын санын эсептөө, сумманы эсептөө. Булар алгоритмдин **натыйжалуулук** касиети менен байланыштуу. Бул касиеттин мазмуну, ар кандай алгоритмдин аткарылышы чектелүү кадамдан соң акыр-аягы белгилүү бир чечимге алып келиши керек экендигинен турат. Дагы айта кетчү нерсе, алгоритм мурдатан көздөлгөн максатка жетишүүгө алып келбестиги да мүмкүн. Буга кээде алгоритмдин туура эмес түзүлгөнү же башка каталык себеп болушу да мүмкүн. Экинчи жактан, коюлган маселе оң чечимге ээ болбостугу да мүмкүн. Бирок терс натыйжа да **натыйжа** деп кабыл алынат.

1-мисал. $x^2 + x + 1 = 0$ квадраттык тендемеси чыгарылсын.

Төмөндө келтирилген «$ax^2 + bx + c = 0$ $(a \neq 0)$ көрүнүшүндөгү квадраттык тендемени чыгаруу» алгоритмин колдоп, тендеменин чечимге ээ эместиги аныкталат. Бул да натыйжа болуп эсептелет.

1) a, b, c маанилер аныкталсын;

2) дискриминант: $D = b^2 - 4ac$ эсептелсин;

3) эгерде $D < 0$ болсо, тендеме чечимге ээ эмес деп алынсын жана 6-пунктка өтүлсүн;

4) эгерде $D = 0$ болсо, жалгыз чечим $-\frac{b}{2a}$ га барабар деп алынсын жана 6-пунктка өтүлсүн;

5) биринчи чечим $\frac{-b-\sqrt{D}}{2a}$ га, экинчи чечим $\frac{-b+\sqrt{D}}{2a}$ га барабар деп алынсын;

6) аяктасын.

Көңүл бурган болсоң, дискриминанттын нөлдөн кичинелиги жана нөлгө барабардыгы текшерилди, бирок нөлдөн чоңдугу текшерилбеди. Себебин ойлоп көр!

Демек, алгоритм дайыма чектүү кадамдан турушу жана кандайдыр бир натыйжа бериши керек экен.

5. Жалпылуулук. Кандайдыр бир маселени чыгаруу алгоритми жалпы абалдар үчүн түзүлөт, б. а. бир гана башталгыч маалыматтар менен айырмалануучу бир түрдөгү маселелердин түрлөрү үчүн түзүлөт. Жогорудагы «$ax^2 + bx + c = 0$ $(a \neq 0)$ көрүнүшүндөгү квадраттык тендемени чыгаруу» алгоритми каалагандай a, b, c сандар үчүн натыйжа берет, б. а. алгоритмдин жалпылуулук касиети орундуу болуп саналат.

Төмөндө берилген эки натуралдык сандын эң чоң жалпы бөлүүчүсүн (ЭЧЖБ) табуунун **Евклид алгоритми** да бардык натуралдык сандар үчүн орундуу болот.

2-мисал. N жана M натуралдык сандардын ЭЧЖБ табылсын.

1) N жана M сандардынын мааниси табылсын.

2) эгерде $N = M$ болсо, натыйжа N деп алынсын жана 4-пунктка өтүлсүн;

3) N жана M сандардын чоңун өзү менен кичине сандын айырмасына барабар деп алынсын жана 2-пунктка өтүлсүн;

4) аяктасын.

Жыйынтыктап муну айтуу мүмкүн: жогорудагы бардык касиеттер аткарылганда көрсөтмөлөрдүн удаалаштыгы алгоритм болот жана кандайдыр бир (оң же терс) натыйжага алып келет.

Суроо жана тапшырмалар

1. Алгоритмдин кандай негизги касиеттери бар?

2. *Түшүнүктүүлүк касиети аткарыла турган жана аткарылбай турган көрсөтмөгө мисалдар келтир.*
3. *Көрсөтмөлөр аткаруучусуна түшүнүктүү болушу үчүн кандай системадан алынышы керек?*
4. *Аткаруучу алгоритмди механикалык түрдө аткаруусу үчүн кандай касиет маанiге ээ болот?*
5. *Алгоритмдин дискреттүүлүк касиетин мисалдардын жардамында түшүндүр.*
6. *Алгоритмдин натыйжалуулук касиетин мисалдардын жардамында түшүндүр.*
7. *Натыйжалуулук касиети аткарылбай турган көрсөтмөлөрдүн удаалаштыгына мисалдар келтир.*
8. *Алгоритмдин жалпылуулук касиетин мисалдардын жардамында түшүндүр.*
9. *Евклид алгоритминин жардамында бир канча натыйжа ал.*

6-сабак. Алгоритм түшүнүгү жана алгоритмдин негизги касиеттери темаларын кайталоо

1. Аткаруучу катарында төмөнкү көрсөтмөлөрдөн кайсыларын аткара албайсың жана эмне үчүн?

А. 200 кг дуу таш көтөрүлсүн.

Б. 7 ге 2 көбөйтүрүлсүн.

В. 1 ден 31622400000 гече саналсын.

2. Алгоритм аткаруучусу коюлган максатка жетишүү үчүн кандай жөнөкөй көрсөтмөлөрдү аткара алышы керектигин, б. а. аткаруучунун көрсөтмөлөр системасын аныкта.

А. Ачык эшик аткаруучунун сол жагынан 5 кадам нарыда болсо, максат «эшиктен чыгуу».

Б. Аткаруучу чорго жана цилиндр формасындагы стакандын алдында турган болсо, максат «жарым стакан суу алуу».

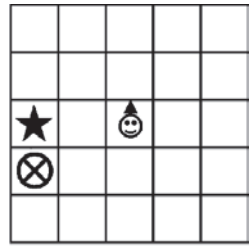
В. Берилген 44·15+12·15:20−43 сандуу туюнтманын мааниси аныкталсын.

3. Берилген көрсөтмөлөрдүн жардамында маселенин чечимине алып келүүчү алгоритм жаз.

А. «Карышкыр, эчки жана капуста» аттуу байыркы маселе. Дыйкан дарыянын сол жээгинде карышкыр, эчки жана капуста менен турат. Ал булардын бардыгын оң жээкке өткөрүшү керек. Анын кайыгы өтө кичине болгондуктан дыйкан өзү менен кошо бир гана жолоочуну — же карышкырды, же эчкини, же капустаны алышы мүмкүн.

Дагы — эгерде карышкыр менен эчки бир жээкте калтырылса карышкыр эчкини жеп коёт, эгерде эчки менен капуста бир жээкте калтырылса эчки капустаны жеп коёт. Жаныбарлар дыйкан бар болсо гана тынч турушат. Дыйкандын көрсөтмөлөр системасы төмөнкүчө:

{эчкини өткөр; карышкырды өткөр; капустаны өткөр; сүзүп өт}.



Б. Боорсок үчүн «алдындагы» чакмак калпагы көрсөтүп жаткан чакмак болуп саналат. Ал оңго бурулганда көрүнүштө болот. Боорсок 1 кадам алдындагы чакмакка жүрө алат же турган чакмагында оңго бурула алат, б. а. **{алдыга; оңго}** көрсөтмөлөрүн аткара алат. Боорсок бир чакмактан бир канча жолу өтүшү мүмкүн, бирок ⊗ формасындагы тосмолуу чакмактан өтө албайт. Боорсок өзү турган чакмактан ★ менен белгиленген чакмакка кандайдыр бир жол менен бара ала турган болсо, зарыл көрсөтмөлөрдүн удаалаштыгын жаз.

7-сабак. Алгоритмди сүрөттөө ыкмалары

Мурдагы сабактарда алгоритмдер сөз аркылуу туюнтулду. Дагы айта кетчү нерсе, алгоритмдерди сүрөттөөнүн да түрдүү ыкмалары бар. Төмөндө алгоритмдерди сүрөттөөнүн кеңири таралган ыкмалары көрүп чыгылат:

1. Алгоритмдин сөздөр жардамында туюнтулушу.

Өткөн сабактарда келтирилген бир топ мисалдар адам оозеки речинде колдонула турган сөздөр аркылуу туюнтулган эле (мисалы, чай демдөө же сумманы эсептөө алгоритми). Алгоритмдин мындай сүрөттөө ыкмасында аткаруучу үчүн көрсөтмө сүйлөмдөр аркылуу команда түрүндө берилет.

Мисал катары суу бассейни алдында турган **А** литрлүү жана **В** литрлүү суу идиши бар аткаруучу үчүн **{А ны толтур; Б ны толтур; А дан Б га куй; Б дан А га куй; А ны бошот; Б ны бошот}** көрсөтмөлөр системасын алуу мүмкүн. Бул аткаруучуга тиешелүү маселенин максаты өлчөп алынышы керек болгон суунун өлчөмүнүн *А* же *Б* идиштерден бирөөсүндө пайда болушу болуп эсептелет.

1-маселе. А=3 жана **Б=5** болгондо Суучу **1 литр** суу өлчөп алуусу үчүн алгоритм түзүлсүн. Бул маселенин максатка жеткирүүчү алгоритмин сөздөрдүн жардамында түзүү ынгайлуу:

Кадамдар	Алгоритмдеги көрсөтмөлөр	А идиште	В идиште
1	А ны толтур;	3 литр	0 литр
2	А дан В га куй;	0 литр	3 литр
3	А ны толтур;	3 литр	3 литр
4	А дан В га куй.	1 литр	5 литр

2. Алгоритмдин формулалар жардамында туюнтулушу.

Бул ыкма математика, физика, химия, биология сыяктуу илимдерде көп пайдаланылат. Эсинде болсо, сөздөрдүн жардамында туюнтулган

4-сабактагы 2-мисалда алгоритмди формула аркылуу туюнткан элек. Формуладагы «+», «−», «×», «:» сыяктуу арифметикалык амалдардын эсептөө эрежелерине амал кылган абалда аткарылышы да алгоритмге мисал болот. 5-сабакта берилген «$ax^2 + bx + c = 0$ $(a \neq 0)$ көрүнүшүндөгү квадраттык теңдемени чыгаруу» алгоритминин төмөндө келтирилген формула аркылуу туюнтмасы менен таанышасын:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

3. Алгоритмдин жадыбал аркылуу туюнтулушу.

Алгоритмдин мындай берилиши да сага тааныш. Мисалы, мектептеги сабак жадыбалы, Пифагордун көбөйтүү жадыбалы, лотореянын утуштар жадыбалы, химиялык элементтер жадыбалы. Мындай жадыбалдардан пайдалануу белгилүү бир алгоритмди колдонууну талап кылат.

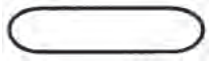
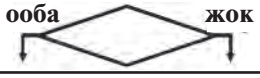
Кандайдыр бир функциянын графигин чийүү үчүн да функциянын аргументтик маанилерине окшош маанилердин жадыбалын түзөбүз. Бул да алгоритмдин жадыбал көрүнүшүнө мисал болот. Мисалы, $y = x^2$ алгоритм негизинде аракет кылып жаткан аткаруучу өтө турган чекиттердин кээ бирлери көрсөтүлгөн төмөнкү жадыбал менен математикадан таанышсын:

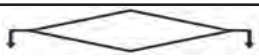
x	−3	−2	−1	0	1	2	3
y	9	4	1	0	1	4	9

4. Алгоритмдин графикалык көрүнүштө туюнтулушу.

Алгоритмдин мындай туюнтулушу менен математикадагы функциянын графиги, керектүү үйдү оңой табуу үчүн кварталдарда орнотулган үйлөрдүн жайгашуу схемасы, автобустардын багытынын схемасы аркылуу таанышсын.

Алгоритмдөө негиздерин үйрөнүүнүн дагы бир ыңгайлуу графикалык фигурасы **блок-схема** ыкмасы болуп саналат. Блок-схемалар багыт сызыктары аркылуу туташтырылган белгилүү бир команда же көрсөтмөнү чагылдыруучу атайын геометриялык фигуралар – **блок**тордон түзүлөт:

	алгоритмдин башталгандыгын жана аяктагандыгын билдирет
	маалыматтарды киритүү жана чыгарууну билдирет
	жөнөкөй аракетти, б. а. маани берүү же тиешелүү көрсөтмөлөрдү берүүнү билдирет
	шарттын текшерилишин билдирет

	кайталануунун башталышын билдирет
	жардамчы алгоритмге кайрылууну билдирет
	схемадагы аракет багытын билдирет
:=	маани берүү көрсөтмөсү

2-маселе. Радиусу *R* ге барабар болгон тегеректин аянтын эсептөө алгоритмин түз.

Бул маселенин алгоритми эки түрдүү ыкмада сөздөрдүн жардамында жана графикалык түрдө түзүлөт:

1) башталсын;
2) *R* дин мааниси аныкталсын;
3) *R* ди *R* ге жана 3,14 кө көбөйтүп *S* деп алынсын;
4) жооп катары *S* жазылсын;
5) аяктасын.

5. Алгоритмдин программа түрүндө туюнтулушу.

Бизге белгилүү болгондой, компьютер программалардын негизинде иштейт жана башкарылат. Сен бүгүнкү күнгө чейин MS Word, MS Paint жана MS Excel сыяктуу колдонмо **программалар** менен иштедиң. Дагы айта кетчү нерсе, ар бир колдонмо **программа** да өтө чоң жана татаал алгоритмдин бир көрүнүшү болуп эсептелет. Демек, бул өндүү алгоритмдер аткарылышы үчүн **алгоритмдин аткаруучусуна**, б.а. **компьютерге түшүнүктүү болушу керек**.

Адатта, алгоритмдин компьютер түшүнө турган тилде жазылышы **программа** деп аталат. Компьютер түшүнө турган тил болсо **программалоо тили** деп аталат. Дүйнөдө миңдеген программалоо тилдери бар жана алар дагы өнүгүп барууда. Бүгүнкү күндө **BASIC, Pascal, VBA, Delphi, C, C++** программалоо тилдери кеңири таралган жана үйрөнүү үчүн ыңгайлуу.

Суроо жана тапшырмалар

1. *Алгоритмдин сүрөттөө ыкмалары жөнүндө маалымат бер.*
2. *Алгоритмдин сөздөр аркылуу туюнтулушуна турмуштук мисалдар келтир.*
3. *Кайсы илимдерде алгоритмди формулалардын жардамында берүү ыңгайлуу?*
4. *Алгоритмдин формулалар аркылуу туюнтулушуна физика предметинен мисалдар келтир.*

5. Алгоритмдин жадыбал түрүндө берилишине мисалдар келтир.
6. Алгоритмдин графикалык түрдө берилишине мисалдар келтир.
7. Блок-схема деген эмне?

Көнүгүүлөр

1. MS Paint графикалык редакторунда **«Өзбекстан келечеги улуу мамлекет!»** текстиндеги сөздөрдү сап-сап кылып жазуу үчүн сөздөрдүн жардамында алгоритм түз.

2. MS Word программасынын WordArt объекти жардамында **«Өзбекстан мекеним менин!»** сөзүн жазуунун алгоритмин ыңгайлуу ыкмада сүрөттө.

3. Берилген эки натуралдык сандын эң кичине жалпы көбөйтүүчүсүн (ЭКЖК) табуунун алгоритмин түз.

8-сабак. Алгоритмди сүрөттөө ыкмалары темасына тиешелүү практикалык көнүгүү

1. Төмөнкү маселелердин алгоритмдерин сөздөрдүн жардамында түз.

А. Берилген x те $y = 23 \cdot x - 1963$ функциясынын маанисин эсептөөнүн алгоритмин түз.

Б. Аткаруучунун көрсөтмөлөр системасы бир гана **{5 ти кош; 3 түкемит}** көрсөтмөлөрүнөн турат. Бул аткаруучу 0 санынан 11 санын пайда кылышы үчүн алгоритм түз.

В. Аткаруучунун көрсөтмөлөр системасы бир гана **{1 ди кош; 2 ге көбөйт}** көрсөтмөлөрүнөн турат. Бул аткаруучу 0 санынан 17 санын пайда кылышы үчүн 3 түрдүү ыкмада алгоритм түз.

Г. $A = 5$ жана $Б = 8$ болгондо аткаруучу 4 литр суу өлчөп алышы үчүн алгоритм түз.

Көмөк: $Б$ жана B маселелерин чечүүдө төмөнкүдөй жадыбал түзүү алгоритм кадамынын аткарылуу натыйжасын көрүп туруу мүмкүнчүлүгүн берет:

Кадам	Көрсөтмө	Натыйжа
0	–	0
1		
2		

2. Төмөнкү маселелердин алгоритмдерин блок-схеманын жардамында түз.

А. Радиусу R ге барабар болгон айлананын ичине сызылган квадраттын жактарын табуу алгоритмин түз.

Б. Үч тыйын берилген. Алардан бири жасалма жана оор. Тартуу үчүн рычагдуу тараза өлчөө таштарысыз берилген. Жасалма тыйынды аныктоо алгоритмин түз.

В. Үч тыйын берилген. Алардан бири жасалма жана салмагы менен гана айырмаланат (анык оор же жеңилдиги да белгилүү эмес). Тартуу үчүн ры-

чагдуу тараза өлчөө таштарысыз берилген. Эң аз тартуу аркылуу жасалма тыйынды аныктоо алгоритмин түз.)

9-сабак. Алгоритмдин негизги түрлөрү

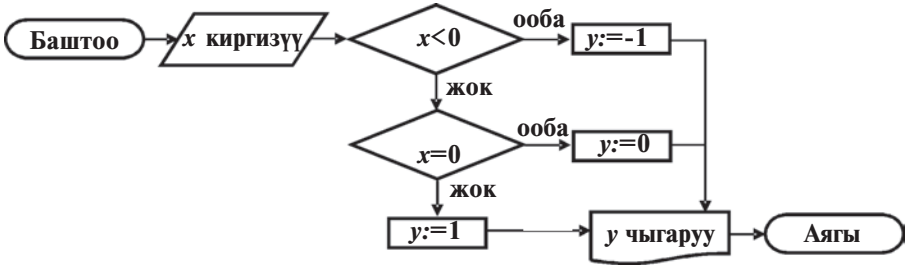
Ар кандай алгоритм логикалык түзүлүшүнө, б.а. аткарылуу тартибине карай үч негизги түргө бөлүнөт: **сызыктуу (удаалаш), багытталуучу жана кайталануучу.**

Сызыктуу алгоритмдер. Бардык көрсөтмөлөрү удаалаш жайгашуу тартибинде аткарып барыла турган алгоритмдер **сызыктуу алгоритмдер** деп аталат. «Чай демдөө», тегеректин аянтын эсептөө алгоритмдери сызыктуу алгоритмдерге мисал болот. Бирок турмуштагы өтө көп жараяндар шарттардын негизинде башкарылат.

Тармакталуучу алгоритмдер. Шартка ылайык аткарыла турган көрсөтмөлөр катышкан алгоритмдер **тармакталуучу алгоритмдер** деп аталат. Алгоритмдердин бул түрү турмушта ар күнү жана ар кадамда кездешет. Эшиктен чыгышыбыз эшик ачык же жабыктыгына, тамактанышыбыз курсагыбыз ач же токтугуна же тамактын түрүнө, көчөгө кийинип чыгышыбыз аба-ырайына, кандайдыр бир жайга баруу үчүн транспорт каражатын тандашыбыз төлөө мүмкүнчүлүгүбүз болгон акчага байланыштуу болуп эсептелет. Демек, тармакталуучу алгоритмдер сызыктуу алгоритмдерден тандоо мүмкүнчүлүгү менен айырмаланат экен. Өткөн сабактардагы квадраттык теңдемени чыгаруу, эки сандын ЭКЖБ сүн табуу алгоритмдери тармакталуучу алгоритмдерге мисал болот.

1-мисал. Алгоритм $y = \begin{cases} -1, \text{ эгерде } x < 0 \\ 0, \text{ эгерде } x = 0 \\ 1, \text{ эгерде } x > 0 \end{cases}$ формула аркылуу берилген.

Функциянын маанисин эсептөө боюнча тармакталуучу алгоритм блок-схеманын жардамында сүрөттөлөт:



2-мисал. Берилген эки A жана B сандардан чоңун табуу үчүн алгоритм түз.

1) башталсын;
2) A жана B киргизилсин;
3) эгерде $A > B$ болсо
   4-пунктка өтүлсүн, антпесе
   5-пунктка өтүлсүн;
4) натыйжа A деп алынсын
жана 6-пунктка өтүлсүн;
5) натыйжа B деп алынсын;
6) аяктасын.

Бул мисалдан төмөнкүчө жыйынтык чыгаруу мүмкүн: эгерде $A > B$ шарт аткарылса 5-пункттагы көрсөтмө каралбайт, антпесе, б.а. $A \leq B$ болгондо 4-пункттагы көрсөтмө каралбайт. Бул алгоритм тармакталууну аныктоо элестетүү мүмкүнчүлүгүн берет.

Кайталануучу (циклдүү) алгоритмдер. Маселелерди иликтөө жараянында алгоритмдеги кээ бир көрсөтмөлөрдүн кайра аткарылышын күзөтүү мүмкүн. Мисалы, эң чоң квадраттарды кесип алуу маселеси (4-сабак 5-мисал), Евклид алгоритми (5-сабак 2-мисал).

Турмушта да өтө көп жараяндар кайталанат. Мисалы, сабактардын ар жума кайталанышы, ар күнү эртең менен чай ичүү же мектепке баруу ж. б. Көрсөтмөлөрү кайра аткарыла турган алгоритмдер **кайталануучу алгоритмдер** деп аталат.

Кайталануучу алгоритмдер «I := I + 1», «S := S + I» же «P : = = P * I» көрүнүштөгү көрсөтмөлөрдүн катышуусу менен айырмаланып турат (* – көбөйтүү амалы). Мындай көрсөтмөлөрдүн мазмунун түшүнүү үчүн кайталануунун бир канча кадамын көрүп чыгуу керек.

Адатта, сумма үчүн башталгыч мааниси (англисчеден SUMMARY, б.а. сумма деген сөздүн баш тамгасы) S:=0 жана көбөйтүндү үчүн (англисчеден PRODUCT, б.а. көбөйтүндү деген сөздүн баш тамгасы) P: = 1 деп алынат, анткени бул маанилер, б.а. 0 жана 1 лер, ылайыктуу түрдө, сумма жана көбөйтүндүнүн натыйжасына таасир кылбайт:

1-кадам: I := 1болсун, анда S := S + I = 0 + 1 = 1, P : = = P*I = 1*1 = 1;

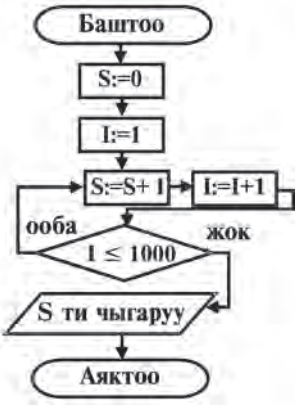
2-кадам I := I + 1 = 1 + 1 = 2, S := S + I = 1 + 2 = 3, P := = P*I = 1*2 = 2;

3-кадам: I := I + 1 = 2 + 1 = 3, S := S + I = 3 + 3 = 6, P := = P*I = 2*3 = 6;

4-кадам: I := I + 1 = 3 + 1 = 4, S := S + I = 6 + 4 = 10, P : = = P*I = 6*4 = 24.

3-мисал. 1 ден 1000 ге чейинки сандардын суммасын, б.а. S = = 1+2+3+...+1000 ди эсептөө алгоритмин түз.

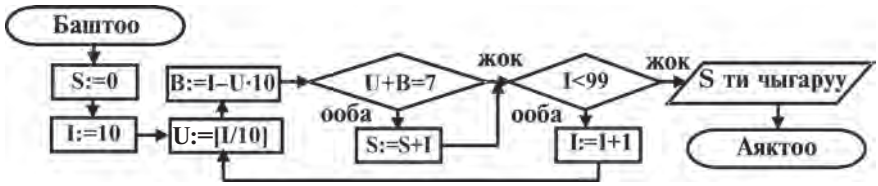
1) башталсын;
2) S = 0 деп алынсын (б.а. S: = 0);
3) I нин мааниси 1 деп алынсын (б.а. I : = 1);
4) S ке I кошулуп S деп алынсын (б.а. S: = S+I);
5) I ге 1 кошулуп I деп алынсын (б.а. I: = I+1);
6) эгерде I ≤ 1000 болсо 4-пунктка өтүлсүн;
7) жооп S деп алынсын;
8) аяктасын.



Сөздөр менен туюнтулган алгоритмде блок-схема менен шайкештикти көрсөтүү үчүн кашаалардын ичинде түшүндүрмө берип бардык. Адатта, кайталануучу алгоритмдерде «I:=I+1» сыяктуу туюнтма **эсептегич** деп жүргүзүлөт. Бул мисалдын чечимин сызыктуу алгоритм түрүндө да түзүү мүмкүн. Ал үчүн ар кандай натуралдык *N* сан үчүн орундуу болгон 1+2+3+...+ N ≡ N·(N+1):2 туюнтмадан пайдалануу жетиштүү (алгоритмди өз алдынча түз).

Төмөнкү мисалдарда бул сыяктуу иш бир кыйла кыйын.

4-мисал. Эки орундуу сандардын ичинен цифраларынын суммасы 7 ге барабар сандардын суммасын эсептөө алгоритмин түз ([*a*] – а санынын бүтүн бөлүгү, / – бөлүү амалы).



5-мисал. «Мекен – сыйынаар жерим» сүйлөмүн 20 жолу жаздыруу алгоритмин түз. Бул мисалдын алгоритми сөздөр аркылуу туюнтулат.

1) I дин мааниси 1 деп алынсын;
2) «Мекен – сыйынаар жерим» жазылсын;
3) I ге 1 ди кошуп I деп алынсын;
4) эгерде I ≤ 20 болсо, 2-пунктка өтүлсүн;
5) аяктасын.

Көрүп өтүлгөн алгоритмдерге көңүл бурулса, алгоритмдер сызыктуу, тармакталуучу же кайталануучу бөлүктөрдөн түзүлгөндүгүн көрүү мүмкүн. Демек, адамдын турмушунда кезиге турган алгоритмдер, негизинен, ушул үч түрдөгү алгоритмдердин үзгүлтүксүз биримдиги түрүндө пайда болот.

Суроо жана тапшырмалар

1. *Кандай алгоритм сызыктуу алгоритм деп аталат? Мисалдар келтир.*
2. *Кандай алгоритм тармакталуучу деп аталат? Мисалдар келтир.*
3. *Кандай алгоритм кайталануучу деп аталат? Мисалдар келтир.*
4. *Сызыктуу, тармакталуучу жана кайталануучу алгоритмдердин бири-биринен айырмасын түшүндүр.*
5. *Үч сандан чоңун (ҮСЧ) аныктап берүүчү алгоритм түз.*

Көнүгүүлөр

1. Төмөнкү алгоритмдер кандай алгоритм түрүнө мисал болушун жана натыйжасын аныкта:

а) a:=3; x:=2*a+a*a. a=?, x=?

б) x:=1; x:=x+11, x:=x*x–4. x=?

в) a:=15; b:=a; a:=a–b. a=?, b=?

г) 1) a:=3;
2) эгерде a>2 болсо, анда x:=2*a+a*a жана 4-пунктка өтүлсүн, антпесе 3-пунктка өтүлсүн;
3) x:=9–a*x;
4) натыйжа x жазылсын;
5) аякталсын.

д) 1) x:=1;
2) эгерде x > 2 болсо, анда x:=x+11 жана 4-пунктка өтүлсөн, антпе - се 3-пунктка өтүлсүн;
3) x:=x*x–4;
4) натыйжа x жазылсын;
5) аякталсын.

е) 1) a:=15;
2) b:= a;
3) эгерде a > b болсо, анда a:=a–b жана 5-пунктка өтүлсүн, антпе - се 4-пунктка өтүлсүн;
4) *a:=a+b;*
5) натыйжа a, b жазылсын;
6) аякталсын.

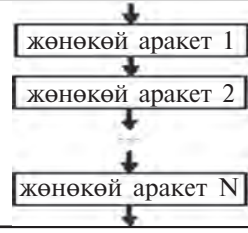
2. Берилген сандын белгисин аныктоочу алгоритмди блок-схема жардамында түз.

3. $y = x^2 - 1$ функциясынын маанилерин x тин [1; 10] аралыгындагы бүтүн маанилеринде эсептөө алгоритмин блок-схема түрүндө түз.

10-сабак. Алгоритмдин өзөктүк структураларына тиешелүү практикалык тапшырма

Айтып өтүлгөндөй, ар кандай алгоритмди сызыктуу, тармакталуучу жана кайталануучу алгоритмдердин өз ара шайкеш бирикмеси түрүндө сүрөттөө мүмкүн. Ошондуктан төмөндө келтириле турган структураларды өздөштүрүп алуу максатка ылайык болот. Бул структуралардын өзгөчө маанилүү жагы аларда бир эле кирүү жана бир эле чыгуу бар экендигинде.

1. Сызыктуу структура. Удаалаш келе турган жөнөкөй аракет, б.а. маани берүү же тиешелүү көрсөтмөлөр берүүдөн гана турат.

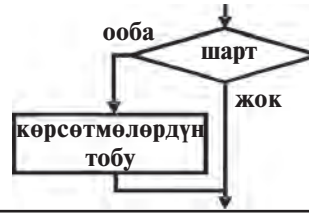
Сөздөр аркылуу	Блок-схема түрүндө
жөнөкөй аракет 1 жөнөкөй аракет 2 ... жөнөкөй аракет N	

1-маселе. Үч *a, b, c* сандар берилген. *a* жана *b* сандардын суммасынын жарымын, *a* жана *c* сандардын айырмасынын модулун, *b* жана *c* сандардын көбөйтүндүсүнүн квадратын эсептөө алгоритмин түз.

2-маселе. Радиусу *R* ге тең айлананын узундугу, тегеректин аянты жана шардын көлөмүн эсептөөнүн алгоритмин түз көмөк: $L = 2\pi R$; $S = \pi R^2$; $V = \frac{4}{3}\pi R^3$).

2. Тармакталуу структуралары. Бул структуралар шарттын текшерилүү натыйжасына (ооба же жок) ылайык эки жолдон бирин тандоо мүмкүнчүлүгүн берет. Бул структуралар, негизинен, 2 түрдүү көрүнүштө болот.

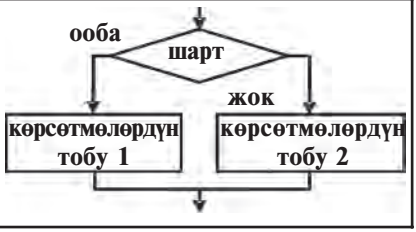
а) эгерде – анда:

Сөздөр аркылуу	Блок-схема түрүндө
эгерде шарт **анда** көрсөтмөлөрдүн тобу **аягы**	

3-маселе. Берилген *a* сан терс болсо, анда анын квадратын жана квадраттык тамырын эсептөө алгоритмин түз.

4-маселе. Үч *a, b, c* сандар берилген. Алардын ичинде терс болбогон сандардын квадраттык тамырын эсептөө алгоритмин түз.

б) эгерде – анда – антпесе:

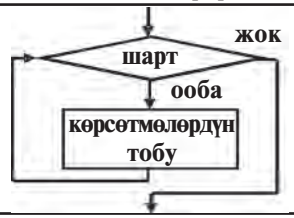
Сөздөр аркылуу	Блок-схема түрүндө
эгерде шарт **анда** көрсөтмөлөрдүн тобу 1 **антпесе** көрсөтмөлөрдүн тобу 1 **аягы**	

5-маселе. Үч *a, b, c* сандар берилген. $a < b - c$ шарт аткарылса, «Ооба», антпесе «Жок» деп жооп чыгаруучу алгоритм түз.

6-маселе. *a* жана *b* сандар берилген. Эгерде алардын көбөйтүүчүсү оң болсо, алардан ар биринин квадратын, антпесе алардын ар бирине 100 дү кошуп чыгаруучу алгоритм түз.

3. Кайталануу структуралары. Бул структуралар бир канча көрсөтмөлөрдүн тобунун көп жолу аткарылышын камсыздайт. Бул структуралар да, негизинен 2 түрдүү көрүнүштө болот.

а) чейин:

Сөздөр аркылуу	Блок-схема түрүндө
чейин шарт көрсөтмөлөрдүн тобу **аягы**	

7-маселе. Натуралдык *x* тин мааниси берилген *a* санынан кичине болгондо $y = ax^2+20$ функциянын маанилерин эсептөө алгоритмин түз.

8-маселе. Берилген *A* жана *B* оң сандардын маанилери барабар болгончо бул сандардын чоңунан кичинесин кемитип, чоңу менен алмаштырып баруучу алгоритм түз.

б) параметр ... дан ... чейин:

Сөздөр аркылуу	Блок-схема түрүндө
параметр Б **дан** А **га чейин** көрсөтмөлөрдүн тобу **аягы**	

Мында параметр эсептегич сыяктуу каралып жаткандыктан ал *Б* менен *А* дан айырмалуу каалагандай тамга болушу мүмкүн.

9-маселе. «Өзбекстан — келечеги улуу мамлекет!» сүйлөмүн ушул окуу жылында Мекенибиздин эгемендүүлүгүнүн белгиленген санына чейин жаздыруу алгоритмин түз.

Эми жогоруда келтирилген структуралардын жардамында өткөн сабактарда берилген маселелерди оңой эле туюнтуу мүмкүн.

Суроо жана тапшырмалар

1. *Сызыктуу алгоритмге тиешелүү структурага мисалдар келтир.*
2. *Тармакталуучу структуралардын кандай көрүнүштөрү бар?*
3. *Тармакталуучу структуралардын ыңгайлуу көрүнүшүнүн колдонулушуна тиешелүү мисалдар келтир.*
4. *Кайталануучу структуралар жөнүндө маалымат бер.*
5. *Кандай маселелерде чейин структурасын колдоо максатка ылайык?*

Көнүгүүлөр

1. Ылдамдыгы ϑ км/саат болгон машинанын T саатта басып өткөн жолун эсептөө алгоритмин түз.

2. Радиустары тиешелүү түрдө, R_1, R_2, R_3 кө тең болгон тегеректердин жалпы аянттарынын квадратын эсептөө алгоритмин түз.

3. Жактары, тиешелүү түрдө a жана b болгон квадраттардын аянттарынын айырмасынын модулун табуучу алгоритм түз.

4. Эки a жана b сандары берилген. Эгер де b саны a дан кичине болсо, анда b ны нөл менен алмаштыруучу, антпесе b ны өзгөрүүсүз калтыруучу алгоритм түз.

11-сабак. Кайталоого тиешелүү тапшырмалар

1. Компьютерде маселе чыгаруунун биринчи үч баскычын жагы a га барабар болгон квадратка ичинен сызылган тегеректин аянтын табуу маселесинин негизинде көрсөтүп бер.

2. Компьютерде маселелер чыгаруунун биринчи үч баскычын төмөнкү маселени чечүүнүн негизинде көрсөтүп бер: 50 литрдүү идиште 5 килограмм аш тузу кошулган 20 литр аралашма бар. Эгерде идишке дагы 10 литр суу кошулса, аралашмадагы туздун өлчөмүн пайыз эсебинде тап.

3. a, b, c сандары берилген. $a+b+c<0$ шарт аткарылса $y = a^2 - b^2$ ты, антпесе $y = a^2+c^2$ ты эсептөө алгоритмин түз.

4. -100 дөн 50 гө чейинки сандардын аралыгындагы так сандардын көбөйтүндүсүн эсептөө алгоритмин түз.

II БӨЛҮМ ПРОГРАММАЛООНУН НЕГИЗДЕРИ

12-сабак. Программа жана программалоо тилдери

Бизге белгилүү болгондой, компьютер технологиясынан өнүмдүү пайдалануу эки бөлүк – техникалык жана программалык маалыматтардын байланышын талап кылат. Бул байланыш компьютердин техникалык камсыздалуусунун ылдам темптерте күчөп баруусуна ылайык программалык камсыздалуунун да кескин темптер менен өнүгүшүнө себеп болот, жана тескерисинче. Мунун себеби белгилүү, тиешелүү программалык камсыздалуусуз ар кандай компьютер «кымбат баалуу оюнчук» болуп кала берет.

Өткөн сабактарда маселени компьютерде чечүүдө керек боло турган объект, модель жана алгоритм түшүнүктөрү жөнүндө маалымат алдың. Белгилүү болгондой, компьютерде кандайдыр бир маселени чыгаруу үчүн, оболу, анын модели жана алгоритми түзүлөт, андан соң ошол алгоритм белгилүү бир мыйзам-эрежелердин негизинде компьютер түшүнө турган тилде көрсөтмө жана командалар көрүнүшүндө жазылат. Пайда болгон компьютер түшүнө турган тилдеги текст **программа тексти,** алгоритм болсо **программа** деп аталат.

Компьютер үчүн программа түзүү жараяны **программалоо** жана программа түзө турган адам **программист** дейилет. Компьютер түшүнө турган тил болсо **программалоо тили** деп аталат.

Программалоо тилдерин шарттуу түрдө төмөнкү үч топко бөлүү мүмкүн:

Төмөнкү даражадагы программалоо тилдери компьютер курулмалары менен түздөн-түз байланыштуу болуп, командалар атайын сан (код)дар жардамында жазылат. Бул сыяктуу коман-

далардан түзүлгөн программалар чоң көлөмдүү болуп, аларды редакциялоо бир кыйла татаал иш болуп эсептелет. Алгачкы электрондук эсептөө машиналарында («ENIAC», «МЭСМ» ж.б.) маселелерди чыгаруу үчүн мына ушундай командалардын жардамында программалар түзүлгөн.

Программалоо тилдери тарыхынан. Программалоо тилдери, негизинен, экинчи дүйнөлүк согуштан кийин жаратыла баштады. Бирок анын башталуу тарыхы бир кыйла узак жылдарга барып такалат.

Археологиялык казылмаларда табылган чопо тактайчада мындан 3800 жыл мурда (б.з.ч. 1800-жылдар) Вавилондо пайыз менен байланыштуу татаал амалдар алгоритми келтирилген. Анда анык маселе иштелген болуп, эгерде буудай түшүмү жылына 20% дан артып барса, анын өлчөмү эки эсе өсүшү үчүн канча жыл жана ай керек болушунун алгоритми түзүлгөн.

Чарлз Беббиж

XIX кылымда француз **Жозеф Мари Жаккард** 1804-жылда жука кездеме өндүрүү жараянында токуу станоктору үчүн перфокартаны эстетүүчү тасма иштеткен жана ушунусу менен перфокартага негиз салган эле.

1836-жылда англис окумуштуусу **Чарлз Беббиж** азыркы компьютерлердин түздөн-түз башкы мууну болгон аналитикалык машина жаратууга киришти жана *бул маселени теориялык түрдө чечти.* Бул машинанын негизги өзгөчөлүгү анын программанын негизинде иштеши жана эсеп-кысап натыйжаларын «эстеп» калышында эле.

Ада Лавлейс

1843-жылда англис математиги **Огаста Ада Байрон** (Лавлейс) — акын лорд Байрондун кызы аналитикалык машина командалардын негизинде иштеши керектигин айтты. Ал берилген шарттар аткарылбаганга чейин кадамдардын удаалаштыгын камсыздоочу командаларды жазды. Мына ушул жагдай менен ал программалоо тилине негиз салды. Ушул жана башка ачылыштар компьютер жаратылгандан соң, аларды иштетүү үчүн зарыл болгон тилдин жаратылышын талап кылды.

Программа түзүүнү жеңилдетүү максатында адамдын тилине жакын болгон командалар системасын колдоо маселеси коюлду жана чечилди. Бул сыяктуу программалоо тилдери **орто даражадагы программалоо тилдери** (кээде ассемблерлер) деп айтыла башталды. Мындай тилдерге **AVTOKOD-BEMSH, AVTOKOD-MADLEN** жана башкалар кирет. Алар **BESM-6, Minsk-22, Minsk-32, IBM-360** электрондук эсептөө машиналарында колдонулат. Мисалы, **ST 5, BSUM** туюнтма 5 цифрасын **BSUM** деп аталган ячейкага жайгаштырылсын (**ST**-store - жайгаштыруу) деген команданы берет.

Жогорку даражалуу программалоо тилдериндеги көрсөтмөлөр адам тилине жакын болгон сөздөр жыйнагынан турат. Алар жардамында амалдарды аткаруу төмөнкү даражадагы тилдерге караганда жеңил болуп, программисттен дээрлик адрестер жана курулмалар менен түздөн-түз байланыштуу

маалыматтарды билүү талап кылынбайт. Бул тилде түзүлгөн программаларды компьютерлер аткара алышы үчүн **трансляторлор** деп аталуучу атайын программалар цифралуу көрүнүшкө өткөрүп берет.

Соңку жылдарда өтө көп жогорку даражадагы программалоо тилдери иштеп чыгарылган болуп, алардын катарына **Pascal, Ada, KARAT, C++, Delphi, Visual Basic Application, Java** сыяктуу тилдерди кошуу мүмкүн. Бүгүнкү күндө иштеп чыгарылып жаткан программалоо тилдери кандайдыр бир багыттагы маселелерди чечүүгө ылайыкташтырылган болуп, аларды **объектке багытталган программалоо тилдери** деп аташат.

Төмөнкү жадыбалда программалоо тилинин өнүгүү тарыхынан маалымат берилген.

Программалоо тили	**Иштеп чыгарылган жылы**
Планкалкюл	1946
Кыска код	1949
Ассемблер «Edsak», АО	1950
Автокод «Madlen»	1953
Ылдам коддоо	1955
A-2, Flou-metik	1956
IPL-1, Mat-metik	1957
Fortran	1958
Algol 58	1959
APT, LISP, Kobol, Algol-60	1960
PL/1, Basic	1964
Algol W	1965

Программалоо тили	**Иштеп чыгарылган жылы**
Logo	1967
Algol 68	1968
APL	1969
Paskal	1970
Fort	1971
Prolog, Ci, Ada	1972
Smoltok	1980
VBA	1990
VC++	1993
Java	1994
Delphi	1995
C #	2000

Бүгүнкү күнгө чейин иштеп чыгарылган программалоо тилдеринен кең тараганы **Паскаль** (Pascal – англ. тили) программалоо тили болуп саналат. Паскаль программалоо тили 1969-жылы Никалас Вирт тарабынан иштеп чыгылды. Паскаль тили программалоону үйрөтүү максатында иштеп чыгылган болсо да, жогорку тажрыйбалуу программисттердин арасында кеңири таралды. Албетте, эффективдүү программалоо тилдери өзгөрүүсүз калбайт. Ошол себептүү да түрдүү маркадагы компьютерлер үчүн Паскаль тилинин аларга ылайыкташтырылган варианттары иштеп чыгылган болуп, алар Паскаль тилинин башталгыч варианттарынан айырмаланышы мүмкүн.

Суроо жана тапшырмалар

1. Программа деп эмнеге айтылат?

2. Программалоо тили дегенде эмнени түшүнөсүң?

3. Түрдүү даражалуу программалоо тилдери кандай өзгөчөлүктөрү менен өз ара айырмаланат?

4. *Программалоо тилдери электрондук эсептөө машиналарынын түрүнө байланыштуу болобу? Жообуңду негизде.*
5. *Жогорку даражадагы программалоо тилдеринен бир канчасын иштеп чыгарылган жылы менен айтып бер.*

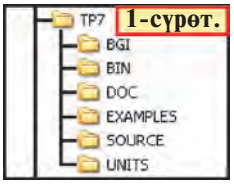
13-сабак. Турбо Паскаль 7.0 интегралдык чөйрөсү

Паскаль программалоо тилинин кеңири таралышы жана колдонулушуна негизги себеп – программанын жөнөкөйлүгү жана андан пайдалануунун ыңгайлуулугу болуп саналат. Эң оболу Паскаль тили университеттерде колдонулган болуп, кийинчерээк анын түрдүү маркадагы компьютерлер үчүн командалары иштеп чыгылды.

1981-жылда Паскаль тилинин эл аралык стандарты сунушталды. Паскаль тилинин Борланд фирмасы тарабынан иштеп чыгылган **Турбо Паскал 7.0** вариантынан азыркы мезгилде кеңири пайдаланылат. Ал пайдалануучулар үчүн өтө ыңгайлуу система – **программалоонун интегралдык чөйрөсүнө ээ.** Интегралдык чөйрө – программалоого жардам берүүчү программа болуп, ал төмөнкү негизги милдеттерди аткарышы керек:

- эң оболу, ал программанын текстин киритүү мүмкүнчүлүгүн берүүсү;
- маал-маалы менен киритилип жаткан программанын текстин сырткы эстутумда сактап туруусу;
- программаны ишке түшүрүү үчүн трансляторго ээ болуусу;
- синтаксистик каталыктарды аныктоо каражатына ээ болушу керек.

Турбо Паскаль 7.0 интегралдык чөйрөсү санап өтүлгөн милдеттерден сырткары дагы көптөгөн милдеттерди да ишке ашыруу мүмкүнчүлүгүн берет.



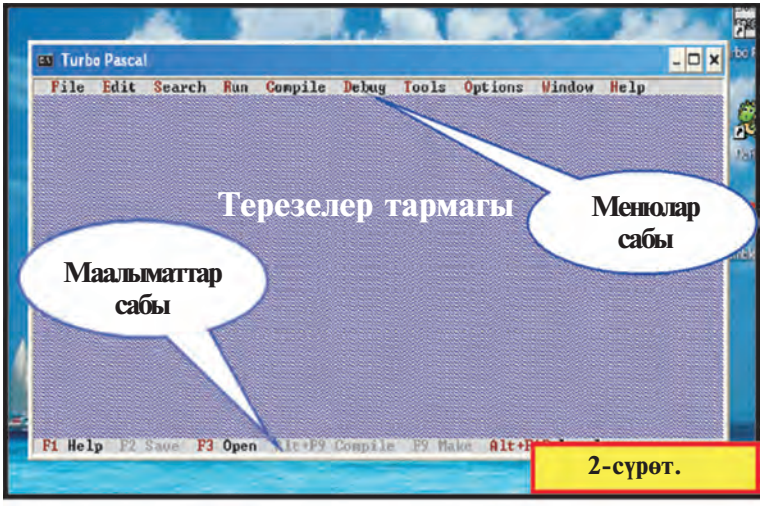
1-сүрөт.

Турбо Паскал 7.0 программалоо системасы сырткы эстутумдун, адатта, «TP7» аттуу каталогуна жайгаштырылат (жалпысынан алганда башка каталогго жайгаштыруу да мүмкүн). Ал жүздөн ашуун файлдан турган болуп, файлдар милдеттерине карай бир канча каталогдорго жайгаштырылган (1-сүрөт). Турбо Паскаль интегралдык чөйрөсүн ишке түшүрүүчү **Turbo.exe** файлы **«BIN»** каталогунда жайгашкан. Калган каталогдордо негизинен жардамчы файлдар жана Турбо Паскалдын мүмкүнчүлүктөрүн көрсөтүүчү программалар жайгашкан.

Мисалы, **«BGI»** каталогунда графикалык абалда иштөө үчүн зарыл болгон файлдар жайгашкан.

Турбо Паскалдын калган каталог жана файлдары жөнүндө окуу китебинде келтирилген кошумча адабияттар тизмесиндеги окуу колдонмолордон жетиштүү маалымат алышың мүмкүн.

Turbo.exe файлы ишке түшүрүлгөндөн соң, экранда Турбо Паскаль интегралдык чөйрөсүнүн интерфейси ачылат. Ал **менюлар сабы, терезелер тармагы жана маалыматтар сабы**нан турат (2-сүрөт). Турбо Паскалда бир канча түрдөгү терезелер болуп, алардан эң негизгиси **программанын текст** редакторунун терезеси болуп саналат. Бул терезени пайда кылуу үчүн **File** (Файл) менюсунун **New** (Жаңы) командасын тандоо жетиштүү.



2-сүрөт.

Менюлар сабына өтүү үчүн F10 клавишасы басылат. Андан соң солго же оңго багыттоо клавишаларынын жардамында керектүү менюну тандап ENTER клавишасы басылат. Керектүү менюну «чычкан» жардамында да тандоо мүмкүн.

Турбо Паскаль интегралдык чөйрөсүнүн интерфейси текст редакторунун интерфейсине окшойт. Ага программанын тексти программа редакторунун терезесинде текст редакторундагы сыяктуу жазылат. MS Wordдогу сыяктуу Турбо Паскалда да бир канча терезе ачылып, алардын ар бири менен өзүнчө иштөө мүмкүнчүлүгү бар. Бул бир мезгилде бир канча программа менен иштөө мүмкүнчүлүгүн берет. Учурда иштетилип жаткан терезе **активдүү терезе** деп аталат. Программанын текст редакто-

рунда тексттин аты **NONAME00.PAS** сыяктуу сунушталып жаткан болуп, тексттин аты **атсыз00** жана файл кеңейтмеси **pas** болушун билдирет.

File (Файл) менюсу Open (Ачуу – сырткы эстутумдагы файлды ылдам эстутумга жүктөө), Save (Сактоо), Save as (башка ат менен сактоо), Exit (чыгуу), **Edit (редакциялоо)** менюсу Cut (кыркып алуу), Copy (үлгү алуу), Paste (жайгаштыруу), **Run** менюсу программаны ишке түшүрүү, **Compile** менюсу программаны компиляция кылуу (программаны «машина тили»не оодарып «EXE» кеңейтилген файл көрүнүшүндө сактоо) амалдарын өз ичине алат.

Менюлардын курамындагы амалдарды белгилүү бир **(ылдам)** клавишаларды басуу аркылуу да аткаруу мүмкүн. Төмөнкү жадыбалда негизги амалдарды аткарууга ылайыкташтырылган клавишалардын тизмеси келтирилген:

F3	ачуу	**Ctrl+F9**	**программаны ишке түшүрүү**
F2	сактоо	**Alt+F5**	программанын натыйжасын экранда көрүү
Alt+F3	активдүү терезени жабуу	**Alt+F9**	программаны компиляция кылуу
Alt + x	чыгуу	**F6**	бир терезеден экинчи терезеге өтүү

Программанын текстин редакциялоодо төмөнкү клавишалардан пайдалануу мүмкүн:

Багыт клавишалары (←,→,↑,↓) – жүргүчтү керектүү багытта жылдыруу;

Shift +(←,→,↑,↓) – жүргүч турган жайдан баштап тандалган багытта программанын текстинин бөлүгүн белгилөө;

Ctrl+Insert – программанын текстинин белгиленген бөлүгүнүн үлгүсүн буфер-эстутумга алуу;

Shift+Insert – эстутумга алынган бөлүктү программа текстинин жүргүч турган жайына жайгаштыруу;

Shift+Delete – программанын текстинин белгиленген бөлүгүн кыркып алуу.

Суроо жана тапшырмалар

1. *Турбо Паскалды ишке түшүрүүчү файл кайсы каталогдо жайгашкан?*
2. *Программалоонун интегралдык чөйрөсү деген эмне?*
3. *Турбо Паскаль интегралдык чөйрөсүндө программанын текст редакторун ач.*
4. *NONAME.PAS эмнени билдирет? Кайсы программада ушул атка окшош ат сунушталат?*

5. *Турбо Паскаль иш майданындагы программаны ишке түшүрүү үчүн кайсы клавишалар басылат?*

Көнүгүүлөр

Паскаль программасынын текст редакторун ачып, төмөнкү тапшырмаларды аткар:

а) Өзбекстан Республикасынын гимнинин 1-төрт катарын киргиз;

б) жаңы терезе ач жана ага Республикабыздын гимнинин 2-төрт катарын киргиз;

в) 2-терезедеги тексттин (гимндин 2-төрт катары) үлгүсүн алып, 1-терезедеги гимндин 1-төрт катарынын уландысына жайгаштыр;

г) 1-терезедеги текстти «Гимн.txt» аты менен сакта;

д) 2-терезени сактабастан жап.

14-сабак. Паскаль программалоо тилинин алфавити жана түзүлүшү

Ар кандай программалоо тили сыяктуу Турбо Паскаль программалоо тили да өзүнүн алфавити жана синтаксис эрежелерине ээ. Турбо Паскаль тили ASCII коддуу белгилер жыйнагын өз ичине алат, мисалы:

Латын алфавитинин 26 баш жана кичине тамгалары: Aa, Bb, Cc, Dd, Ee, Ff, Gg, Hh, Ii, Jj, Kk, Ll, Mm, Nn, Oo, Pp, Qq, Rr, Ss, Tt, Uu, Vv, Ww, Xx, Yy, Zz (түшүндүрмөлөр жана тексттерди жазуу үчүн кирилл тамгаларын да колдоо мүмкүн);

Он араб саны: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Он алтылык сандар: 0 дөн 9 га чейинки араб цифралары жана A, B, C, D, E, F жана a, b, c, d, e, f тамгалары;

Атайын белгилер: "." (чекит), "," (үтүр), ":" (кош чекит); ";" (үтүрлүү чекит), "'" (апостроф), """ (тырмакча), "!" (илеп), "?" (суроо), "%" (пайыз), "$" (доллар), "@" (коммерсант белгиси), "&" (амперсанд), "#" (торчо), "^" (басым жасоо); түрдүү кашаалар: (,), {, }, [,]; жуп белгилер: :=, .. , (*, *), (., .).

Башкаруу белгилери: #0 дөн #31 ге чейин коддуу белгилер (# – белги кодунун ондуктагы маанисин билдирет, башкаруу белгилери иштегенде экранда көрүнбөйт).

Турбо Паскаль тилинде, негизинен, төмөнкү амалдар жана аларга мүнөздүү белгилер колдонулат:

Арифметикалык амалдар: "+" (кошуу), "–" (кемитүү),"*" (көбөйтүү), "/" (бөлүү);

Теңдештик амалдары: "=" (барабар), "<" (кичине), ">" (чоң); жуп белгилер: "<>" (барабар эмес), "<=" (чоң эмес), ">=" (кичине эмес);

Логикалык амалдар:

AND («ЖАНА» – логикалык көбөйтүү амалы)	**OR** («ЖЕ» – логикалык кошуу амалы
NOT («ЭМЕС» – логикалык тануу амалы)	**XOR** (окшоштукту тануу амалы)

Бардык программалоо тилдери сыяктуу Паскаль программалоо тили да өзүнүн жазуусу, мыйзам жана эрежелерине ээ болуп, алардын негизинде жогоруда келтирилген тамгалар, белгилер жана амалдардын жардамында көрсөтмө жана командалар түзүлөт. Ар бир көрсөтмө же команда ";"(үтүрлүү чекит) белгиси менен аякталат. Программанын текстинде бир сапка көбү менен 127 белги жазуу мүмкүн.

Көбүнчө, программа түшүнүктүү болушу үчүн ага түшүндүрмөлөр киритилет. Түшүндүрмөлөрдүн жардамында программа жана анын бөлүктөрү кандай милдетти аткаруусу мүнөздөлөт. Паскалда түшүндүрмө {жана} же (* жана *) көрүнүшүндөгү кашаалардын ичине жазылат. Мисалы, {бул түшүндүрмө мисал үчүн келтирилди} же (*түшүндүрмөнү ушундай жазуу мүмкүн*).

Адатта, Паскаль тилиндеги программалар **Program** атайын сөзү менен башталат. Бул сөздөн кийин программанын аты жазылат. Мисалы: **Program** квадраттык_теңдеме; {квадраттык теңдемени чыгаруу программасы.}

Программанын аты программанын милдетине тиешелүү болушу максатка ылайык болот. Бул зарыл программаны башка программалардын арасынан тез ажыратып алуу мүмкүнчүлүгүн берет. Дагы айтып өтчү нерсе, программанын аты программанын ишине эч кандай таасир көрсөтпөйт жана жалпысынан алганда, программага ат берүү да шарт эмес. Паскаль программалоо тилинде программа жазууда төмөнкүлөр колдонулат:

Константалар (туруктуу) – программа иштегенде мааниси өзгөрбөй турган чоңдуктар;

Өзгөрүүчүлөр – программа иштегенде мааниси өзгөрө турган чоңдуктар;

Туюнтмалар – тиешелүү амалдар менен байланышкан туруктуулар, өзгөрүүчүлөр жана функциялар;

Операторлор – программалоо тилинин аякталган кандайдыр бир амалын берүү үчүн ылайыкташтырылган көрсөтмөсү;

Функция жана процедуралар – өз атына ээ болгон өзүнчө программанын бөлүктөрү. Аларга негизги программадан кайрылуу жасалат;

Белгилер – программада башкаруу узатылып жаткан операторду көрсөтөт.

Паскаль тилинин команда жана көрсөтмөлөрү **модул** деп аталуучу **TPU** кеңейтилген атайын файлдарда жайгашкан. Аларга мисал катары **system** (систем), **crt** (sierti), **graph** (граф) модулдарын келтирүү мүмкүн. Модулдардын ар бири белгилүү бир багыттагы команда жана көрсөтмөлөрдү өз ичине алган. Мисалы, **system** модулу Паскалдын стандарттык (негизги) командаларын, **crt** модулу экран жана клавиатура менен иштөөгө (экранды тазалоо, экранда бир канча өзүнчө терезе пайда кылуу,...), ал эми **graph** модулу болсо, графикалык абалда иштөөгө ылайыкташтырылган команда жана көрсөтмөлөрдү өз ичине алат. Программада модулдар зарылдыкка карап колдонулат. Кандайдыр бир модулдун курамына кирген командадан пайдалануу үчүн программанын башында (аталыштан кийин) ушул жөнүндө көрсөтмө берилиши керек. Бул Паскалдын атайын сөзү **Uses** жардамында ишке ашырылат. Мисалы, программада графикадан пайдалануу үчүн, ага **Uses graph**; жазуусун киритүү керек. Эгерде программада бир канча модул колдонулса, алар өз ара үтүр менен бөлүп жазылат. Мисалы: **Uses crt, graph**;

Паскаль интегралдык чөйрөсү ишке түшүрүлгөндө **system** модулу автоматтык түрдө эстутумга жүктөлөт. Ошондуктан **Uses system**; жазуусу иштетилбейт. Көпчүлүк программалар үчүн **system** модулунун өзү жетиштүү болот.

Программа жазуудан мурда анда катыша турган чоңдуктарды аныктап алуу, өзгөрүүчүлөргө ат берүү жана аларды **мүнөздөө** (түрүн көрсөтүү) керек болот. Андан соң **программанын негизги бөлүгү** башталат, б.а. Паскалда программа эки бөлүктөн турат.

Паскаль тилиндеги программалар жалпы абалда төмөнкү түзүлүшкө ээ:

Program программанын аты; {милдеттүү эмес}
Uses {Модулдардын тизмеси}
Label {Белгилердин тизмеси}
Const {Константаларды мүнөздөө}
Var {Өзгөрүүчүлөрдү мүнөздөө}
Процедура жана функциялар
Begin

{Негизги бөлүк}
End.

Label, Const, Var, Begin, End – Паскаль тилинин атайын сөздөрү болуп, **label** – белги, const (constant – константа) – туруктуу маани, **va**r (variable) – өзгөрүүчүлөрдү мүнөздөө, **begin** – башталуу, **end** – аяктоо маанилерин билдирет.

Идентификаторлор дегенде туруктуулар, өзгөрүүчүлөр, процедуралар, функциялар, модулдар, программалардын аты түшүнүлөт. Идентификаторлор **стандарттык** жана **пайдалануучу** түрлөрүнө бөлүнөт. Стандарттык идентификаторлор - программа тарабынан алдын ала белгиленген болот.

Пайдалануучу идентификатору программист тарабынан тандалат жана ыктыярдуу узундукта болушу мүмкүн, бирок биринчи 63 белгиси **мааниге ээ** (айырмалантыруучу) болот. Идентификатор аты латын тамгасынан же ылдый сызык (_) белгисинен башталышы жана пробелдерсиз жазылышы шарт. Биринчи белгиден кийин тамгалар, сандар жана ылдый сызык белгиси жазылышы мүмкүн. Турбо Паскаль тилинде идентификатордун аттарынын, кайсы регистрде (төмөн же жогору) жазылышынын мааниси жок, б.а. **ага, Ага, аГа** сыяктуулар бир түрдүү **ат** деп каралат. Анткени, Турбо Паскаль транслятору программаны компиляция кылуу (программаны машина тилине которуу) учурунда идентификатор аттары жана кызматчы сөздөрдөгү бардык чоң тамгаларды кичине тамгаларга алмаштырып алат. Аттар апостроф ичине алынбайт, б.а. **‘Мен‘** жана **‘мен‘** ат боло албайт.

Паскаль программалоо тилинде төмөндө келтирилген сөздөр запасталган болуп, аларды пайдалануучу идентификатору катары колдонуу мүмкүн эмес:

and, asm, array, begin, case, const, constructor, destructor, div, do, downto, else, end, exports, file, for, function, goto, if, implementation, in, absolute, assembler, export, external, far, forward, index, interrupt, near, private, public, resident, virtual, inherited, inline, interface, label, library, mod, nil, not, object, of, or, packed, procedure, program, record, repeat, set, shl, shr, string, then, to, type, unit, until, uses, var, while, with, xor.

Паскаль программалоо тили бул сөздөрдү программада иштетүүгө жол койбойт жана ката кабарды экранда **«Error 2: Identifier expected»** (запас идентификатор) жазуусу аркылуу туюнтат.

Суроо жана тапшырмалар

1. Паскаль программалоо тилинин алфавити жөнүндө айтып бер.
2. Логикалык амалдарды чындык жадыбалынын жардамында түшүндүр.
3. Оператор деген эмне?
4. Программаны атоо жөнүндө айтып бер.
5. Идентификатор жөнүндө маалымат бер.
6. Паскалда программа кандай бөлүктөрдөн түзүлгөн болот?
7. Программанын мүнөздөө бөлүгү жөнүндө маалымат бер.

Көнүгүүлөр

1. Оң мамычадагы белгилерден сол мамычадагы тобуна ылайыктуусун аныкта.

Логикалык амалдар	**%, $, @ , & , (,), {, }, [,]**
Тендештик белгилери	NOT, OR
Атайын белгилер	<, <=, >, >=

2. Сол мамычадагы сөздөргө оң мамычадагы мүнөздөмөлөрдүн ылайыктуусун аныкта.

Модулдарды колдонуу	Program 9_класс
Белгилерди мүнөздөө	Var a21: integer;
Программаны атоо	Label 19;
Өзгөрүүчүнү мүнөздөө	Uses Crt;

3. Идентификатор аттарын «Туура жазылган» жана «Туура эмес жазылган» топторуна ажырат жана себебин түшүндүр.

a	1 күн	Менин Биринчи Программам	BMA
Чек ара#4	Кийинки жыл	Күн_21_июл_1963	and

15-сабак. Туруктуу жана өзгөрүүчү чоңдуктар

Паскаль тилинде, негизинен, үч түрдүү: **туруктуу, өзгөрүүчү** жана **жадыбал** түрүндөгү чоңдуктар иштетилет. Алар белгилүү, саптуу, логикалык жана сандуу түрдөгү маанилерди кабыл алышы мүмкүн.

Туруктуу чоңдуктар

Белгилүү туруктуулар апостроф ичине алынган бир белги – тамга, цифра же атайын белгиден турат. Мисалы: 'a'; 'B'; '9'; '-', ' ' жана башка.

Саптуу туруктуулар (белгилерден турган сап) саны 0 дөн 122 белгиге чейин болгон жана апостроф ичине алынган тамга, цифра жана атайын белгилердин удаалаштыгынан турат. Мисалы:

'Ташкент'; 'А 549'; 'В***М.'; '47%'; 'ВМА = '; '..-...-' ж.б.

Апостроф ичинде эч нерсе жазылбаса ал **башкы сап** деп аталат.

Логикалык туруктуулар True (чын) же False (жалган) логикалык маанилерден бири болуп саналат.

Сандуу туруктуулар эки түрдө – **бүтүн** же **чыныгы** болушу мүмкүн. **Бүтүн сандар** белгилүү же белгисиз көрүнүштө −2147483648 ден +2147483647 ге чейин болгон бүтүн сандар болуп эсептелет. Эгерде бүтүн сандуу туруктуу бул аралыктан чыгып кетсе, транслятор бул ката жөнүндө кабар берет. Чыныгы сандар өз кезегинде туруктуу чекиттүү жана өзгөрмөлүү чекиттүү сандарга бөлүнөт.

Ондук бөлчөктөрдүн бүтүн жана бөлчөк бөлүгүн бөлүүчү «үтүр»дүн ордуна Паскаль тилинде «чекит» жазылат.

Туруктуу чекиттүү сандар – ондук бөлчөк көрүнүшүндөгү сандар. Мисалы:

– 2.753; 283.45; 0.517; – 0.0013.

Өзгөрмөлүү чекиттүү сандар – экспоненциялдык көрүнүштө (E же e нин жардамында) туюнтулган сандар болот. Сандардын жазылышынын бул усулу өтө кичине же өтө чоң сандарды туюнтууда өтө ыңгайлуу.

Окулушу төмөнкүчө:

2.1E+07 – «2.1 көбөйтүрүлгөн ондун 7-даражасы»;

2.301e−63 – «2.301 көбөйтүлгөн ондун минус 63-даражасы».

Мисалы, 3400000000 = $3{,}4 \cdot 10^9$ саны Турбо Паскалда 3.4E+09 сыяктуу экспоненциалдык көрүнүштө жазылат. **E** тамгасынан мурда жазылган сан **мантисса, E** тамгасынан кийин жазылган сан болсо **тартип** деп аталат. Мантисса бүтүн же өзгөрмөлүү чекиттүү сан, ал эми тартип болсо бүтүн сан гана болушу мүмкүн. Мисалы: 37.3879 E−3= 0.0373879; 5.31 E+5= 531000; −0.075 E−5=−0.00000075; −2.37 E−4= −0.000237

Паскаль тилинде түзүлгөн программада **мүнөздөлгөн туруктуулар** катышышы мүмкүн. Мисалы,

Const A=21071963; _m10m10='2301'; Pi=3.141516;

Өзгөрүүчү чоңдуктар

Өзгөрүүчүлөр программанын мүнөздөө бөлүгүндө сөзсүз мүнөздөлүшү, б.а. алардын түрү көрсөтүлгөн болушу керек. Программада өзгөрүүчүлөрдү мүнөздөө Паскалдын **Var** кызматчы сөзү менен башталат:

Var өзгөрүүчү : түрү;
өзгөрүүчү : түрү;

Эгерде бир канча өзгөрүүчүнүн түрү бир түрдүү болсо, аларды өзүнчө мүнөздөбөстөн, чогуу мүнөздөө да мүмкүн:

Var 1-өзгөрүүчү, 2-өзгөрүүчү,..., n-өзгөрүүчү : түрү;

Бүтүн сандуу маанилер кабыл ала турган өзгөрүүчүлөр бүтүн сандуу өзгөрүүчүлөр деп аталат. Алар 5 түргө бөлүнүп, бири-биринен кабыл ала турган маанилеринин чек арасы жана компьютер эстутумунан ээлей турган жайы (көлөмү) менен айырмаланат. Төмөнкү жадыбалда бүтүн сандуу өзгөрүүчүлөрдү мүнөздөө үчүн зарыл болгон Паскалдын атайын сөздөрү, аларга мүнөздүү маанилердин чек арасы жана ээлей турган эстутумдун көлөмү келтирилген:

Түрү	Маанилердин чек арасы	Ээлей турган эстутумдун көлөмү
Byte	0 ...255	8 бит = 1 байт
ShortInt	−128 ...127	8 бит = 1 байт
Word	0 ...65 535	16 бит = 2 байт
Integer	−32 768 ...32 767	16 бит = 2 байт
LongInt	−2 147 483 648 ...2 147 483 647	32 бит = 4 байт

Мисалы: var i, j: Integer; bma: longint; mmr: Shortint; тартип_цифра: Byte; nat_0: word;

Бүтүн сандардын үстүндө **div** (бүтүн болуу) жана **mod** (калдык) амалдары орундуу. Мисалы:

25 div 4 = 6; 25 mod 4 = 1; 49 div 7 = 7; 49 mod 7 = 0.

Чыныгы сандуу маанилер кабыл ала турган өзгөрүүчүлөр **чыныгы сандуу өзгөрүүчүлөр** деп аталат. Алардын түрлөрү төмөнкү жадыбалда келтирилген.

Түрү	Маанилердин чек арасы	Разряды	Ээлей турган эстутум көлөмү
Real	$-2{,}9 \cdot 10^{39}$ ...$1{,}7 \cdot 10^{38}$	11—12	6 байт
Single	$-1{,}5 \cdot 10^{45}$...$3{,}4 \cdot 10^{38}$	7—8	4 байт
Double	$-5{,}0 \cdot 10^{324}$ ...$1{,}7 \cdot 10^{308}$	15—16	8 байт
Extended	$-3{,}4 \cdot 10^{4932}$ ...$1{,}1 \cdot 10^{4932}$	19—20	10 байт
Comp	$-9{,}2 \cdot 10^{18}$ ...$9{,}2 \cdot 10^{18}$	19—20	8 байт

Мисалы:

```
var     бурч, жаа_узундугу : Real; mab : extended;
        даражa : Single; куб : double; чыныгы : Comp;
```

Жадыбалдагы «Разряды» сандын анык цифраларынын санын билдирет. Өтө көп абалдарда реалдуу түрдүү өзгөрүүчүлөрдөн пайдалануу жетиштүү болот.

Белгилүү туруктуулардын маанисин кабыл алуучу өзгөрүүчүлөр **белгилүү өзгөрүүчүлөр** деп аталат. Алар Паскалдын **Char** атайын сөзүнүн жардамында мүнөздөлөт. Мисалы: var тамга, белги: char;

Саптуу өзгөрүүчүлөрдү мүнөздөө үчүн Паскалдын **String** атайын сөзү колдонулат. Мындай өзгөрүүчүлөр үчүн компьютер эстутумунан 255 байт (255 белги үчүн) жай ажыратылат. Эгерде саптуу өзгөрүүчү кабыл ала турган саптагы белгилердин саны программанын ишинде белгилүү бир чоңдуктан, мисалы, 10 белгиден ашпаса, компьютер эстутумун үнөмдөө максатында, аны String[10] аркылуу мүнөздөө максатка ылайык болот. Мисалы: Var катар:String; {катар аттуу өзгөрүүчүгө эстутумдан 255 байт ажыратылды}

_сап : String[24]; {_сап аттуу өзгөрүүчүгө эстутумдан 24 байт ажыратылат}

Логикалык туруктуу манилерди кабыл ала турган өзгөрүүчүлөр логикалык өзгөрүүчүлөр деп аталып, Паскалдын **Boolean** атайын сөзү аркылуу мүнөздөлөт. Мисалы:

```
var  натыйжа : Boolean;
     чоң, кичине : Boolean;
```

Паскаль тилинде түзүлгөн программада бир гана мүнөздөлгөн өзгөрүүчүлөр катышышы мүмкүн. Паскаль транслятору мүнөздөлбөгөн өзгөрүүчүлөрдү программада иштетүүгө жол койбойт жана ката кабарды экранга **«Error 3: Unknown identifier»** (белгисиз идентификатор, б.а. бул абалда белгисиз өзгөрүүчү) жазуусу аркылуу туюнтат. Өзгөрүүчүлөргө бир гана мүнөздөөдө көрсөтүлгөн түрдөгү маанилерди гана берүү мүмкүн болоорун эсте тутуу зарыл.

Суроо жана тапшырмалар

1. *Белгилүү туруктуу дегенде эмнени түшүнөсүң? Мисал келтир.*
2. *Саптуу туруктуулардын белгилүү туруктуулардан айырмасы эмнеде?*
3. *Сандуу туруктуулардын кандай түрлөрүн билесиң?*
4. *Логикалык туруктуулар кандай маанилерди кабыл алышы мүмкүн?*

5. *Өзгөрүүчүлөрдүн туруктуулардан айырмасы эмнеде?*
6. *Бүтүн сандуу өзгөрүүчүлөрдүн түрлөрүнө мисалдар келтир.*
7. *Чыныгы сандуу өзгөрүүчүлөрдүн түрлөрүнө мисалдар келтир.*
8. *Белгилүү өзгөрүүчүлөр кандай мүнөздөлөт? Мисалдар келтир.*
9. *Саптуу өзгөрүүчүлөр кандай мүнөздөлүшү мүмкүн? Мисалдар келтир.*

Көнүгүүлөр

1. Төмөнкү туруктуулардын түрлөрүн айтып бер.
   а) '7!'; 'информатика'; '−987378'; 'ВМА';
   б) ','; 'u'; '0'; ' ';
   в) 99; -200; 101; 87;
   г) 0.01; 8.909; 132.001; 878887.1;
   д) 0.07 E−3; −9.8 E6; е) True; False.
2. Төмөнкү өзгөрүүчүлөрдүн түрлөрүн аныкта жана түшүндүр.
   а) мен : Boolean; б) баатыр : String[7];
   в) жашоо : Real; г) сан : char;
   д) бакыт : Integer; е) ыр : Single;

16-сабак. Туруктуу жана өзгөрүүчү чоңдуктар темасын кайталоо

1. Төмөнкү туруктуулардын түрлөрүн айтып бер.
   а) −9.22 E−2; 0.01 E+5; 1.11 E−4;
   б) 21; 21; 7; 7; 19; 19; 63; 63;
   в) true; true; true; false; false; false;
   г) '555'; 'аар';'ммр';'ббж';'ага';
   д) 'Ыйык'; 'Мекен';'Эгемендүү';
   е) 'а'; 'д'; 'а'; 'м'; 'з'; 'а'; 'т';
2. Өзгөрүүчүлөргө ат берип, түрүнө карай мүнөздө.
   а) белгилүү; б) чыныгы; в) логикалык; г) саптуу;
   д) белгилери 7 ден ашпай турган саптуу.
3. Ар бир бөлүмдө бир өзгөрүүчүнүн бардык мааниси берилген же өзгөчөлүгү туюнтулган. Ушул өзгөрүүчүлөргө ат бер жана мүнөздө.
   а) −5; 0; 7; 58; −15; 9. б) 'Эл'; 'Мекен'; 'Ата'.
   в) 7.21; 4.2; 50.1902; −1.23. г) биринчи 7 чыныгы сан.
   д) true; true; false; true; false. е) '000'; '001'; '002'; '003'.
   ё) алфавит тамгалары.
   ж) 'Жогорку'; 'руханият', 'жеңилгис'; 'күч'.
4. Бардык мааниси берилген бүтүн өзгөрүүчүлөргө ат берип мүнөздө. Өзгөрүүчүнүн түрүн тандоодо эстутумдан аз жай алууга жетиш.
   а) −4; 0; −4; 8; 12; б) 1; 16; 256; 4096; 65536;
   в) 0; 2; 4; 6; 8; 10; г) 29350; −2; 8000; 250;
   д) 5; −32767; 46; 0; 32767; е) 200000; 2000; −20; 99999;

5*. Берилген амалдар натыйжасында пайда боло турган өзгөрүүчүлөрдү мүнөздө.

а) f: бүтүн; g: чыныгы; d:=f*g+f+g;
б) d: бүтүн; n: бүтүн; k:=d+2*n;
в) s: логикалык; e: логикалык; k:= not(s or e);
г) k: так; m: жуп; vvv:= k+m/2;

17-сабак. Жадыбал түрүндөгү чоңдуктар

Күндөлүк турмушта көп түрдөгү жадыбалдардан пайдаланылат: сабактар жадыбалы, шахмат же футбол оюндарынын өтүү жадыбалы, Пифагордун (көбөйтүү) жадыбалы, жөндөмөнүн жадыбалы жана башкалар. Жадыбалды түзүүчүлөр анын **элементтери** деп аталат. Жадыбал түрүндөгү чоңдуктар **бир өлчөмдүү** (сызыктуу), **эки өлчөмдүү** (туура төрт бурчтуу), **үч өлчөмдүү** (параллелопипеддүү) ж. б. болот. Биз бул окуу китебинин чегинде сызыктуу жана туура төрт бурчтуу жадыбалдарды көрүп чыгабыз.

Сызыктуу жадыбалдар сап же мамыча түрүндө туюнтулат. Мисалы, классындагы окуучулардын тизмеси класс журналында мамыча түрүндөгү жадыбал көрүнүшүндө жазылган. Окуучулардын аты-жөнү бул жадыбалдын элементтерин түзөт. Алардын ар бири өз тартип номерине ээ жана ар бир тартип номерине бир гана окуучунун фамилиясы туура келет.

Эки өлчөмдүү жадыбалдар мамычалар жана саптардан түзүлөт (текст процессору жана электрондук жадыбалдарга тиешелүү темаларды эсте). Алардын элементтери мамыча жана жолчолор кесишкен чакмактарда жайгашат. Мындай жадыбалдарда кандайдыр бир элементти көрсөтүү үчүн анын канчанчы жолчо жана канчанчы мамычада жайгашкандыгын, б.а. жолчо жана мамыча боюнча тартип номерлерин билүү керек болот. Демек, эки өлчөмдүү жадыбалдын ар бир элементине эки (жолчо жана мамыча боюнча) тартип номери туура келет.

Паскаль программалоо тилинде жадыбалдар менен иштөө үчүн **массив** түшүнүгү киритилген. **Массив** – жадыбал түрүндөгү чондук болуп, ал анык сандагы бир түрдүү жана иреттелген (б.а., катар номерине ээ) элементтер тобунан турат. Массив элементтеринин катар номери бүтүн сандарда туюнтулат, б.а. алар **терс** сан болушу да мүмкүн.

Паскалда ар бир массив өз атына ээ болуп, аларды атоо өзгөрүүчүлөрдү атоо сыяктуу болот. Мисалы: a5, сабак_жадыба-

лы, чыныгы_сандар. Массив элементтеринин тартип номери индекс деп аталат жана **индекс** квадраттык кашаанын ичинде жазылат. Мисалы, а[5] жазуусу – а аттуу массивдин бешинчи элементин билдирет, б.а. массивдин аты – а, индекси – 5.

1-мисал. А аттуу 7 элементтен турган сызыктуу жадыбалды пайда кыл.

Катар номери	1	2	3	4	5	6	7
Мааниси	Б	е	г	а	й	ы	м

Демек, жадыбалдын элементтери жана аларга мүнөздүү маанилер төмөнкүчө тиешелүүлүктө экен:

Жадыбалдын элементи	А[1]	А[2]	А[3]	А[4]	А[5]	А[6]	А[7]
Мааниси	Б	е	г	а	й	ы	м

Эки өлчөмдүү массив элементтери эки индекс аркылуу аныкталып, алар өз ара үтүр менен бөлүп жазылат жана биринчи индекс жолчонун тартип номерин, экинчи индекс мамычанын тартип номерин билдирет. Мисалы, S[4,3] жазуусу – S аттуу массивдин 4-жолчосу жана 3-мамычасы кесишкен чакмакта жайгашкан элементин билдирет.

2-мисал. S аттуу 4×5 (4 кө 5, б.а. 4 жолчолуу жана 5 мамычалуу) туура төрт бурчтуу жадыбалды сүрөттө (чакмакта көк түстө жадыбал элементтери жазылган).

Жолчо боюнча катар номери	Мамыча боюнча катар номери				
	2	3	4	5	6
1	3.2 S[1,2]	1.37 S[1,3]	−1.25 S[1,4]	7.12 S[1,5]	−11.4 S[1,6]
2	0.5 S[2,2]	1.1 S[2,3]	1.2 S[2,4]	−1,1 S[2,5]	4.22 S[2,6]
3	−0.1 S[3,2]	1.01 S[3,3]	71.2 S[3,4]	4.1 S[3,5]	−4.11 S[3,6]
4	6.3 S[4,2]	−7.01 S[4,3]	1.5 S[4,4]	7.5 S[4,5]	−1.09 S[4,6]

Жадыбалдан көрүнүп тургандай, мисалы, S[1,3]=1.37, S[2,2]=0.5, S[4,6]= −1.09.

Программада массивдер өзгөрүүчүлөр сыяктуу сөзсүз сүрөттөлүшү зарыл. Ал үчүн Паскаль программалоо тилинин **Array** кызматчы сөзү иштетилет. Бул сөздөн кийин квадраттык кашаанын ичинде массивдин биринчи жана акыркы элементтеринин тартип номерлери өз ара **эки чекит** (..) менен бөлүп жазылат. Уландысында Паскалдын **of** кызматчы сөзү жана андан кийин массив элементтеринин түрү жазылат. Мисалы:

```
var
```

A: array[1..7] of char; {1-мисалдагы элементтери 1 ден 7 ге чейин иреттелген char (белгилүү мааниси) түрдөгү **A** аттуу сызыктуу массив}

S: array[1..4,2..6] of real; {2-мисалдагы жолчолору 1 ден 4 кө чейин жана мамычалары 2 ден 6 га чейин иреттелген **real** (чыныгы маанидеги) **S** аттуу эки өлчөмдүү массив}

bma: array[−2..100] of integer; {−2 ден 100 гө чейин иреттелген **бүтүн** түрдөгү **bma** аттуу сызыктуу массив}

Демек, массив (жадыбал көрүнүшүндөгү чондук) дегенде, **жалпыз ат менен белгиленген бир түрдөгү иреттелген чондуктардын жыйындысы** түшүнүлөт.

3-мисал. Бир өлчөмдүү A жадыбал беш элементке ээ болсун:

Катар номери	−1	0	1	2	3
Мааниси	3	2	12	10	−8

Паскалда бул жадыбалдын элементтери төмөнкүчө туюнтулат:

A[−1] := 3; A[0] := 2; A[1] := 12; A[2] := 10; A[3] := −8;

Массив элементтер индексин кандайдыр бир бүтүн маанилүү өзгөрүүчү (мисалы, i) аркылуу туюнтуу мүмкүн, мисалы, эгерде i = 1 болсо, A[i] = −12, эгерде i = 3 болсо, A[i] = −8 болот.

4-мисал. Эки өлчөмдүү бүтүн маанилүү B массив берилген болсун:

$$B = \begin{bmatrix} 3 & 10 & 5 \\ 2 & 7 & 9 \end{bmatrix}.$$

Массив элементтерине өзүбүз тартип номерлерин берип B[0,0], B[0,1], B[0,2], B[1,0], ... сыяктуу жазып алабыз:

$$B = \begin{bmatrix} 3 & 10 & 5 \\ 2 & 7 & 9 \end{bmatrix} = \begin{bmatrix} B_{00} & B_{01} & B_{02} \\ B_{10} & B_{11} & B_{12} \end{bmatrix} = \left[B_{ij} \right],$$

мында i = 0, 1 жана j = 0, 1, 2 (*i* − жолчонун тартип номери, *j* − мамычанын тартип номери) маанилерин кабыл алат. Бул жадыбал Паскалда төмөнкүчө сүрөттөлөт:

```
var  b: array[0..1, 0..2] of Integer;
```

Эскертип өтөбүз, идентификатор атынын кайсы регистрде жазылышынын мааниси жок!

Жалпысынан индекс түрүндө өзгөрүүчү же туюнтма колдонулат. Мисалы, *I*=0, *J*=2 болсо, 4-мисалда B[I, J] = 5 жана (I+1=0+1=1 жана J−2=2−2=0 болгону үчүн) B[I+1, J−2] = 2 болот. Биз жадыбалдардын бир гана сызыктуу жана туура төрт бурчтуу түрлөрү менен таныштык. Чындыгында Паскаль тилинде көп өлчөмдүү (255 ке чейин) жадыбал түрүндөгү чондуктардан да пайдалануу мүмкүн. Мындай жадыбалдарды мүнөздөөгө бир канча мисал келтиребиз.

1) var s: array[1..4, 1..7, 0..10] of Byte; {s − Byte түрдүү 3 өлчөмдүү жадыбал};

2) var t, k: array [1..100, 1..80, 1..50] of string; {t жана k – 3 өлчөмдүү жолчолуу жадыбалдар};

3) var f: array [–5..10, 0..10, 2..10] of char; { f – 3 өлчөмдүү белгилүү жадыбал}.

Паскаль программалоо тилинде мүнөздөлгөн массивдер үчүн эстутумдан жай запастап коюлат. Ошол себептүү эстутумдун керегинен ашык бөлүгүн ээлебестик үчүн массив түрүнөн сырткары канча элементтен тургандыгын билүү максатка ылайык болот. Жалпысынан, сызыктуу K дан S ке чейин тартиптелген массив элементтеринин саны **S–K+1** өө, эки өлчөмдүү жолчолору B дан M га чейин жана мамычалары A дан G га чейин тартиптелген массив элементтеринин саны **(M–B+1) · (G–A+1)** болот.

Мисалы, 3-мисалдагы –1 ден 3 кө чейин тартиптелген A массивде (3 – (–1)+1=3+1+1=) 5 бүтүн маанилүү элемент, 4-мисалдагы жолчолору 0 дөн 1 ге чейин жана мамычалары 0 дөн 2 ге чейин тартиптелген B массивде ((1–0+1)·(2–0+1)=2·3=) 6 бүтүн маанилүү элемент бар.

Суроо жана тапшырмалар

1. *Турмушта кездеше турган жадыбал түрүндөгү чоңдуктарга мисалдар келтир.*
2. *Сызыктуу массив кандай өлчөмдөрдө болот?*
3. *Массивде индекс эмне үчүн зарыл?*
4. *Массив элементтеринин индекстери кандай маанилер кабыл алышы мүмкүн?*
5. *Жадыбал түрүндөгү чоңдуктардын түрлөрүн кандай бөлүү мүмкүн?*

Көнүгүүлөр

1. Төмөндө келтирилген удаалаштыктар кандай өлчөмдүү массивдердин туюнтулушун жана канча элементтен тургандыгын аныкта.
   а) A[0], A[1], A[2], A[3], ... , A[99];
   б) B[0,0], B[0,1], B[0,2], ... , B[3,5];
   в) M[0,0,0], M[0,0,1], ... , M[1,1,1];
   г) G[– 22,3], G[– 22,4], G[– 22,5],..., G[– 20,5].
2. Бүтүн түрдөгү, сызыктуу 100 элементтүү жадыбал кайсы жоопто туура мүнөздөлгөн?
   а) var B: array [1..100] of real;
   б) var M: array [1..100] of char;
   в) var A: array [0..99] of string;
   г) var G: array [5..104] of integer;
3. 2-көнүгүүдөгү массивдердин түрүн, өлчөмүн жана элементтеринин санын аныкта.
4. Бирден бир өлчөмдүү бүтүн жана белгилүү жадыбалдарды жана эки өлчөмдүү чыныгы түрдөгү жадыбалды мүнөздө.

18-сабак. Жадыбал түрүндөгү чоңдуктар темасын кайталоо

1. Туура төрт бурчтуу чыныгы түрдөгү сегиз жолчолуу жана он бир мамычалуу F жадыбалдын кандай мүнөздөлүшүн аныкта.

а) var A: array [8..11] of real;
б) var B: array [1..8,1..11] of integer;
в) var D: array [8..11,8..11] of real;
г) var M: array [0..8,0..10] of integer;
д) var F: array [0..7,0..10] of real;
е) var F: array [0..7,0..10] of char;

2. 1-көнүгүүдө мүнөздөлгөн массивдердин түрүн, өлчөмүн жана элементтеринин санын аныкта.

3. Пифагордун жадыбалын түз. Жадыбалдын элементтерин иликте. Массивге ат берип мүнөздө.

4. Үй-бүлө мүчөлөрүндүн аты, туулган жылы жана маалыматы жөнүндө жадыбал түз. Массивди мүнөздө жана элементтерин чечмеле.

5. Төмөнкү **integer** түрүндөгү M сызыктуу массив элементтерине бош чакмакта түрүнө ылайыктуу маани бер. Массивди мүнөздө.

M[−7]	M[−6]	M[−5]	M[−4]	M[−3]	M[−2]	M[−1]

6. Төмөнкү **char** түрдөгү B сызыктуу массив элементтерине бош чакмакта түрүнө ылайыктуу маани бер. Массивди мүнөздө жана элементтерин мамыча түрүндө жаз. Массивди эки өлчөмдүү массив түрүнө өткөрүп кайра мүнөздө.

B[9]	B[10]	B[11]	B[12]	B[13]	B[14]	B[15]	B[16]

7. Төмөнкү **string** түрдөгү A массив элементтерине бош чакмакта түрүнө ылайыктуу маани бер. Жадыбалды мүнөздө. Массивге башка тартип номерлери берип, G ат менен кайра мүнөздө.

19-сабак. Стандарттык функциялар жана процедуралар, алгебралык туюнтмалар

Функция түшүнүгү математика предметинен маалым. Функциялар өзгөчөлүгүнө карай түрлөргө бөлүнгөн. Мисалы, сызыктуу, квадраттык, тригонометриялык ж.б. Ушундай функциялардын кээ бирлеринен Паскаль программалоо тилинде да пайдаланылат. Паскалда функциялардан пайдалануу ыңгайлуу болушу үчүн кээ бир функциялар программа трансляторунун камсыз-

доосуна киритилген. Программа транслятору камсыздоосуна киритилген функциялар **стандарттык функциялар** деп аталат. Бул функциялардын көпчүлүгү сага MS Excel программасы аркылуу тааныш. Ошону менен бирге Паскаль тилинде так бир амалдарды аткарууга арналган **стандарттык процедуралар** да иштетилет.

Төмөнкү жадыбалда Паскалдын кээ бир стандарттык функцияларынын түшүндүрмөсү менен келтирилген:

Функция-нын аты	**Аргументтин түрү**	**Сандык маани-синин түрү**	**Түшүндүрмө**
Математикалык функциялар			
Abs(x)	бүтүн/чыныгы	бүтүн/ чыныгы	**x** тин абсолюттук мааниси (модулу): **|x|**
Sin(x)	бүтүн/чыныгы	чыныгы	**x** тин синусу (радиан) : **sinx**
Cos(x)	бүтүн/чыныгы	чыныгы	**x** тин косинусу (радиан): **cosx**
Arctan(x)	бүтүн/чыныгы	чыныгы	**x** тин арктангенси **arctgx**
Sqrt(x)	бүтүн/чыныгы	чыныгы	**x** тин квадраттык тамыры **(x≥0):** $\sqrt{x}$
Sqr(x)	бүтүн/чыныгы	бүтүн/ чыныгы	**x** тин квадраты: x^2
Exp(x)	бүтүн/чыныгы	чыныгы	e^x (e = 2.718282...)
Ln(x)	бүтүн/чыныгы	чыныгы	**x** тин натуралдык логарифми **(x > 0): ln x**
Frac(x)	бүтүн/чыныгы	чыныгы	**x** тин бөлчөк бөлүгү: **{x}**
Int(x)	бүтүн/чыныгы	чыныгы	**x** тин бүтүн бөлүгү: **[x]**
Random	—	чыныгы	**[0, 1)** аралыгындагы кокус сан
Random(x)	Word	Word	**[0, x)** аралыгындагы кокус сан
Өзгөрүүчүлөрдүн түрүн өзгөртүүчү функциялар			
Trunc (x)	чыныгы	LongInt	**x** тин бүтүн бөлүгү
Round (x)	чыныгы	LongInt	**x** ти бүтүнгө чейин тегеректейт
Odd (x)	бүтүн	логика-лык	**x** так сан болсо «чыныгы» маани алынат
Chr (x)	Byte	Char	**x** тин ондук ASCII кодуна ылайык белги
Ord ('m')	Char	Byte	**'m'** белгинин ондук ASCII коду

Математикалык процедуралар			
Inc (x)	бүтүн	бүтүн	**x** тин маанисин 1 ге арттырат **(x:=x+1)**
Dec (x)	бүтүн	бүтүн	**x** тин маанисин 1 ге азайтат **(x:=x−1)**

1-мисал. Кээ бир функциялардын колдоонуу:

Функция	Сандык мааниси	Функция	Сандык мааниси	Функция	Сандык мааниси
abs(−5)	5	abs(−4.9)	4.9000000000e+00	abs(4.9)	4.9000000000e+00
sqr(4)	16	sqr(2.5)	6.2500000000e+00	Sqrt(16)	4.0000000000e+00
sqr(−4)	16	Sqr(0.0)	0.0000000000e+00	Sqrt (0.16)	4.0000000000e−01
sqr(0)	0	Sin(0)	0.0000000000e+00	Sin(1)	8.4147098481e−01
trunc(5.3)	5	Int(5.3)	5.0000000000e+00	Int(5)	5.0000000000e+00
trunc(−5.3)	−5	Int(−5.3)	−5.0000000000e+00	frac(5.3)	3.0000000000e−01
Round(5.49)	5	frac(−5.3)	−3.0000000000e−01	frac(5)	0.0000000000e+00
Round(5.5)	6	Odd(5)	TRUE	Odd(−5)	TRUE
Round(−5.49)	−5	Odd(4)	FALSE	Odd(−4)	FALSE
Round(−5.5)	−6	Odd(0)	FALSE	Chr(65)	'A'
Chr(97)	'a'	Ord('A')	65	Ord('a')	97

Математикалык формулаларда көп иштетиле турган π санын туюнтуу үчүн Паскалда атайын Pi туруктуу (константа) ажыратылган (Pi=3.1415...).

Паскаль программалоо тилинде **алгебралык туюнтмалар** арифметикалык амалдардын жардамында байланышкан туруктуулар, өзгөрүүчүлөр жана функциялардан түзүлөт. Алгебралык туюнтмалар бир жолчодо жазылат, б.а. жолчодон ылдыйга түшүрүп же жогоруга көтөрүп жазуу мүмкүн эмес. Мисалы, $3ab^2$ туюнтма Паскал тилинде **3*a*sqr(b)** же **3*a*b*b** сыяктуу,

$\frac{a}{b^2}$ туюнтма **a/sqr(b) же a/(b*b)** сыяктуу жазылат.

Туюнтмаларды жазууда амалдарды аткаруу тартибин көрсөтүү үчүн жөнөкөй кашаалар гана колдонулат. Кашаанын ичиндеги амалдарды аткаруу солдон оңго карап, математикада кабыл алынган тартипти сактаган түрдө удаалаш аткарылат:

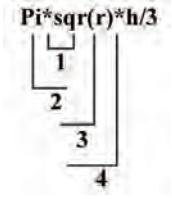
- функциялардын маанилери эсептелет;
- көбөйтүү же бөлүү амалы аткарылат;
- кошуу же кемитүү амалы аткарылат.

Мисалы, $\frac{a+b}{c}$ арифметикалык туюнтманы Паскалдагы жазылышы **(a+b)/c** сыяктуу болуп, аны эсептөөдө баштап кашаанын ичиндеги амал, б.а. *a+b* аткарылат, андан соң натыйжа *c* га бөлүнөт. Амалдарды аткаруу тартиби да кашаалардын жардамында тартиптелет: $\sqrt{a^2 - b^2}$ туюнтманы Паскалда **sqrt (sqr**(*a*) − **sqr**(*b*)) сыяктуу, |x + tgx| туюнтма **abs(x + tan(x))** түрүндө жазылат.

2-мисал. *R* жана *H* өзгөрүүчүлөрдүн белгилүү бир маанилеринде туюнтманын мааниси эсептелсин:

$$\frac{1}{3}\pi R^2 H.$$

Бул туюнтма Паскалда **Pi*sqr(r)*h/3** сыяктуу жазылат. Мында амалдар төмөнкү тартипте аткарылат:



1. **sqr(r)**
2. **Pi*sqr(r)**
3. **Pi*sqr(r)*h**
4. **Pi*sqr(r)*h/3**

Албетте, эки арифметикалык амал удаалаш келгенде туюнтманы кашаа менен жазуу мүмкүн. Мисалы: 5*(−1) же a+(−b).

Кээ бир абалдарда Паскаль программалоо тилинде жазылган туюнтманы адаттагы математикалык көрүнүштө жазуу талап кылынат. Мисалы, Паскаль программалоо тилинде жазылган **0.5*(sin(x)+cos(x))** туюнтма математикалык көрүнүштө төмөнкүчө болот:

$$\frac{1}{2}(sinx + cosx).$$

Паскалдын стандарттык функциялары бардык математикалык амалдарды өз ичине албаган. Ушул себептен кээ бир математикалык амалдарды Паскалдын бир канча стандарттык функциясы аркылуу же бир эле стандарттык функцияны бир нече жолу колдонуу аркылуу туюнтууга туура келет. Мисалы, Паскалда санды каалагандай даражага көтөрүү функциясы жок. Ошондуктан a^3 туюнтманы Паскалда **a*a*a** же **sqr(a)*a** сыяктуу, a^4 туюнтманы болсо **a*a*a*a** же **sqr(sqr(a))** же **sqr(a)*sqr(a)** сыяктуу жазуу мүмкүн.

Жалпысынан, a^b ($a>0$) түрүндөгү туюнтма үчүн математикада $a^b = e^{b \cdot \ln a}$ формула орундуу. Ошондуктан Паскалда a^b ($a>0$) туюнтма **exp(b*Ln(a))** түрүндө жазылат.

3-мисал. $\frac{x-y}{x^5 - y^3}$ алгебралык туюнтманы Паскалдагы көрүнүшүн жаз.

Чыгаруу. Бул туюнтманы Паскалда бир канча түрдүү усулда мүнөздөө мүмкүн. Ушулардан бири төмөнкүчө:

(x−y)/(exp(5*ln(x))−sqr(y)*y).

Паскаль программалоо тилиндеги стандарттык функциялардын аргументи дайыма кашаанын ичинде жазылаарын эсте тут!

Суроо жана тапшырмалар

1. *Кандай функциялар стандарттык функция деп аталат?*
2. *Стандарттык функциялардын адаттагы жана Паскалда жазылышынын кандай айырмасы бар?*
3. *Алгебралык туюнтмалар эмнелерден турат?*
4. *Кандайдыр бир туюнтмада бир түрдүү амалдар катышса, алардын аткарылуу тартиби кандай болот?*
5. *Амалдарды аткаруу тартибин өзгөртүү үчүн эмнелерден пайдаланылат?*
6. *Trunc(4.7)=Round(4.7) орундуубу? Жообуңду түшүндүр.*
7. *sin x−c түрүндөгү жазуу Паскалда эмне үчүн ката эсептелет?*
8. *2*−v түрүндөгү жазуу Паскалда туура жазылганбы? Жообуңду түшүндүр.*
9. *sqr(abs(x +sin(x))−pi) туюнтмада амалдарды аткарылуу тартибин түшүндүр.*

Көнүгүүлөр

1. Теманын 1-мисалындагы өзгөрмөлүү чекиттүү сандарды туруктуу чекиттүү сандарга өткөр.

2. Төмөнкү алгебралык туюнтмаларды Паскаль программалоо тилинде жаз.

а) $ax+b$; б) $xyz-23$; в) ax^2+bx+c;

г) $a^4x^3-(1-y^2)^2$; д) $\frac{a+5}{2b}$; е) $8(a+b^2c)$.

3. Төмөнкү туюнтмаларды Паскаль программалоо тилинде жаз.

а) $25^{20}+|1-y^2|$; б) [5m]+{100b}; в) xsina+ycosb−5^2;

г) sinsin$x$+cossin$y$; д) $21-\sqrt{2011-b^2}$.

4. Паскалда жазылган төмөнкү туюнтмалардын арасынан туура эмес жазылганын тап.

а) 2*a+b; б) sqr(x*b^2); в) sin(−3*x);
г) sin((a+b+cos(x)); д) 2*(−b)+a2.

5. Паскалда жазылган төмөнкү туюнтмаларды жөнөкөй жазуу түрүнө өткөр.

а) a* (Sqr(x)+1);
б) sin (x*x*x−sqr(sqr(x))+5);
в) pi*h*(sqr(r1) + sqr(r2) + r1*r2)/3.

20-сабак. Стандарттык функциялар жана процедаралар, алгебралык туюнтмалар темасын кайталоо

1. Төмөнкү туюнтмаларды Паскаль программалоо тилинде жаз.

а) $\frac{x-y}{x^2-y^3}$;

б) $\frac{x+y}{xyz}+\sin^2 x$;

в) $(5a^2+2x)+\frac{3x}{a^3}+tg^5a^3$;

г) $\cos^3\sin^2 x+\cos a^5$;

д) $\sqrt{5+x}-\sqrt{z}\,\frac{3x}{a^3}+\sqrt[7]{a}$.

2. Төмөндө Паскалда жазылган туюнтмалардын маанисин эсепте.

а) sqr(trunc(4.95));
б) trunc(int(4.95)+0.7);
в) round(trunc(3.5)+0.7);
г) 3+frac(12.5);
д) sqrt(sqr(16));
е) sqrt(sqrt(256)+9);
ё) sqr(5−abs(−5));
ж) abs(−sqrt(6)).

3. a = 5, b = 4 болсо, төмөндө Паскалда жазылган туюнтмалардын манисин эсепте.

а) abs(a+b−a*b);
б) sqr(a+b−a*b)−110;
в) round(a/b+0.3)+9;
г) 3+frac(b/a);
д) sqrt(sqr(a)−b*b);
е) sqrt(sqrt(a+b)+6);
ё) sqr(a−abs(b−a));
ж) abs(9−sqrt(a*b+a)).

4. Төмөндө Паскалда жазылган туюнтмалардын мааниси кандай түрдөгү туруктуу болушун аныкта.

а) abs(−sqrt(2011));
б) abs(sqr(2))+19;
в) frac(abs(−20));
г) int(1.9)*trunc(0.2).

21-сабак. Өздөштүрүү жана маалыматтарды экранга чыгаруу оператору

Паскаль программалоо тили, адатта, мүнөздөлгөн өзгөрүүчүлөр үчүн эстутумдан жай бөлүп, алардын түрүнө ылайык башталгыч маанилерин жазып коёт:

Өзгөрүүчүнүн түрү	Башталгыч маани	Өзгөрүүчүнүн түрү	Башталгыч маани
бардык бүтүн сандуу	0	бардык чыныгы сандуу	0.0000000000e+00
char	" (пробел)	boolean	FALSE
string	" (бош жолчо)	string[7]	" (бош жолчо)

Өздөштүрүү оператору. Өздөштүрүү оператору өзгөрүүчүлөргө маани берүү үчүн колдонулат. Ал := белги аркылуу туюнтулат. Өздөштүрүү операторунун жалпы көрүнүшү төмөнкүчө:

өзгөрүүчү := туюнтма;

Бул оператор аткарылганда төмөнкүчө иштер аткарылат:

1) туюнтманын мааниси эсептелет;

2) туюнтманын мааниси өзгөрүүчүгө өздөштүрүлөт, б.а. эстутумдун өзгөрүүчү үчүн ажыратылган бөлүгүнө (өзгөрүүчүнүн «эски» мааниси өчүп кетет) туюнтманын мааниси жазылат.

1-мисал. Төмөнкү программа аткарылышы натыйжасында **a** аттуу өзгөрүүчүнүн мааниси **22** санына барабар болот.

```
var  a: integer;
begin
   a := 22;
End.
```

2-мисал. Төмөнкү программанын аткарылышы натыйжасында **мөмө** аттуу жолчолуу өзгөрүүчүнүн мааниси **«алма»** сөзүнө барабар болот.

```
var  мөмө : string;
begin
   мөмө := 'алма';
End.
```

3-мисал. Бул мисалда **a** жана **b** өзгөрүүчүлөрдүн мааниси кандай өзгөрүшү даана көрүнөт.

```
   var      a,b,m: integer;
   begin
a := 8;          {a нын мааниси 8 ге барабар болду}
b := a*5;        {b нын мааниси a*5=8*5= 40 ка барабар болду}
b := b+10;  {эми b нын мааниси b+10= 40+10= 50 гө барабар болот}
m:=m*b     {m дин башталгыч мааниси берилбегендиктен 0 деп
              алынат, демек m дин мааниси 0*50 = 0 гө барабар
            болот}
End.
```

Жогорудагы мисалдарда өзгөрүүчүлөр түрдүү маанилерди өздөштүрдү. Бирок биз алардын натыйжасын көрбөдүк. Анткени алар компьютердин эстутумунда калып, экранга чыгарылбайт. Маалыматтарды компьютер экранына чыгаруу үчүн **чыгаруу операторунан** пайдаланылат. Паскалда чыгаруу оператору төмөнкү эки түрдүү көрүнүшкө ээ:

Write(чыгаруу тизмеси) жана **WriteLn(чыгаруу тизмеси)**

бул жерде Write (англ. – жазуу) жана **WriteLn** Паскалдын кызматчы сөздөрү; чыгаруу тизмеси – өз ара үтүр менен бөлүнгөн жана экранга чыгарылышы керек болгон туюнтма, өзгөрүүчү же туруктуулардын удаалаштыгы. Чыгаруу тизмесинде туюнтма катышса, оболу туюнтма эсептелип, пайда болгон натыйжа экранга чыгарылат. Чыгаруу тизмесиндеги туруктуулар белгилүү же жолчолуу болсо, албетте апостроф ичине алынышы шарт.

Write жана **WriteLn** операторлорунун айырмасы, **Write** оператору жардамында маалыматтар экранга чыгарылгандан соң жүргүч экрандын ушул

жолчосунда калат жана экранга чыгарыла турган кийинки маалыматтар ушул жолчого жүргүч турган жайдан баштап чыгарылышында. Ал эми **WriteLn** операторунда болсо маалыматтар экранга чыгарылгандан соң, жүргүч кийинки жолчонун башына өтөт.

4-мисал.

```
begin
 write('Гүлдөп, '); write('жайна ');
   write('Ата Мекеним!');
End.
```

Программа аткарылгандан соң, компьютер экранында

Гүлдөп, жайна Ата Мекеним!

жазуусу пайда болот.

5-мисал.

```
begin
writeln('Гүлдөп, ');
writeln('жайна ');
 write('Ата Мекеним!');
End.
```

Программа аткарылгандан соң, компьютер экранында

Гүлдөп,
жайна
Ата Мекеним!

жазуусу пайда болот.

6-мисал.

```
программа өздөштүрүү_чыгаруу;
var a,b:integer;
begin   a:=23; b:=a+21;
write('b нын мааниси ', b, ' га
барабар');
End.
```

Программа аткарылгандан соң, компьютер экранында

b нын мааниси 44 кө барабар

жазуусу пайда болот.

Маалыматтарды чыгарууда **чыгаруу форматын** көрсөтүү мүмкүн. Чыгаруу форматы чыгарылып жаткан маалыматтардын көрүнүшү (форматы)н белгилейт. Ал үчүн чыгарылып жаткан өзгөрүүчүдөн кийин «:» (эки чекит) белгиси коюлат. Мисалы, a – чыныгы түрдөгү өзгөрүүчү болсо, чыгаруу форматында эки параметр – чыгарылып жаткан санга бөлүнгөн бөлмөлөр саны көрсөтүлөт. Мисалы, **WriteLn(a:10:2)**; оператору a нын маанисин чыгаруу үчүн 10 бөлмө ажыратат, ошондон бир бөлмөсү чекит жана эки бөлмөсү бөлчөктүн бөлүгү үчүн ажыратылат. Эгерде сан бүтүн болсо, чыгаруу форматында бир параметр – чыгарылып жаткан санга бөлүнгөн бөлмөлөрдүн саны көрсөтүлөт. Мисалы, **WriteLn(b:6);**. Жолчолуу жана белгилүү өзгөрүүчүлөр үчүн чыгаруу форматы алардын маанисин чыгаруу үчүн бөлүнгөн жайдын (аянт) узундугун аныктайт.

Чыгарылып жаткан сан же текст ага бөлүнгөн жайдын оң чек арасы боюнча тегизделип чыгат. Мисалы, a: = 3.24; болсо, WriteLn('a = ',a:6:2); оператору экранга $a = 3.24$ түрүндөгү жазууну чыгарат (= белгисинен кийин эки бош жай (пробел) калат).

Чыгаруу форматында бөлүнгөн жай өзгөрүүчүнүн маанисинин «узундугу»нан кичине болсо, чыгаруу форматы бекер кылынат жана өзгөрүүчүнүн

мааниси толук бойдон экранга чыгарылат. Бир гана чыныгы сан бөлчөк бөлүгүнүн форматы бекер кылынбайт. Чыныгы санды чыгарууда форматы көрсөтүлбөсө, ал экранга экспоненсиалдык көрүнүштө чыгарылат.

7-мисал.

```
var a,b : real;
Begin     a:=3.24; b:=5;
 writeln('a=',a); writeln('b=',b);
end.
```

Компьютер экранында

```
a=3.2400000000E+00
b=5.0000000000E+00
```

8-мисал.

```
var a,b : real;
Begin   a:=3.24; b:=5.3;
writeln('a=', a:6:2);
  writeln('b=',b:1:0);
end.
```

Компьютер экранында

```
a=3.24
b=5
```

Келтирилген эки мисалда экранга чыгарылган *a* жана *b* өзгөрүүчүлөрдүн маанилери бир түрдүү, бирок алардын көрүнүшүндө чоң айырма бар. 8-мисалдагы экранга чыгарылган маалымат албетте 7-мисалдагыга караганда так жана түшүнүктүү көрүнүшкө ээ. Сандын бөлчөк бөлүгүндөгү керектүү цифралардын санын так билбеген абалдарда экранга туура эмес натыйжа чыгарбастык үчүн чыгаруу форматынан этияттык менен пайдалануу зарыл.

Эсте тут: экрандагы натыйжаны көрүү үчүн **ALT+F5** клавишалар жуптугу басылат.

Суроо жана тапшырмалар

1. *Өздөштүрүү оператору кандай милдетти аткарат?*
2. *Өздөштүрүү операторунун жалпы көрүнүшүн мисалдардын жардамында түшүндүр.*
3. *Маалыматтарды экранга чыгаруу операторунун жалпы көрүнүшү кандай?*
4. *Write жана Writeln оператору айырмасын мисалдардын жардамында түшүндүр.*
5. *Маалыматтарды экранга чыгаруу операторунун мүмкүнчүлүктөрүн мисалдар менен түшүндүр.*
6. *Маалыматтарды чыгаруу операторунда апостроф ичиндеги жазуулар эмнени билдирет?*
7. *Чыгаруу форматы эмне жана ал эмне үчүн колдонулат?*

Көнүгүүлөр

1. Төмөнкү туюнтмаларды өздөштүрүү операторунун жардамында жаз.

а) a= 48; b= 51; б) x= 0; a= 3,6x+sinx;

в) g= 4; g= g+16; г) a= 9,81; m= 50; F= m a;

д) x= 1; y= $\frac{x-63}{21-7x}$; е) z= 25; z= $\sqrt{z}$.

2. Чыгаруу операторлору аткарылгандан соң натыйжа экранда кандай чагылдырылышын жаз.

а) write('a='); write(2+3);
write('='); write('2+3');
б) writeln('a='); write(2+3);
write('='); writeln('2+3');
в) write('a='); writeln(2+3);
write('='); writeln('2+3');
г) write('a='); write(5);
writeln('='); write('2+3');
д) writeln('a='); writeln(5);
write('='); write('2+3');
е) write('a='); writeln(5);
writeln('='); write('2+3');

3. Паскалда жазылган төмөнкү программа үзүндүлөрүндөгү бардык өзгөрүүчүлөрдүн аралык маанисин жана экранга чыга турган натыйжаны аныкта.

а) a:= −cos(pi)−sin(pi/2); x:= x*x + a;
writeln('a=',a, 'x=',x);
б) a:= 'Мен '; g:= 'эгемендүү ';
b:= 'Өзбекстандын'; m:= ' перзентимин!'
write(a, g, b, m);
в) a:= 9; b:=a+a; a:= a*a−b;
write('a= ', a); write(' b=', b);

22-сабак. Өздөштүрүү жана маалыматтарды экранга чыгаруу оператору темасын кайталоо

1. Төмөнкү туюнтмаларды өздөштүрүү операторунун жардамында жаз.

а) $y = \frac{x-21}{7-x^{63}}$;

б) $a = 3{,}6x + sin\ x$;

в) $z = \sqrt{x-5y+xtgx}$; г) $S = \pi r^2$;

д) $F = ma$; е) $S = \frac{ah}{2}$.

2. Чыгаруу операторунун натыйжасы кандай болушун аныкта.

а) a:=123.45;
write('a=', a:2:1);
б) a:=123.45;
write('a=', a:5:1);

в) a:= '2011';
writeln(a:3, ' жыл':3);

г) a:= '2011';
writeln(a:4, ' жыл':5).

3. Программадагы өзгөрүүчүлөрдүн мааниси түрүнө ылайык болушу үчүн суроо белгиси ордуна зарыл стандарттык функцияны жаз жана экранга чыга турган натыйжаны аныкта.

```
а) var a, b, c: integer;
   begin a:=25; b:= ?(sqrt(a)); c:= ?(a/b);
         writeln(a, ' ', b, 'c= ', c);
   End.
б) var x, y, z: word;
   begin x:=?(?(−7.21)); y:= ?(sqrt(x*x));
   z:= ?(x+y−100); write(z−x, y);
   End.
```

23-сабак. Маалыматтарды эстутумга байланыш усулунда киргизүү оператору

Паскалда өзгөрүүчүлөргө маани берүүнүн өздөштүрүү операторунан пайдаланууудан башка усулдары да бар. Алардан бири **маалыматтарды эстутумга байланыш усулунда киргизүү** деп аталат жана **киргизүү оператору** жардамында ишке ашырылат. Киритүү операторунан өзгөрүүчүлөргө программанын аткарылышы учурунда компьютер клавиатурасынан маани берүү үчүн колдонулат. Киргизүү операторы төмөнкү эки түрдүү көрүнүшкө ээ:

Read (киргизүү тизмеси); жана **ReadLn(киргизүү тизмеси);**

бул жерде **Read** (read (англ.) − окуу) жана **ReadLn** Паскалдын кызматчы сөздөрү, **киргизүү тизмеси** болсо бир өзгөрүүчү же өз ара үтүр менен бөлүнгөн бир канча өзгөрүүчүлөрдүн удаалаштыгы. Мисалы: Read(a); Read(alfa,betta); ReadLn(_name);.

Киргизүү оператору программанын иштешин токтотот жана тизмедеги өзгөрүүчүлөргө клавиатура аркылуу маани берилишин күтөт. Эгерде тизмеде бир канча өзгөрүүчү болсо, алардын маанилери өз ара пробел (бош жай) менен бөлүп киритилиши же ENTER клавишин басып киритилиши мүмкүн. Эки абалда да акыркы өзгөрүүчүнүн мааниси киритилгенден соң ENTER клавишин басуу шарт.

Read жана **ReadLn** операторлорунун айырмасы төмөнкүчө: бир же бир канча өзүнчө жазылган **Read** оператору жардамында киритиле турган өзгөрүүчүлөрдүн мааниси бир жолчо-

до пробелдер менен бөлүп киритилиши мүмкүн. ReadLn оператору болсо бир гана өзүнүн тизмесинде келтирилген өзгөрүүчүлөрдүн гана маанисин бир жолчодо пробелдер менен бөлүп киритилишине мүмкүнчүлүк берет. Ошондуктан **ReadLn** операторундагы тизме бүткөндөн соң кийинки киргизүү операторун иштеши үчүн албетте ENTER клавишин басуу шарт.

1-мисал.

```
Var a,b:Integer;
Begin
Read(a);
Read(b);
WriteLn('a+b=', a+b);
    End.
```

2-мисал.

```
Var
a,b:Integer;
Begin
Read(a, b);
WriteLn('a+b=', a+b);
    End.
```

Эки мисалда да маани киргизүү төмөнкү эки усулдан биринде ишке ашырылышы мүмкүн.

1-усул: программа ишке түшүрүлгөндөн соң, экранда жолчонун башында жүргүч чыгат жана программа **a** нын мааниси киритилишин күтүп турат. Мисалы, **a** нын мааниси катары 10 киритилет, кийин пробелди басып **b** нын мааниси катары 11 деп киритилет. Эми ENTER клавиши басылса, экранда төмөнкүлөр көрүнөт:

10 11
a+b=21

2-усул: программа ишке түшүрүлгөндөн соң, экранда жолчонун башында жүргүч чыгат жана программа **a** нын мааниси киритилишин күтүп турат. Мисалы, **a** нын мааниси катары 10 киритилет, кийин ENTER клавишин басып **b** нын мааниси катары 11 деп киритилет. Эми ENTER клавиши басылса, экранда төмөнкүлөр көрүнөт:

10
11
a+b=21

3-мисал.

```
Var a,b,g,m:Integer;
Begin
Read(a, b);
Read(g); m:=a+g+b;
WriteLn('Натыйжа= ', m);
    End.
```

4-мисал.

```
Var a,b,g,m:Integer;
Begin
Readln(a, b);
Read(g); m:=a+g+b;
WriteLn('Натыйжа= ', m);
    End.
```

3-мисалда да маани киритүү 1–2-усулдан биринде ишке ашырылышы мүмкүн.

4-мисалда **a** менен **b** өзгөрүүчүлөрдүн мааниси пробел же ENTER клавиши жардамында киртилиши мүмкүн. Өзгөрүүчү g нын маанисин киритүү үчүн **b** өзгөрүүчүнүн мааниси киритилгенден соң ENTER клавишин басуу шарт. Мында экранда төмөнкүлөрдөн бири болушу мүмкүн.

10 11 **12** **Натыйжа = 33**	**10** **11** **12** **Натыйжа = 33**

Өзгөрүүчүлөргө маани берүүдө өздөштүрүү оператору колдонулса, программа бир маани үчүн аткарыла берет, б.а. өзгөрүүчүнүн мааниси эстутумга бир жолу киритилет. Кандайдыр бир өзгөрүүчүнүн маанисин алмаштыруу үчүн болсо дайым программага кирип өзгөртүү зарыл болот.

Программада кандайдыр бир өзгөрүүчүгө маани берүүдө киргизүү оператору колдонулган болсо, анда программаны ишке түшүрүп өзгөрүүчүнүн маанисин клавиатурадан киритиле берет, б.а. **байланыш усулунда** өзгөртүлө берет.

Байланыш усулунда бир аз ыңгайсыздык бар, б.а. кайсы өзгөрүүчүгө маани киритип жатканыбызды эсте тутуубуз шарт. Бул ыңгайсыздыктан кутулуу үчүн **Write** же **WriteLn** операторунан пайдалануу мүмкүн. Мисалы, Write('a= '); ReadLn (a); Write('b= '); ReadLn (b); жазылса, кайсы өзгөрүүчүгө маани киритилип жатканы экранда көрүнүп турат.

ReadLn операторы бере турган дагы бир мүмкүнчүлүк бар. Бизге белгилүү болгондой, Паскаль программалоо тили амалдарды өтө тез аткарганы үчүн натыйжаны көрүүгө үлгүрүп болбойт. Мурда айтып өтүлгөндөй, экрандагы натыйжаны көрүү үчүн **ALT+F5** клавишалар жуптугу басылат.

Бардык натыйжалар экранга чыгарылгандан соң натыйжаны көрүп алганыбызга чейин программаны күттүрүү үчүн **end**. операторунан мурда тизмесиз **ReadLn** операторун жазуу жетиштүү. Мындай абалда программанын ишин аяктоо үчүн **ENTER** клавиши басылат. Албетте, бул оператор күтүлгөн натыйжаны бериши үчүн андан мурда жазылган акыркы киргизүү операторуна да LN кошумчасы менен жазылган болушу шарт.

Суроо жана тапшырмалар

1. *Маалыматтарды байланыш усулунда киргизүү операторунун милдетин түшүндүр.*
2. *Эмне үчүн бул усул маалыматтарды байланыш усулунда киргизүү деп аталат?*

3. Маалыматтарды киргизүү операторлорунун айырмасы эмнеде?
4. Өзгөрүүчүлөргө маани берүүдө киргизүү операторунун ыңгайлуу жагын түшүндүр.
5. Киргизүү операторундагы маани берилип жаткан өзгөрүүчүнү кандай усулда билүү оңой?
6. Натыйжаны көрүп алууда ReadLn оператору бере турган ыңгайлуулукту түшүндүр.

Көнүгүүлөр

1. Read оператору жардамында *N* санынын квадратын *N* дин 10; 11; 12; 13; 14; 15 маанилеринде эсептөө программасын түз.

2. Төмөнкү программада *a* өзгөрүүчүнүн «ӨЗБЕКСТАНЫМ»; «МЕКЕНИМ»; «АТА ЖУРТУМ» маанилерин эстутумга ыңгайлуу усулда киритип натыйжа ал.

```
Var a, b, g: string;
Begin
        b:= ' – МЕКЕН СЫЙЫНААР'; g:= ' ЖЕРИМ!';
            write(a, b, g); ReadLn;
End.
```

3. «Матиз» автомобили ордунан козголуп *T* секундда *S* метр жол басты. Анын орточо ылдамдыгын м/с дарда төмөнкү маанилерде эсептөө программасын маалыматтарды байланыш усулунда киритүү аркылуу түз (көмөк: *V=S/T*).

а) T= 10; S= 150; б) T= 12; S= 200;
в) T= 20; S=400; г) T= 45; S= 900.

24-сабак. Маалыматтарды эстутумга байланыш усулунда киргизүү оператору темасын кайталоо

1. Төмөнкү программада a өзгөрүүчүнүн «азат»; «абат» маанилерин эстутумга ыңгайлуу усулда киритип, натыйжа ал.

```
Var a, b, g: string;
Begin
   b:= 'Бизден  '; g:= ' мекен калсын!'; write(b, a, g); ReadLn;
End.
```

2. Төмөнкү программадагы суроо белгисинин ордуна ушул жылды m өзгөрүүчү жардамында кирит жана эгемендүүлүгүбүз менен куттуктоочу натыйжа ал.

```
 Var a, b, g: string; m: word;
 Begin
write('Ушул жылды кирит: '); ?;
 a:= 'Эгемендүүлүктүн '; g:= ' жылдыгы менен '; b:='куттуктайбыз!';
 WriteLn(a); WriteLn(m-1991, g); write(b); ReadLn;
```

End.

3. Эгерде телого таасир этип жаткан күч *F*, алган ылдамдануусу a болсо, төмөнкү маанилерде телонун массасын эсептөөнүн программасын киргизүү операторунан пайдаланып түз (көмөк: m=F/a) жана натыйжалар ал.

а) F=15, a=55; б) F=55, a=15;

в) F=10, a=100; г) F=100, a=10;

4. a=19, b=2, d=1950 маанилерди эстутумга ыңгайлуу усулда киритип, төмөнкү туюнтмалардын маанисин эсептөө программасын түз.

а) $y = a + b^2 + ad$; б) $t = \sqrt{a+b} - \sqrt[3]{d-a}$;

в) $s = b \cos a + \sin d$; г) $n = \pi d^2 + ab$.

5. Жактары a, b, c болгон үч бурчтуктун аянтын эсептөө программасын киргизүү операторунан пайдаланып түз жана натыйжалар ал.

а) a = 5, b = 7, c = 4; б) a = 8, b = 6, c = 10;

в) a = 3, b = 4, c = 5; г) a = 10, b = 8, c = 10;

6. $y = 23x+1$ функциянын маанисин x тин −5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5 маанилерин эстутумга ыңгайлуу усулда киритип, эсептөө программасын түз жана натыйжалар ал.

7. $y = 21x^2 + 7x + 1963$ функциянын маанисин x тин −5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5 маанилерин эстутумга ыңгайлуу усулда киритип эсептөө программасын түз жана натыйжалар ал.

25-сабак. Текст абалында экран менен иштөө

Биз өткөн сабактарда маалыматтарды экранга чыгаруу усулдары менен тааныштык. Бул усулдарда маалыматтын ар бири экранга өзүнөн мурда чыгарылган маалыматтын уландысынан чыгарылат. Бирок, Паскаль программалоо тилинде маалыматтарды экрандын көрсөтүлгөн жайынан чыгаруу мүмкүнчүлүгү да бар. Мындан сырткары, экранга чыгарылып жаткан белгилерди жана алардын фонун ар түрдүү түстөрдө берүү да мүмкүн. Айтып өтүлгөндөй, Паскалдын экран менен иштөөгө багытталган процедура жана функциялары **Crt** модулунда жайгашкан. Ушул себептүү алардан пайдалануу үчүн ушул модулга кайрылуу керек. Бул үчүн программа башында **Uses Crt;** көрсөтмөсү берилет.

Crt модулу экранга түстүү маалымат чыгаруу мүмкүнчүлүгүн берсе, кандай түстөрдөн пайдалануу мүмкүндүгүн билип алалы. Паскалда, негизинен, 16 түрдүү түс иштетилиши мүмкүн. Алар 0 дөн 15 ке чейин бүтүн сандар менен коддолгон. **Crt** модулунда бул сандарга мүнөздүү константалар да бөлүнгөн. Бул константалардын аттары аларга мүнөздүү түстөрдүн англис тилиндеги туюнтмасы менен төп келет.

Төмөнкү жадыбалда Паскалда колдонула турган түстөрдүн коддору жана аларга мүнөздүү константалардын аттары келтирилген:

Түс	Коду	Константанын аты
Кара	0	Black
Көк	1	Blue
Жашыл	2	Green
Хрусталь	3	Cyan
Кызыл	4	Red
Кызгылт-көк	5	Magenta
Күрөң	6	Brown
Ачык боз	7	LightGray

Түс	Коду	Константанын аты
Ток боз	8	DarkGray
Көгүш	9	LightBlue
Ачык жашыл	10	LightGreen
Ачык хрусталь	11	LightCyan
Ачык Кызыл	12	LightRed
Ачык кызгылт-көк	13	LightMagenta
Сары	14	Yellow
Ак	15	White

Кандайдыр бир түстү тандоо үчүн Паскалдын атайын процедуралары жардамында кашаанын ичинде ушул түстүн коду же ага мүнөздүү константанын аты көрсөтүлөт. Түстөр текст жана фон үчүн түрдүүчө тандалышы максатка ылайык, антпесе текст фон ичинде көрүнбөй калат. Текст жана фон үчүн белгиленген акыркы түстөр ылайыктуу түрдө тексттин учурдагы түсү жана фондун учурдагы түсү деп аталат. Эгерде мурдатан кандайдыр бир түс тандалбаган болсо, текст үчүн ак, фон үчүн болсо кара түс учурдагы деп эсептелет.

Эми **Crt** модулунун курамына кирген кээ бир процедуралар менен таанышабыз. Текст жана текст фонунун түсү менен иштей турган процедуралар төмөнкүлөр:

Тексттин түсүн белгилей турган процедуралар	Текст фонунун түсүн белгилей турган процедуралар
TextColor(түс);	**TextBackGround(түс);**

бул жерде түс – өзгөрүүчү же туруктуу ченем болуп, тандалган түстүн коду же константанын атын туюнтат.

1-мисал.

```
Uses Crt;
Begin
TextColor(14); {же TextColor(yellow)}
WriteLn('Бул текст экранга сары түстө чыгат');
End.
```

Бул текст экранга сары түстө чыгат

2-мисал.

```
Uses Crt;
Begin
  TextColor(Yellow); TextBackGround(Blue);
WriteLn('Бул текст экранга сары түстө чыгат');
End.
```

Бул текст экранга сары түстө чыгат

Кээде экранда маалыматтар көбөйүп кеткендиктен керектүүсүн табуу кыйындашат. Мындай абалда колдонгон ClrScr процедурасы экранды тазалайт жана жүргүчтү экрандын башына (жогорку сол бурчка) орнотот. Эгерде программада ClrScr процедурасы текст фону түсүнөн кийин жазылса, анда экран текст фону түсүнө боёлот.

3-мисал.

```
Uses Crt;
Begin
  ClrScr;  {экран тазаланып, жүргүч экрандын башына орнотулат}
      TextColor(14); TextBackGround(2);
  WriteLn('Бул текст экранга жашыл фондо сары түстө чыгат');
End.
```

Бул текст экранга жашыл фондо сары түстө чыгат

4-мисал.

```
Uses Crt;
Begin
       TextColor(14); TextBackGround(2);
    ClrScr;  {экран тазаланып, экрандын түсү жашылга боёлот,
жүргүч экрандын башына орнотулат}
   WriteLn('Бул текст жашыл экранга сары түстө чыгат');
End.
```

Бул текст жашыл экранга сары түстө чыгат

Программанын натыйжасы «кооз» көрүнүштө болушу үчүн түрдүү түстөрдөн пайдалануу менен бирге аны экрандын керектүү жайынан чыгаруу да мааниге ээ. Ал үчүн жүргүчтү экрандын керектүү жайына орнотуу зарыл. Паскаль тилинде бул милдетти **GotoXY** процедурасы ишке ашырат. Анын жалпы көрүнүшү төмөнкүчө: **GotoXY(A,B)**;, бул жерде **A** жана **B** бүтүн сандуу өзгөрүүчү же туруктуулар болуп, **GotoXY(A,B)** процедурасы жүргүчтү экрандын A-мамыча жана B-жолчо кесишкен жайына орнотот. Экран текст абалында, негизинен, 80×25 өлчөмгө ээ. Башкача айтканда, 80 мамыча жана 25 жолчо (атайын операторлор жардамында бул өлчөмдү өзгөртүү да мүмкүн). Ошондуктан бул процедурада $1 \le A \le 80$ жана $1 \le B \le 25$ шарт аткарылышы зарыл.

5-мисал.

```
Uses Crt;
Begin
  ClrScr;     {экран тазаланды}
  GotoXY(23,12); {жүргүч  23-мамыча жана 12-жолчо кесишкен
жайга орнотулду}
  Write('Бул текст экрандын  ортосунда чыгат');
End.
```

Бул текст экрандын ортосунда чыгат

Бул программада экранга чыгарыла турган текст 34 белгиден турат. Тексттин сол жана оң жагынан бир түрдүү жай калтырып, экрандын ортосунан чыгаруу үчүн **GotoXY** процедурасындагы **A** жана **B** нын мааниси төмөнкүчө эсептелет:

A=[25/2]=12, B=[(80-33)/2]=23.

Суроо жана тапшырмалар

1. *Паскалда текст абалында экран менен иштөө үчүн кандай модулдан пайдаланылат?*
2. *Паскалда негизинен канча түрдүү түс иштетилиши мүмкүн?*
3. *Тексттин түсү кандай процедура аркылуу өзгөртүлөт?*
4. *Текст фонунун түсү кандай процедура аркылуу өзгөртүлөт?*
5. *ClrScr процедурасы кандай максаттарда колдонулат? Жообуңду мисалдар менен түшүндүр.*
6. *Экран текст абалында канча жолчо жана мамычадан турат?*
7. *Текстти экрандын каалаган жайына чыгаруу мүмкүнбү? Жообуңду түшүндүр.*

Көнүгүүлөр

1. Төмөнкү программанын натыйжасында экрандын түсү, текст фонунун түсү жана текст түстөрү кандай болушун жана тексттердин жайын аныкта.

```
Uses crt;
  Begin
  textbackground(yellow); WriteLn('Өзбекстан'); clrscr;
  textcolor(4); write('келечеги '); textbackground(blue);
  WriteLn('улуу'); textcolor(2); write('МАМЛЕКЕТ!'); ReadLn;
  End.
```

2. «Өзбекстандын Конституциясы – эркиндик сакчысы» текстин кызыл, текст фонуна көк түстү тандап экранга чыгар.

3. Төмөнкү программага процедуралар кошконунда, бардык текст көк түстө, текст фондору сары түстө экранга чыксын. Программанын иштешин түшүндүр.

```
Var a,b: string; m, s : real;
Begin
  a:= 'Квадраттын жагын киргиз: '; b:= 'Квадраттын аянты: ';
  Write(a); ReadLn(m); s:=sqr(m); write(b, s:8:2, 'квадрат  бирдик');
ReadLn;
 End.
```

4. A жана B өзгөрүүчүлөрдүн берилген маанисин киритип, төмөнкү программанын иштешин түшүндүр.

```
Uses Crt;
Var a,b: integer;
Begin  ClrScr;  write('A= '); ReadLn(a); write('B= '); ReadLn(b);
GotoXY(A,B);  WriteLn('Китеп билим булагы'); ReadLn;
End.
```

а) A=1, B=1; б) A=8, B=1; в) A=1, B=8;
г) A=8, B=8; д) A=25, B=25; е) A=100, B=10;

5. Экранга атың, фамилияң жана атаңдын атын 3 түрдүү түстө, текстти 3 түрдүү фон түсүндө жана экрандын түрдүү жайларында чыгар.

26-сабак. Текст абалында экран менен иштөө темасын кайталоо

1. «Ата-бабалардын мурасын барктайлы» текстине көк, текст фонуна жашыл түстү тандап экранга чыгар.

2. «Мекенди сүйүү ыймандуулуктан!» текстин экрандын оң тарабынан 12-жолчодо жашыл түстө кызыл фондо чыгар.

3. «Эң жогорудагы жолчо, солдон», «Эң жогорудагы жолчо, ондон», «Эң жогорудагы жолчо, ортодон», «Эң төмөнкү жолчо, солдон», «Эң төмөнкү жолчо, ондон», «Эң төмөнкү жолчо, ортодон», «Борбордогу жолчо, солдон», «Борбордогу жолчо, ондон», «Борбордогу жолчо, ортодон» тексттерин экрандын текст мазмунуна тиешелүү жайларында чыгышын камсыздоочу программа түз.

4. «Суу – жашоонун булагы» деген сөздү экранда 5 түрдүү түстө түрдүү жайларда чыгаруучу программа түз.

5. Экранга 5 классташындын атын түрдүү түстөрдө жана экрандын сары түсүндө чыгар.

27-сабак. Сызыктуу программалар түзүү

Адатта, сызыктуу алгоритмдердин программа түрүндө жазылышы **сызыктуу программа** деп аталат. Демек, сызыктуу программада бардык процедуралар удаалаш келүү тартибинде аткарылат жана эч кандай шарт текшерилбейт.

1-мисал. Радиусу *R* болгон айлананын узундугун эсептөө программасы түзүлсүн жана *R*=9 бирдик маани үчүн аткарылсын.

Чыгаруу. Айлананын узундугун эсептөө формуласын эске алабыз: L=2πR. Паскаль программалоо тилинде ал L := 2*pi*R түрүндө жазылат. Программада бир туруктуу pi жана эки өзгөрүүчү **R** жана **L** катышат. Маселенин шарты боюнча R=9, б.а. бүтүн сан. Ушул себептүү R өзгөрүүчү түрү Integer деп алынат. Айлананын узундугу L болсо көбөйтүүдө π катышкандыгы үчүн, албетте чыныгы (Real) түрдө болот. Айтылгандарды эсепке алып төмөнкү программа түзүлөт:

```
Program айлананын_узундугу;
   Var  R:Integer;  L:Real;


   Begin
      r := 9; L := 2*pi*R; WriteLn('L=',L,' бирдик.'); ReadLn;
   End.
```

Программа ишке түшүрүлгөндөн соң **(Ctrl+F9** клавишалар жуптугу басылгандан соң же **Run** менюсунан **Run** бөлүгү тандалгандан соң), экранда төмөнкү натыйжа пайда болот: **L= 5.6548667765E+01 бирдик.**

Программада L – real түрдө болгондуктан натыйжа экспоненциалдык түрдө чыкты. Эгерде чыгаруу операторунда L:7:2 форматынан пайдаланылса радиусу 9 бирдик болгон айлананын узундугу 56,54 бирдикке тең экени көрүнөт. Бул программанын жардамында каалагандай бүтүн радиустуу айлананын узундугун эсептөө мүмкүн. Болгону, дайыма программадагы **R** дин маанисин өзгөртүп туруу керек. Программага ар жолу өзгөртүү киргизбестик үчүн **R** дин мааниси киргизүү операторунун жардамында берилет. Радиустун мааниси дайым бүтүн сан боло бербестигин эсепке алып, ал **Real** түрдөгү өзгөрүүчү катары мүнөздөлөт. Буларды эсепке алып, төмөнкү программа түзүлөт:

Программасы	**Экрандагы натыйжа**
Program айлананын _узундугу; Var r,L : Real; Begin Write('Радиусту кирит : '); ReadLn(r); L := 2*pi*r; WriteLn('L= ',L,'бирдик); ReadLn; End.	Радиусту кирит : 9 L=5.6548667765E+01 бирдик

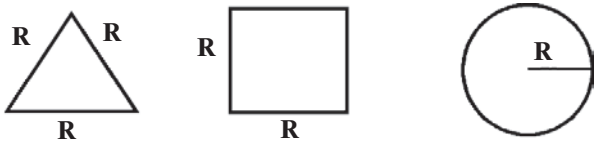
Бул программа ишке түшүрүлгөндөн соң, экранга «Радиусту киргиз:» жазуусу чыгат жана жүргүч ушул жолчодо калат. ReadLn оператору программанын ишин токтотуп, R өзгөрүүчүгө маани берилишин күтөт. Клавиатура аркылуу радиустун сандуу мааниси 9 ду киритип, ENTER клавишасы басылса, **R** өзгөрүүчүнүн мааниси 9 га барабар деп алынып, программа иштөөнү улантат. Натыйжада компьютер экранында изделген

натыйжа пайда болот. Программаны кайра-кайра иштетип, түрдүү радиустуу айланалардын узундугун эсептөөнү аткаруу мүмкүн.

2-мисал. Жактары тиешелүү түрдө *a, b, c* болгон каалагандай үч бурчтуктун аянтын Герондун формуласы менен эсептөө программасын түз жана $a = 3$, $b = 4$, $c = 5$ маанилерде эсепте.

I усул	II усул
Program Үч_бурчтуктун_аянты; Var a,b,c:Integer; {үч бурчтуктун жактары} жп,s:Real; {жп−жарым периметр, s−аянт} Begin a:=3; b:=4; c:=5; жп:=(a+b+c)/2; s:=sqrt(жп*(жп−a)*(жп−b)*(жп−c)); WriteLn('S= ',s,' квадраттык бирдик'); ReadLn; End.	Program Үч_бурчтуктун_аянты; Var a,b,c:Integer; {үч бурчтуктун жактары} жп,s:Real; {жп−жарым периметр, s−аянт} Begin Write('a,b,c нын маанилери киритилсин'); ReadLn(a,b,c); жп:=(a+b+c)/2; s:=sqrt(жп*(жп−a)*(жп−b)*(жп−c)); WriteLn('S=',s:2:2,'квадраттык бирдик'); ReadLn; End.
S = 6.0000000000E+00 квадраттык бирдик	a,b,c нын маанилери киритилсин 3 4 5 S = 6.00 квадрат бирдик

3-мисал. Жактары *R* болгон тең жактуу үч бурчтук, квадрат жана радиусу R ге тең болгон тегеректин аянтын эсептөө программасын түз жана R= 4 төэсепте.



Программасы	Экрандагы натыйжа

Program Аянттарды_эсептөө;
var r: Integer; s1,s2,s3:Real;
begin
Write('R дин маанисин кирит :');ReadLn(r);
s1:=sqr(r)*sqrt(3)/4; s2:=sqr(r);
s3:=pi*sqr(r);
WriteLn('Үч бурчтуктун аянты = ',s1);
WriteLn('Квадраттын аянты = ',s2);
WriteLn('Тегеректин аянты = ',s3); ReadLn;
End.
	R дин мааниси киритилсин 4
	Үч бурчтуктун аянты = 6.9282032303E+00
	Квадраттын аянты = 1.6000000000E+01
	Тегеректин аянты= 5.0265482457E+01

Ушул программаны өздөштүрүү оператору жардамында өзгөртүп, натыйжа алуу өз алдынча иш катары калтырылат.

Суроо жана тапшырмалар

1. Сызыктуу программа дегенде эмнени түшүнөсүң?
2. Кандай операторлордон пайдаланып байланыш программаларын түзүү мүмкүн?
3. Эмне үчүн программада π санына маани берүү шарт эмес?
4. Чыгаруу формату жөнүндө айтып бер.
5. Экрандагы натыйжаны көрүү үчүн кайсы клавишалар жуптугу басылат?

Көнүгүүлөр

1. Төмөнкү сызыктуу программалардын ишин түшүндүр жана натыйжасын аныкта.

а)
```
Var a,b:String;
  Begin
   a:= 'Өзбекстан ';
   b:= 'Мамлекет';
  WriteLn(a, 'Эгемендүү ', b);
 End.
```

б)
```
var a,b:Integer;  s:Real;
  Begin a:=4;
   a:=sqr(a); b:=b−a;
   s:=2*a+3*b;
    WriteLn('S= ',s);
 End.
```

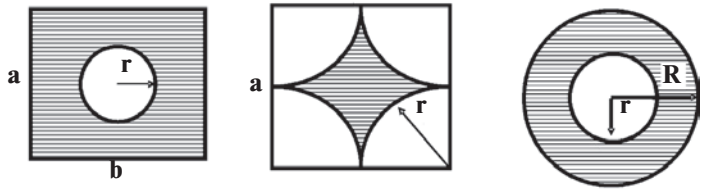
2. Төмөндө трапециянын аянтын эсептөө программасы тартипсиз жазылган. Операторлорду маанисине карай туура удаалаштыкта жайгаштыр.

p := (a+b) /2; s := p*h; Program трапециянын_аянты; End. WriteLn('S =', s,'квадраттык бирдик'); ReadLn(a,b,h); Begin Write('A,B,H маанилерди кирит:'); Var a,b,h: Integer; p,s:Real;

3. Үч бурчтуктун *a, b* жактары жана алардын арасындагы α бурч берилген. Үч бурчтуктун аянтын эсептөө программасын түз (көмөк: $S = \frac{1}{2}\, a{\cdot}b{\cdot}\sin a$).

28-сабак. Сызыктуу программалар түзүү темасын кайталоо

1. Төмөндө берилген фигуралардын штрихтелген бөлүктөрүнүн аянттарын эсептөөнүн программасын түз (көмөк: кандай фигуранын аянтынан кайсы фигуранын аянтын кемитүү керек?).



2. Берилген **a** жана **b** бүтүн сандардын маанисин алмаштыруучу программа түз, б.а. a = 7 жана b = 2 киритилсе, a = 2 жана b = 7 натыйжа чыксын (көмөк: орун алмаштыруу m: = a, a: = b, b: = m).

3. Тиешелүү түрдө R_1, R_2, R_3, R_4 каршылыкка ээ болгон өткөргүчтөр параллель уланганда пайда боло турган R каршылыкты эсептөө программасын түз (көмөк: $1/R = 1/R_1+1/R_2+1/R_3+1/R_4$).

29-сабак. Өтүү жана тармакталуу операторлору

Биз бүгүнгө чейин сызыктуу, б.а. командалары удаалаш аткарыла турган программалар менен тааныштык. Кээде берилген масалени чечүүдө операторлордун аткарылуу тартибин бузууга, б.а. башкарууну программа боюнча аркага же алдыга өткөрүү зарыл болот. Бул үчүн программада башкаруу узатылып жаткан операторго белги коюлат. Белги өзгөрүүчүнүн аты сыяктуу латын тамгалары жана цифралары жардамында пайда кылынат. Мисалы, 7, N1, белги2. Аларда иштетиле турган сан 0 дөн 9999 га чейин боло алат. Биринчи келген нөлдөр эсепке алынбайт.

Программада колдонула турган белгилер программанын мүнөздөмө бөлүгүндө **Label** кызматчы сөзү жардамында көрсөтүлүшү шарт. Белгилерден программада өтүү оператору колдонулса гана пайдаланылат. Өтүү оператору төмөнкү көрүнүшкө ээ: **GOTO <белги>;**, бул жерде **GOTO** (англ.- ге өтүлсүн) оператору башкарууну программанын алдына **белги** коюлган операторуна узатат.

1-мисал. Label N1; Var a,b,c:Integer; Begin a:=15; b:=13; c:=a+b; GoTo N1; {Башкаруу N1 белгилүү операторго узатылды} c:=a–b; N1: WriteLn(c); End.	Бул программанын иштөөсү натыйжасында экранда пайда болгон *c* нын мааниси 28-ге тең. Анткени, башкаруу N1 белгилүү чыгаруу операторуна узатылганы үчүн *c*:= a–b амалды аткарбастан өткөрүп жиберди.

Өтүү операторунда эч кандай шарт текшерилбестен башкаруу көрсөтүлгөн белгилүү операторго узатылат. Бирок көпчүлүк маселелерди чечүүдө кандайдыр бир шарттын аткарылышына карап ал же бул амалдардын удаалаштыгын аткаруу керек болот. Мисалы, квадраттык теңдемени чыгарууда натыйжаны эсептөө үчүн дискриминанттын белгисине карап үч багыттан бири тандалат. Бул түшүндүрмөлөр тармакталуучу алгоритмдер темасын эсиңе салган болушу керек. Мындай маселелерди чечүү үчүн Паскалда **тармакталуу оператору** колдонулат.

Тармакталуу операторунун жалпы формасы төмөнкүчө:

If <шарт> **Then** <оператор же операторлор> **Else** <оператор же операторлор>;

Бул жерде **if, then** жана **else** Паскалдын кызматчы сөздөрү болуп, алардын окулушу жана мааниси төмөнкүчө: **If** (иф) – «эгерде», **Then** (зен) – «анда», Else (элз) – «антпесе». Адатта, <шарт> чын же жалган маанилерден бирин кабыл алуучу логикалык туюнтма; <оператор же операторлор> Паскалдын каалагандай оператору же операторлорунун удаалаштыгы болуп саналат. **Else** кызматчы сөзүнөн мурда жазылган оператордон кийин «;» (үтүрлүү чекит) коюлбастыгын эсте тут. Тармакталуучу операторду төмөнкүчө иштейт: оболу **шарт** текшерилет, эгерде анын мааниси **чын** болсо; **then** ден кийинки оператор же операторлордун удаалаштыгы, антпесе **else** ден кийинки оператор же операторлордун удаалаштыгы аткарылат.

2-мисал. Киритилген сан 25 тен чоң болсо, экранга «чоң», антпесе «чоң эмес» деген жазууну чыгаруучу программа түз.

Чыгаруу. Берилген сандын түрү көрсөтүлбөгөндүктөн аны чыныгы түрдө деп алынат.

```
Program Салыштыруу;
Var a:Real;
Begin Write('Каалагандай сан кирит: '); ReadLn(a);
  If  a>25 Then WriteLn('Чоң')
           Else WriteLn('Чоң эмес');
End.
```

Тармакталуучу оператордун бөлүктөрүн өзүнчө жолчолордо жазуу да мүмкүн.

Эгерде THEN же ELSE сөздөрүнөн кийин аткарылышы керек болгон эки же андан ашык операторлордун удаалаштыгы жазылган болсо, бул операторлордун удаалаштыгы албетте begin кызматчы сөзү менен баштапып, **end; кызматчы сөзү** менен аякташы керек.

3-мисал. Киритилген *a* санынын *b* санга көбөйтүндүсүн жана катышын эсептөөчү программа түз.

Чыгаруу. Берилген сандардын түрү көрсөтүлбөгөндүктөн алар чыныгы түрдө деп алынат.

```
Program Катыш;
Label аягы;
Var a, b: Real;
Begin
   Write('a санды кирит: '); ReadLn(a);
   Write('b санды кирит: '); ReadLn(b);
   WriteLn('Көбөйтүүчү: ', a*b);
```

```
If b=0 Then begin WriteLn('Бөлүүнү аткаруу мүмкүн эмес'); goto аягы; end;
WriteLn(' Бөлүүчү: ', a/b);
аягы: End.
```

Тармакталуучу оператордун **Else** бөлүгү зарылдыкка карап иштетилет. Тактап айтканда, тармакталуучу оператордон төмөнкү түрдө да пайдалануу мүмкүн:

If <шарт> **Then** <оператор же операторлор>

Бул тармакталуучу оператордун **кыска** түрү деп аталат. Мында **шарттын** мааниси чын болсо **Then** ден кийинки оператор же операторлордун удаалаштыгы аткарылат, антпесе башкаруу кезектеги (тармакталуучу оператордон кийинки) операторго өтөт.

4-мисал. Берилген бүтүн сан терс болсо, бул сандын кубу менен алмаштыруучу программа түз.

Чыгаруу.

```
var a:Integer;  {берилген сан}
begin  Write('Каалагандай бүтүн сан кирит: '); ReadLn(a);
  If a<0 Then a:= a*a*a; {сан терс болсо кубу менен алмаштырылат}
 WriteLn(a); readln; {ReadLn оператору программа натыйжасын көрүп алуу үчүн жазылат}
 End.
```

Тармакталуучу оператордун курамында дагы тармакталуучу оператор колдонушу мүмкүн.

5-мисал. Сандын белгисин аныктоочу программа түз.

```
Var a:Integer; b:String; {Бир жолчодо бир канча өзгөрүүчүнү мүнөздөө мүмкүн}
Begin
  Write(Каалагандай сан кирит: '); ReadLn(a);
  If a<0 Then b:= 'терс' Else If a>0 Then b:= 'оң' Else b:=  'нөл';
   WriteLn(b);
End.
```

6-мисал. Эки сандан чоңун табуу (ЭЧТ) программасын түз.

```
Var a,b,чоң:Real;
Begin
  Write('Биринчи санды кирит = '); ReadLn(a);
  Write('Экинчи санды кирит = '); ReadLn(b);
  If a>b Then чоң:=a Else чоң:=b; WriteLn('Чоң сан= ', чоң);
End.
```

Бул программада *a>b* шарт аткарылышы же аткарылбастыгынан көз карандысыз түрдө WriteLn('Чоң сан= ', чоң) оператору сөзсүз аткарылат. Анткени, ал программада тармакталуучу оператор менен бир жолчодо жа-

зылган болсо да анын курамына кирбейт. Эмне үчүн ушундай экендигин ойлоп көр!

Суроо жана тапшырмалар

1. *Белги эмне үчүн колдонулат?*
2. *Өтүү операторунун жалпы көрүнүшү кандай болот?*
3. *Өтүү оператору колдонулган программада белгилер иштетил бестиги мүмкүнбү?*
4. *Тармакталуучу оператор эмне үчүн колдонулат?*
5. *Тармакталуучу операторду операторлордун удаалаштыгы катышса, алар кандай кызматчы сөздөрдүн арасында жазылат?*
6. *Тармакталуучу операторду кыска жана толук көрүнүштөрү жөнүндө эмнелерди билесиң?*
7. *Кайсы операторду кийин үтүрлүү чекит жазылбайт?*

Көнүгүүлөр

1. Төмөнкү өтүү операторлорунан ката жазылганын аныкта.

а) Goto 10; б) goto 30;
в) goto −5; г) GoTo _5;
д) goto sin; е) goto 2_5;
ё) GOTO a_5;

2. Тармакталуучу оператор үчүн төмөнкү салыштыруу шарттарынан ката жазылганын тап.

а) a<>b; б) a<−b;
в) a><b; г) −a>0;
д) −1>0; е) a>>b;
ё) a:=b;

3. Төмөнкүлөрдөн ката жазылганын тап.

а) IF a=b THEN a:=a+1; ELSE b:=a;
б) IF a:=1 THEN a:=a+1 ELSE b:=a;

4. x тин берилген маанисинде функция $y = \begin{cases} -1, & \text{эгерде } x > 0, \\ x^2, & \text{эгерде } x \le 0 \end{cases}$ маанисин эсептөө программасын түз.

5. Үч сан берилген. Алардын ичинде терс сандардын кубун эсептөөчү программа түз.

6. Берилген бүтүн сан терс болсо, анын модулун эсептөө программасын түз. Программаны 2 түрдүү усулда, модул эсептөө үчүн *abs*(x) функциясынан пайдаланып жана пайдаланбастан түз.

7. Пароль «информатика» болсо, анда пароль туура киритилгенин текшерүүчү программа түз.

30-сабак. Өтүү жана тармакталуу операторлору темасын кайталоо

1. Берилген маанинин негизинде шарттын маанисин аныкта.
   а) a:=10; b:=a*3; шарт: «a<b/3»;
   б) a:=10; b:=a*3; шарт: «a<=b/3»;
   в) a:=10; b:=a; шарт: «a+b=2*b»;
   г) a:=10; b:=a+3; шарт: «a+3>=b−3»;
2. Тармакталуу натыйжасында пайда боло турган маанилерди аныкта.
   а) aa:=7; bb:=6.6; if aa=round(bb) then mm:='Ооба' else mm:='Жок';
   б) ag:=true; if ag then aa:=21 else aa:=7; a:=a+1963;
   в) ag:=true; if ag then aa:=21 else begin aa:=7; a:=a+1963; end;
   г) ms:=50; aa:=10; if ms div aa = aa*5 then ms:=trunc(ms/3) else aa:= ms mod aa;
3. Эки *a* жана *b* сандар берилген. Эгерде *b* сан *a* дан кичине болсо, анда *b* ны нөл менен алмаштыруучу, антпесе *b* ны өзгөрүүсүз калтыруучу программа түз.
4. Үч *a*, *b* жана *c* сандар берилген. Бул сандардын бир гана ондорунун квадраттык тамырын эсептеп чыгаруучу программа түз.
5. $ax + b = 0$ тендеменин тамырын *a, b* нын төмөнкү маанилеринде эсептөө программасын түз.
   а) a= −1, b=1; б) a=0, b=4;
   в) a=1, b=0; г) a=1, b= −5.
6. Берилген *A* бүтүн сан берилген нөлдөн айырмалуу *B* бүтүн санга калдыксыз бөлүнүшү же бөлүнбөстүгүн аныктоочу программа түз.
7. Үч *a*, *b* жана *c* сандар берилген. Эгерде $a^2+b^2=c^2$ шарт аткарылса, бул сандардын суммасын, антпесе алардын модулдары көбөйтүндүсүн эсептөөчү программа түз.

31-сабак. Тармакталуучу структуралуу программалар түзүү

Буга чейин биз жөнөкөй шарттардын негизинде программа түзүүнү көрүп чыктык. Бирок тармакталуу операторунда татаал шарттардан да пайдалануу мүмкүн. Татаал шарттар жөнөкөй шарттарга **NOT** − «эмес», **AND** − «жана», **OR** − «же» логикалык амалдарын колдонуу натыйжасында пайда кылынат.

Эсинде болсо **NOT** − логикалык тануу, **AND** − логикалык көбөйтүү жана **OR** − логикалык кошуу амалдары деп жүргүзүлөт. Мындай амалдар менен биз 8-класстан таанышпыз. **NOT** өзүнөн кийин жазылган шарттын танылган маани-

син берет. **AND** эки жанында жайгашкан шарттардын экөө тең чын болгондо гана чын маани берет. **OR** эки жанында жайгашкан шарттардан эч болбогондо бириси чын болгондо гана чын маани берет.

Логикалык туюнтмаларда биринчи кезекте **NOT** амалы, экинчи кезекте **AND**, үчүнчү кезекте **OR** амалы аткарылат. Эгерде логикалык туюнтмаларда кашаалар катышкан болсо, алардын ичиндеги туюнтма биринчи болуп аткарылат. Тең күчтүү амалдар удаалаш келгенде, амалдар солдон оңго карап аткарылат. Логикалык амалдар колдонулганда, шарттар кашаа ичинде жазылат.

Мисалы,

1) $x \in [a,b]$ (б.а., $a \le x \le b$) Паскалда (A<=X) **AND** (X<=B) түрүндө жазылат;

2) $\overline{t_1 = t_2}$ Паскалда **NOT** (T1=T2) түрүндө жазылат;

3) $y < -5$ же $y > 2$ Паскалда (Y<−5) **OR** (Y>2) түрүндө жазылат.

1-мисал. *x* тин берилген маанисинде төмөнкү функциянын маанисин эсептөө программасын түз.

$$y = \begin{cases} x^2, & \text{эгерде } x \in (0,1] \text{ болсо,} \\ x, & \text{эгерде } x \in (0,1] \text{ болсо.} \end{cases}$$

Чыгаруу.

```
Var x, y: real;
Begin   Write('x= '); ReadLn(x);
        If (0<x) And (x<=1) Then  y:=Sqr(x)  Else y:=x;
        WriteLn('y= ',y); ReadLn;
End.
```

Бул жерде 0<*x* жана *x*<=1 шарттардын экөөсү тең аткарылганда, б.а. *x* тин мааниси (0,1] аралыкка тиешелүү болсо, y:=Sqr(x) өздөштүрүү оператору, антпесе, б.а. *x* тин мааниси (0,1] аралыкка тиешелүү болбогондо, *y*:=*x* өздөштүрүү оператору аткарылат.

Төмөнкү тармакталууга таандык мисалдар көңүл бурууга ылайык:

1) If (*A*>*B*) And (*B*>*C*) Then S:=B+7 ELSE S:=A*B−1;

Эгерде *A*>*B* жана *B*>*C*, б.а. *A*>*B*>*C* болсо, анда S:=B+7 оператору аткарылат, антпесе S:=A*B−1 оператору аткарылат.

2) If 5*B=M*M Then Goto 200 ELSE Goto 400;

Эгерде 5*B=M*M болсо, анда башкаруу «200» белгилүү операторго, антпесе «400» белгилүү операторго өтөт.

3) If R1<=R2 Then begin WriteLn(S); R:=R1+R2 end
Else begin WriteLn(S*R1); R1:=R2; R2:=0; end;

Эгерде R1<=R2 болсо, анда WriteLn(S) жана R:=R1+R2 операторлору аткарылат, антпесе WriteLn(S*R1); R1:=R2; жана R2:=0; операторлору аткарылат.

4) If SR ='ЖАШЫЛ' Then WriteLn('Өтүү мүмкүн') Else

Эгерде СТ (сфетофордун түсү) нүн мааниси «ЖАШЫЛ» болсо WriteLn ('Өтүү мүмкүн') оператору, антпесе WriteLn('Өтүү мүмкүн эмес') оператору аткарылат.

Көрүнүп тургандай, эгерде берилген шарт орундуу болсо, анда THEN кызматчы сөзүнөн кийин жазылган көрсөтмөлөр аткарылат, антпесе ELSE кызматчы сөзүнөн кийин жазылган көрсөтмөлөр аткарылат. Бул жерде үчүнчү абал болбостугун терең түшүнүп алуу зарыл.

Эми тармакталуучу программаларга мисалдар көрөбүз:

2-мисал. Берилген үч *a*, *b*, *c* сандарынын ичинен чоңун табуу (ҮЧТ) программасын түз.

```
Program ҮЧТ;
Var a,b,c,max : Real;
Begin
Write('a,b,c сандарынын
маанисин кирит: ');
ReadLn(a,b,c);
 If a>b Then max:=a Else max:=b;
  If c>max Then max:=c;
WriteLn('Берилген үч сандан
чоңу= ',max);
End.
```

3-мисал. Берилген натуралдык сандын так же жуптугун аныктоо программасын түз.

```
Program Так_жуп;
Var n : word;
Begin
 Write('Натуралдык санды
кирит:');
      ReadLn(n);
If Odd(n) Then WriteLn('ТАК')
Else WriteLn('ЖУП ');
End.
```

4-мисал. $ax^2 + bx + c = 0$ квадраттык тендемени чыгаруу программасын түз.

```
Program Квадраттык_тендеме;
Label Аяктоо;
Var a,b,c,d,x1,x2 : Real;
Begin
  Write('a,b,c лардын маанисин кирит: '); ReadLn(a,b,c);
  d:=Sqr(b) −4*a*c;    {Дискриминант эсептелди}
  If d<0 Then begin WriteLn('Чыныгы чечим жок'); Goto аягы;
       End;
  If d=0 Then begin WriteLn('Чечим бирөө: ');
  WriteLn('x= ', −b/(2*a)); Goto Аяктоо; end;
```

```
    WriteLn('Чечим экөө: ');
     x1:=(−b−Sqrt(d))/(2*a);  x2:=(−b+Sqrt(d))/(2*a);
    WriteLn('x1= ',x1); WriteLn('x2= ',x2);
Аяктоо: ReadLn;
End.
```

Жогорудагы маселелердин чечимдеринен көрүнүп тургандай, тармакталуучу программаларда тармакталууну түзүү коюлган маселенин маанисинен келип чыгат.

Суроо жана тапшырмалар

1. *Тармакталуу операторунун кыска жана толук түрлөрүнүн арасында кандай айырма бар?*
2. *Паскалда кандай логикалык амалдар колдонулат?*
3. *Паскалдагы татаал логикалык шарттарга мисалдар келтир.*
4. *Логикалык туюнтмада амалдардын аткарылуу тартибин түшүндүрүп бер.*
5. *Логикалык туюнтмада качан кашаалар колдонулат?*

Көнүгүүлөр

1. Төмөнкү берилген операторлордогу каталарды аныкта жана түшүндүр.

а) IF d>0 THEN 63 ELSE s:=d+a;

б) IF s1<>s2 THEN ELSE g1:=s1*s2;

в) IF i*j THEN goto vo ELSE goto ne;

г) IF x<>0 AND x<=5 THEN y=4*sin(x);

2. Төмөнкү логикалык туюнтмалардагы амалдардын аткарылуу тартибин аныкта.

а) a<−6 OR a>=0 AND a<4 ;

б) x*x +y >0 AND a=0.1 OR (b>3.7 AND s<>k4) ;

в) v= 'ооба' AND x1>0 AND x2>0 ;

г) a>0 OR a<1 OR NOT x*x+x*x<=1 ;

д) NOT v<=b AND (f<=f1 OR t='.')

е) NOT(NOT(NOT(a>b) OR TRUE) AND FALSE) ;

3. Узундуктары аркылуу берилген үч кесиндиден үч бурчтук пайда кылуу мүмкүн же мүмкүн эместигин аныктоочу программа түз.

4. Кенже класстын окуучусун көбөйтүү жадыбалы менен сыноочу программа түз. Туура жооп берилгенде «Азаматсын», антпесе «Кайра иште» тексттери түрдүү түстө чыксын.

5. Киритилген 1 ден 7 ге чейин аралыкта болгон цифрага негизинен апта күнүн экранга чыгаруучу программа түз.

32-сабак. Тармакталуучу структуралуу программалар түзүү темасын кайталоо

1. Татаал логикалык амалдардын натыйжасын аныкта.

а) a:=true; b:=true; m:=false; bb:=NOT(a AND m) AND (a OR b) OR m;

б) a:=77; b:=11; m:=7; ms:= (a div b=m) AND (a mod m=0) AND NOT((a>b) OR (b<m));

2. Тармакталуунун натыйжасында пайда боло турган маанилерди аныкта.

а) x:= −1; y:=0; a:= 0.1; IF (x*x +y >0) AND (a=1/10) THEN mm:=true else mm:= false;

б) x1:=sqr(−1); v:= 'h a'; x2:=sqrt(x1); IF (v= 'ooба') AND (x1>0) AND (x2>0) THEN x1:=0;

3. Үч *a*, *b* жана *c* сан берилген. $a<b<c$ барабарсыздыктын аткарылуу же аткарылбастыгын текшерүүчү программа түз.

4. Берилген *A* бүтүн сан берилген *B* бүтүн санга калдыксыз бөлүнсө, бул эки сандын суммасынын квадратын, антпесе көбөйтүндүсүн чыгаруучу программа түз.

5. Берилген бүтүн *N* сан оң жана 5 ке эселүү болсо, ушул сандын квадраттык тамырын, антпесе квадратын эсептөө программасын түз.

6. *M* жана *N* сандар берилген. Эгерде алар оң жана суммасы 100 дөн чоң болсо, *M* сандын *N* санына катышын, алар оң жана суммасы 100 дөн чоң болбосо *M* дин *N* ге көбөйтүндүсүн эсептөө программасын түз.

7. Берилген *N* сандын бүтүн бөлүгү бөлчөк бөлүгүнүн 1000 ге көбөйтүлгөнүнөн чоң болсо, сандын бүтүн бөлүгүнүн, антпесе бөлчөк бөлүгүнүн биринчи 3 цифрасын чыгаруучу программа түз.

33-сабак. Параметрлүү кайталоо оператору

Көпчүлүк маселелерди чыгарууда белгилүү бир амалдардын удаалаштыгын бир канча жолу кайталоо зарыл болот. Мурдагы бөлүмдө бул сыяктуу маселелер менен таанышылды жана аларды чечүү үчүн кайталануучу алгоритмдер түздүң. Эми кайталануучу программалар түзүү усулдары көрүп чыгылат.

Кайталануучу программалар түзүү үчүн кайталоо операторлорунан пайдаланылат. Паскалда алар үчөө болуп, бул сабакта алардан бири — **параметрлүү кайталоо оператору** үйрөнүп чыгылат. Бул оператордун жалпы көрүнүшү төмөнкүчө:

For I: = N1 To N2 Do <кайталаныш денеси>;

бул жерде **For** (үчүн), **To** (чейин) жана **Do** (аткар) Паскалдын кызматчы сөздөрү; I — бүтүн түрдүү каалагандай өзгөрүүчү бо-

луп, ал кайталоо параметри дейилет; **N1** – кайталоо параметринин кабыл ала турган башталгыч маанiси; **N2** – кайталоо параметринин кабыл ала турган акыркы маанiси; <кайталануу денеси> – кайталанышы керек болгон оператор же операторлор удаалаштыгы. Кайталануу денесин операторлор удаалаштыгы түзгөн болсо, алар сөзсүз **begin** көрсөтмөсү менен башталып, **end;** көрсөтмөсү менен аяктайт. Кайталоо параметринин башталгыч жана акыркы маанилери туруктуу, өзгөрүүчү же туюнтма көрүнүшүндө болушу мүмкүн.

Бул оператор төмөнкүчө иштейт:

1. Баштап кайталоо параметри башталгыч маанини кабыл алат;
2. Эгерде кайталоо параметринин маанiси акыркы маанiден чоң болбосо, кайталоо денесин түзүүчү операторлор аткарылат, антпесе кайталануу токтотулат жана башкаруу кезектеги операторго узатылат;
3. Кайталоо параметринин маанiси бирге артат (ага 1 кошулат) жана 2-пунктка өтүлөт.

For оператору, негизинен, кайталануулар саны мурдатан белгилүү болгондо колдонулат.

1-мисал. «Өзбекстан – мекеним менин!» текстин экранга 20 жолу чыгаруучу программа түз.

Чыгаруу. Шарт боюнча экранга «Өзбекстан – Мекеним менин!» текстин 20 жолу чыгарылышы керек. Тактап айтканда WriteLn ('Өзбекстан – Мекеним менин!') операторун 20 жолу кайталанышы зарыл. Төмөнкүчө программа түзүлөт.

```
Program Кайталоо;
Var I : Integer;
Begin
  For I:=1 To 20 Do  WriteLn('Өзбекстан – Мекеним менин!');
End.
```

Ушул программада кайталоо параметри I нин башталгыч маанiси 1 ге, акыркы маанiси 20 га тең. Кайталоо денеси бирөө – WriteLn('Өзбекстан – Мекеним менин!') операторунан турат. Программа аткарылганда кайталоо параметри кезек менен 1, 2, 3,..., 20 маанилерди кабыл алат жана ар жолу WriteLn('Өзбекстан – Мекеним менин!'); операторун аткарылат. Натыйжада экранга «Өзбекстан – Мекеним менин!» текстин 20 жолу жаңы жолчодон чыгарылат. Программада кайталоо параметринин башталгыч маанисин 41 жана акыркы маанисин 60 ка өзгөртүлсө да натыйжа ушундай болот, анткени кайталануулардын саны 60–41+1=20.

2-мисал. Экранга 1 ден 20 га чейинки бүтүн сандарды өсүү тартибинде чыгаруучу программа түз.

Чыгаруу. Экранга чыгарыла турган сандар S менен белгиленет. Баштап S:=0 деп алынат. *S* тин маанисин *S*:=*S*+1 жардамында бирге арттырып WriteLn(S) жардамында экранга чыгарылат. Бул амалдарды 20 жолу кайталоо зарыл, ошондуктан For операторунан пайдаланылат.

```
Program Удаалаштык;
Var I, S : Integer;
Begin
        S:=0;
For I:=1 To 20 Do begin S:=S+1; WriteLn(S); end;
End.
```

Кайталоо параметрин кайталоо денесинде колдонуу да мүмкүн. Бирок анын маанисин өзгөртүп болбойт. 2-мисал программасын иликтеп, кайталануу учурунда *S* өзгөрүүчү *I* менен бирдей маанилер кабыл алып жаткандыгын көрүү мүмкүн. Андыктан, экранга *S* тин ордуна *I* ни чыгаруу да мүмкүн. Анда программада *S* өзгөрүүчүнү колдонуу зарылдыгы калбайт. Буларды эсепке алып, программа төмөнкүчө өзгөртүлөт:

```
Program Удаалаштык;
Var I : Integer;
Begin
  For I:=1 To 20 Do WriteLn(I);
End.
```

3-мисал. 1 ден 100 гө чейин болгон бүтүн сандарды азаюу тартибинде чыгаруучу программа түз.

Чыгаруу. Чыгарыла турган сандар сандык кайталоо параметри *i* деп белгиленет.

```
Program Сандар;
Var i,сан : Integer;
Begin
 сан:=101;
 For i:=1 to 100 Do Begin  сан:=сан−1; WriteLn(сан);  end;
End.
```

For операторунда кайталоо параметри чоң мааниден кичине мааниге карап азайып барышы да мүмкүн. Ал үчүн **To** кызматчы сөзүнүн ордуна **Downto** кызматчы сөзү колдонулат. Муну эсепке алып жогорудагы программаны төмөнкүчө жөнөкөйлөштүрүү мүмкүн:

```
Program Сандар;
        Var i : Integer;
Begin
        For i:=100 Downto 1 Do WriteLn(i);
End.
```

4-мисал. 1 ден 21 ге чейинки так сандардын суммасын эсептөө программасын түз.

Чыгаруу. 1 ден 21 ге чейинки сандарды карап чыгуу үчүн кайталоо оператору параметрин колдонуу мүмкүн. S=1+2+3+...+21 суммадагы кошулуучулар 255 тен чоң эмес, демек, мында параметр үчүн **byte,** S бүтүн жана терс болбогону үчүн **word** түрүн тандоо мүмкүн. Индекстердин так экендигин текшерүү үчүн Odd(x) функциясынан пайдалануу ыңгайлуу.

```
Program Сумм;
Var s: word; i: byte;
Begin
  S:=0;
For i:=1 to 21 Do If odd(i) then S:=S+i;
 WriteLn('S= ', S);
End.
```

5-мисал. Берилген A[1..21] массивдин жуп индекстүү элементтеринин суммасын эсептөө программасын түз.

Чыгаруу. Массивди киритүү жана индекстерин карап чыгуу үчүн дагы кайталоо оператору параметринен, индекстердин жуптугун текшерүү үчүн NOT(Odd(x)) тен пайдаланылат.

```
Program Массив;
Var i: Integer; s: real;
a: array[1..21] of real;
Begin
   For i:=1 to 21 Do Begin  WriteLn('a[', i, ']= '); ReadLn(a[i]); end;
  S:=0;
For i:=1 to 21 Do If NOT(odd(i)) then S:=S+a[i];
  WriteLn('S= ', S); ReadLn;
End.
```

Көрүп турганындай, массивдерди киритүү, чыгаруу жана алардын индекстерин саноо сыяктуу милдеттерди аткаруу үчүн параметрлүү кайталоо оператору өтө ыңгайлуу болуп саналат.

Суроо жана тапшырмалар

1. *Кайталануучу алгоритмдерге мисалдар келтир.*
2. *Параметрлүү кайталоо операторунун көрүнүшү кандай болот?*
3. *Кайталоо параметри кандай маанилерди кабыл алат?*
4. *Параметрлүү кайталоо операторунун башталгыч жана акыркы маанилерин түшүндүр.*
5. *Кайталоо параметринин маанилери чектелгенби?*
6. *Кайталоо операторунун иштөөсүн түшүндүр.*
7. *Кандай абалда To нун ордуна Downto кызматчы сөзү колдонулат?*

Көнүгүүлөр

1. Төмөнкү операторлордогу кайталануулар санын аныкта.

а) for i:=1 to 88 do b:=1;

б) for i:=73 to 161 do m:=2;

в) for i:= −21 to 0 do a:=3;

г) a:=5; b:=34; for i:=a+7 to b−1 do s:=s+1;

д) a:=5; b:=19; for i:=a*a to 2*b+8 do s:=s+1;

2. $y = 21x^2 + 7x + 1963$ функциянын маанисин x тин −5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5 маанилеринде эсептөө программасын түз.

3. $y = 23x + 1$ функциянын маанисин x тин [−15, 5] аралыктагы бүтүн маанилеринде эсептөө программасын түз.

4. A[1..17] массив берилген. Массивдин нөлгө тең элементтеринин индексин чыгаруучу программа түз.

5. $y = 2x + 19$ функциянын маанисин x тин [0, 10] аралыкта 0,25 кадам менен эсептөө программасын түз (көмөк: i=0 дө x=0; i=1 де x=0,25; ...; i=40 та x=10).

34-сабак. Параметрлүү кайталоо оператору темасын кайталоо

1. Төмөнкү операторлордогу каталарды аныкта жана түшүндүр.

а) for I= −15 to 5 do s:=s+I;

б) for kub:=100/10+11 to 1963 do begin a:=7;end;

в) for mag:=99 dawnto 1 do readl(aa);

г) for bma:= 0.5 to 10 do writeln(k);

2. Төмөнкү операторлордогу кайталануулар санын аныкта.

а) for k:=trunc(23/5) downto trunc(1/2) do m:=1991;

б) for s:=23 to 1 do m:=1963;

в) for J:=2 downto 19 do m:=1950;

г) for d:=23 downto 1 do m:=2009;

д) for i:=abs(−25) to 25 do s:=s+i*i;

е) for h:=round(9.6) downto trunc(3*3) do a:=21;

3. S=10+12+14+...+50 сумманын эсептөө программасын түз.

4. $S = \frac{7}{11} + \frac{17}{21} + \frac{27}{31} + ... + \frac{2007}{2011}$ сумманын эсептөө программасын түз (көмөк: J ни 10 го бөлгөндө калдык 7).

5. P=1*3*5*...*23 көбөйтүүсүн эсептөө программасын түз.

6*. A[1..5] массив берилген. Массив элементтерин тескери тартипте чыгаруучу программа түз.

7*. A[1..15] массив берилген. Массивдин так индекстүү элементтеринин суммасынан жуп индекстүү элементтеринин суммасын кемитип чыгаруучу программа түз.

35-сабак. Шарт боюнча кайталоо операторлору

Мурдагы көрүлгөн мисалдарда кайталануулардын саны анык болчу. Бирок кандайдыр бир амалдардын удаалаштыгын белгилүү бир шарт аткарылганда кайталоо керек боло турган маселелер да кездешет. Мында кайталануулардын санын алдын ала билип болбойт. Мындай абалдарда шарт боюнча кайталоо операторлорунан пайдаланылат. Паскалда мындай операторлор экөө: **While** жана **Repeat**.

While оператору төмөнкү жалпы көрүнүшкө ээ:

While <шарт> Do <кайталануу денеси>;

мында **While** (англ. т.) же **Do** Паскалдын кызматчы сөздөрү; **<шарт>** – жөнөкөй же татаал логикалык туюнтма; **<кайталануу денеси>** – кайталоо денесин түзүүчү оператор же операторлор удаалаштыгы. Эгерде кайталануу денесинде операторлор удаалаштыгы жазылса, алар **begin** менен башталып, **end;** менен бүтөт.

Бул кайталоо оператору төмөнкүчө иштейт:

Баштап **шарт** текшерилет. Эгерде анын мааниси **чын** болсо, кайталоо денесин түзүүчү операторлор иштейт жана дагы **шарт** текшерилет. Бул жараян шарт **жалган** маани кабыл алганга чейин уланат.

1-мисал. Эки сандын эң чоң жалпы бөлүүчүсүн (ЭЧЖБ) табуучу программа түз.

Чыгаруу. ЭЧЖБ табуунун эң ыңгайлуу усулу – Евклид алгоритми. Бул алгоритм сага мурдагы бөлүмдөн тааныш. Программасы төмөнкүчө:

```
Program ЭЧЖБ;
Var a, b : Integer;
Begin
        Write('Биринчи санды кирит: '); ReadLn(a);
        Write('Экинчи санды кирит: '); ReadLn(b);
        While a<>b Do  If a>b Then a:=a–b Else b:=b–a; {a=b болгондо
кайталоо бүтөт}
        WriteLn('ЭЧЖБ= ', a);
End.
```

Repeat оператору да шарт боюнча кайталоону ишке ашырат. Анын жалпы көрүнүшү төмөнкүчө:

Repeat
<кайталануу денеси>
Until <шарт>

мында **Repeat** (англ. кайталоо) жана **Until** (англ. чейин) Паскалдын кызматчы сөздөрү болуп, **Repeat** – кайталоонун башы, **Until** – кайталоонун аягын билдирет; **<шарт>** – жөнөкөй же татаал логикалык туюнтма. Кайталануу денеси шарт чын маани кабыл алынганга чейин аткарыла берет.

2-мисал. $S = 1{,}1+1{,}5+1{,}9+2{,}3+...+45{,}5$ сумманын эсептөө программасын түз.

Чыгаруу. Мисалды иликтеп көшулуучулар 0,4 кө айырмаланышы аныкталат. Параметрлүү кайталоо операторунда параметрдин мааниси чыныгы боло албайт. Мында шарт боюнча кайталоо операторлорунан пайдалануу ыңгайлуу. Төмөнкү эки чечимди салыштырып көр.

While жардамында	**Repeat жардамында**
Program Summ_real; Var J,S: real; Begin S:=0; J:=1.1; While J<= 45.5 do begin S:=S+J; J:=J+0.4; end; WriteLn('S= ',S); End.	Program Summ_real; Var J,S: real; Begin S:=0; J:=1.1; Repeat S:=S+J; J:=J+0.4; Until J>=45.5; WriteLn('S= ',S); End.

3-мисал. Кокустук сандар генератору (Random) нан пайдаланып, экранга ар түрдүү түстө «A» тамгасын чыгаруучу программа түз. Кызыл түстүү «A» белгиси чыгуусу менен программанын иши аякталсын.

Чыгаруу. Паскалда түстөр 0 дөн 15 ке чейин бүтүн сандар менен коддолушу маалым. Кокустук сандар генератору болгон Random(x) функциясы [0, x) аралыктан кокустук сандарды алып берет. Ошондуктан [0, 15] аралыктагы бүтүн сандарды кокустан алуу үчүн Random(16) функциясы колдонулат.

Random(x) функциясы ар жолу программа ишке түшүрүлгөндө бирдей сандар удаалаштыгын пайда кылат. Түрдүү сандарды пайда кылуу үчүн **Randomize** операторунан пайдаланылат. Бул оператор программада Random функциясынан баштап жазылышы керек.

Кызыл түс коду 4 кө тендигин эсепке алып, төмөнкү программа түзүлөт:

```
Program Түстүү_тамгалар;
Uses Crt;
Var түс : Integer;
Begin
        Randomize;
        Repeat
        түс:= Random(15); TextColor(түс); Write('A');
        Until түс=4;     {түс=4 (кызыл) болсо
кайталоо операторунан чыгылат}
End.
```

Repeat операторунун While операторунан биринчи айырмасы, While операторунда шарт кайталоонун башында текшерилсе, Repeat операторунда шарт кайталоонун аягында текшериле тургандыгында. Ошол себептен While операторунда кайталоо денесин түзүүчү операторлор бир жолу да аткарылбастыгы мүмкүн (шарт алдындан аткарылбаса), Repeat операторунда болсо жок дегенде бир жолу аткарылат. Экинчи айырмасы While операторунда кайталануу шарты аткарылбаганда («жалган» маани кабыл алганда) аяктаса, Repeat операторунда кайталануу шарт аткарылганда («чын» маани кабыл алынганда) аяктайт.

Жогорудагы 1-мисалда баштап шарт текшерилиши зарыл. Ошол себептүү анда While операторунан пайдаланылды. 3-мисалда болсо баштап түс мааниси аныкталып, кийин шарт текшерилиши керек. Ошондуктан анда Repeat оператору колдонулду. Жалпысынан алганда, 1-мисалда Repeat операторунан, 3-мисалда болсо While операторунан пайдаланса да болот. Болгону мында программага кээ бир кошумча операторлор киритүү зарыл болот. Бирок программа «көркөм» жана «түшүнүктүү» көрүнүшкө ээ болушу үчүн While жана Repeat операторлорун өз ордунда колдонуу максатка ылайык.

Суроо жана тапшырмалар

1. *Шарт боюнча кайталоо операторлорунан кайсыларын билесиң?*
2. *While операторунун иштешин түшүндүр.*
3. *Repeat операторунун иштешин түшүндүр.*
4. *Шарт боюнча кайталоо операторлорунун параметрлүү кайталоо операторунан айырмасы эмнеде?*
5. *Кайталоо операторлорун колдонуу ыңгайлуу болгон абалдарга ылайык кылып түшүндүр.*

Көнүгүүлөр

1. Төмөнкү операторлордогу каталыктарды аныкта жана түшүндүр.
   а) whIle 5*6 do SH:=sqr(2);
   б) WHILE 5>6 do Od:=Od+1;
   в) Repeat i<j Until s:=0;
   г) rEpEaT s:=0 UntiL s:=0;
2. Төмөнкү операторлордогу кайталануулар санын аныкта.
   а) x:= −5; while X>0 do x:=x+2;
   б) x:= −5; while X<10 do begin x:=x+2; x:=2*x; end;
   в) i:=0; while i*i <=1.2 do i:=i+0.1;
   г) k:=5; while k /5 <= 2.5 do k:=k+1.5;
   д) t:=100; repeat t:= t/10; until t<=0.1;
   е) x:=0; repeat x:=x+1/10; until sqr(x)>=6/5;
3. *N* натуралдык сан берилген. Квадраты *N* ден чоң болбогон бардык натуралдык сандарды чыгаруучу программа түз.

4. y = x*sin x функциянын маанилерин [−π,π] аралыкта 0,3 кадам менен эсептөө программасын түз.

5*. *N* натуралдык сан жана A_1, A_2, ..., A_N бүтүн сандардын удаалаштыгы берилген. Аларды удаалаш кошуп барып, сумма берилген *N* бүтүн сандан артышы менен экранга чыгаруучу программа түз. Эгерде бардык сандар суммасы *N* ден ашпаса, бул жөнүндө кабар чыгар.

36-сабак. Шарт боюнча кайталоо операторлору темасын кайталоо

1. S=0,5+1,5+2,5+...+98,5+99,5 сумманын эсептөө программасын түз.

2. S=1*2+3*4+5*6+...+101*102 сумманын эсептөө программасын түз.

3. N натуралдык санынын бардык бөлүүчүлөрүн чыгаруучу программа түз.

4. Берилген *N* натуралдык сандын канча цифрадан тургандыгын аныктоочу программа түз (көмөк: канча жолу *N*=*N* div 10 аткарылса *N*=0 болот?).

5. *N* натуралдык сан берилген. 1 ден *N* ге чейин болгон натуралдык сандар ичинде акыркы цифрасы 3 кө эселүү сандарды чыгаруучу программа түз.

6. Эки орундуу натуралдык сандардын ичинен цифраларынын суммасы жуп болгон сандарды чыгаруучу программа түз (көмөк: K сандын бирдик цифрасы K1=K mod 10, ондук цифрасы K10=K div 10).

7. A_1, A_2, ..., A_N бүтүн сандардын удаалаштыгы берилген. Ушул удаалаштыктын так элементтеринин көбөйтүндүсүнөн жуп элементтеринин суммасын кемитүүчү программа түз.

8*. *N* натуралдык сан жана *A*[1..*N*] массив берилген. *K*-элементи *A* массивдин биринчи *K* даана элементинин орточо арифметикалыгына тең болгон B[1..N] массивди пайда кылуучу программа түз (көмөк: B[K] = (A[1] + A[2] + ... + A[K]) / K).

9*. 1 ден чоң *A* сан берилген. $7^K > A$ шарт аткарыла турган эң кичине терс болбогон бүтүн *K* санды табуу программасын түз.

37-сабак. Кайталоого тиешелүү тапшырмалар

1. Төмөнкү сумманын мааниси берилген *M* натуралдык сандан артык болгонго чейин эсептей турган программа түз.

$$y = \frac{1}{3} - \frac{1}{10} + \frac{1}{21} - ... + \frac{(-1)^{j+1}}{j \cdot (2 \cdot j + 1)} - ...$$

2. 1-көнүгүүнү параметрлүү кайталоо оператору, шарт боюнча кайталоо операторлору жардамында түрдүүчө чеч.

3. N натуралдык сан берилген. (1/2), (3/4), (5/6), (7/8), ... удаалаштыктын N даана мүчөсүнүн суммасын табуучу программа түз.

4. A оң сан берилген. Эгерде k-квадраттын жагы $\frac{A}{k}$ болсо (k=1,2,...), k нын кандай маанисинде бардык квадраттардын аянттарынын суммасы биринчи жолу A^2 тан чоң болушун аныктоочу программа түз.

5. P га эселүү жана ага тең болбогон кандайдыр бир үч орундуу санды аныктоочу программа түз.

6. N натуралдык сан жана $A[1..N]$ массив берилген. Массивдин эң чоң жана эң кичине элементин табуучу программа түз.

7. Берилген M натуралдык сандын цифраларынын суммасын табуучу программа түз.

8*. B, M, A натуралдык сан берилген. Удаалаштык $Y_1=B$; $Y_i = \sqrt{M + A \cdot Y_{i-1}}$, $i = 2,3,...$ мыйзам ченемдүүлүктүн негизинде пайда кылынат. Удаалаштыктын $B*M*A$ санынан кичине болгон бардык мүчөлөрүн чыгаруучу программа түз.

9*. Клиент банкка B сом акча койду. Банктагы акчага жылына M пайыз үстөк кошулат. Канча жылдан кийин клиенттин акчасы A сомдон ашуусун аныктоочу программа түз.

10*. Кичи ишкана 1-күн B даана товар иштеп чыгарды. Кийинки ар бир күндө алдыңкы күндөгүгө караганда M даана ашыкча товар иштеп чыгарды. Иштеп чыгарылган бардык товардын саны пландаштырылган A даанадан биринчи жолу канча күндөн кийин ашканын аныктоочу программа түз.

11*. Санап жаткан адамдын тегерегинде айлана түрүндө B даана адам турат. Саноочу M ге чейин санап баргандан соң, M-саналган адам айланадан чыгат жана саноочу адам кийинки адамды саноону 1 ден баштайт. Саноо 1 адам калганга чейин уланат. Аягында канчанчы тартип сандагы адам калганын аныктоочу программа түз.

38-сабак. Белгилүү жана жолчолуу чоңдуктар менен иштөө

Паскалда белгилүү жана жолчолуу чоңдуктар менен иштөө үчүн атайын функция жана процедуралар киритилген. Төмөнкү жадыбалда алардын кээ бирлери келтирилген.

Программада функциялар колдонулганда алардын мааниси кандайдыр бир өзгөрүүчүгө өздөштүрүлөөрүн, ал эми процедуралар иштетилгенде өздөштүрүү операторусуз жазыла тургандыгын белгилеп өтүү керек.

Эми жөнөкөй мисалдар көрүп чыгылат.

Жазылышы	Функциясы
Стандарттык функциялар	
Concat(S1,S2, ...,SN)	**S1,S2,...,SN** жолчолуу (белгилүү) өзгөрүүчүлөр (туруктуулар)дү бири-бирине удаалаш улайт
Length(S)	**S** жолчонун узундугун (белгилүү) аныктайт.
Pos(b,S)	**S** жолчонун ичинен сөздү издейт
Copy(S,n1,n2)	**S** жолчонун **n1**-белгисинен баштап **n2** даана белгисинин нускасын алат
Ord(B)	**B** белгинин ASCII кодун аныктап берет
Chr(a)	ASCII коду **a** га тең белгини аныктайт
Стандарттык функциялар	
Delete(S,n1,n2)	**S** жолчону **n1**-белгисинен баштап **n2** даана белгисин алып таштайт
Insert(S1,S,n)	**S** жолчого **n**-орундан баштап **S1** жолчону орнотот
Str(a,S)	**S** жолчолуу өзгөрүүчүнүн мааниси а сандын жолчо түрүндөгү туюнтмасына тең болот
Val(S,a,c)	**a** сандуу өзгөрүүчүнүн мааниси **S** жолчолуу өзгөрүүчүнүн сан түрүндөгү туюнтмасына, ал эми **c** болсо нөлгө тең болот (эгерде берилген жолчону сан түрүндө туюнтууга болбосо **a** нын мааниси нөлгө тең болот, **c** нын мааниси нөлдөн айырмалуу болот)

1. a='Соо денеде ', b='соо акыл.' болсо, c:= **Concat**(a,b); оператору аткарылганда с нын мааниси 'Соо денеде соо акыл.' га тең. Бирок c:= Concat(a,b); ордуна c:=**a+b**; деп жазуу да ректүү натыйжаны берет.

2. a='информатика' болсо, n:=**Length**(a); оператору аткарылганда n дин мааниси 11 ге тең болот.

3. a:=**Pos**('m', 'информатика'); оператору аткарылганда а нын мааниси 6 га, a:=Pos('ma', 'информатика'); оператору аткарылганда да а нын мааниси 6 га, a:=Pos('sn', 'класс'); оператору аткарылганда а нын мааниси 0 гө, a:=Pos('v', 'класс'); оператору аткарылганда болсо, а нын мааниси 0 гө тең болот.

4. a:=**Copy**('информатика',3,5); оператору аткарылса, *а* нын мааниси 'форма' сөзүнө тең болот.

5. a:='A' болсо, **Ord**(a) функциянын мааниси 65 ке тең болот. Анткени 'A' (латын) тамгасынын ASCII коду 65. Ord функциясынын аргументи туруктуу болсо, ал апостроф ичинде жазылат. Мисалы, Ord('A').

6. cod:=65 болсо, **Chr**(cod) функциясынын мааниси латын **'A'** тамгасына, Chr(66) функциясынын мааниси болсо латын 'B' тамгасына тең болот.

7. a:='аткарылбады' болсо, **Delete**(a,8,2); процедурасы аткарылгандан соң натыйжа a:='аткарылды' болот. Муну схемалык көрүнүштө төмөнкүчө сүрөттөө мүмкүн:

(a:='аткарылбады' → Delete(a,8,2); → 'аткарылбады' → a:='аткарылды')

8. a:='аткарылды', b:='ma' болсо, **Insert**(b,a,8); процедурасы аткарылгандан соң натыйжа a:='аткарылбады' болот. Муну схемалык көрүнүштө төмөнкүчө сүрөттөө мүмкүн:

(a:='аткарылды', b:='ma' → Insert(b,a,8); → 'аткарыл'+'ба'+'ды' → a='аткарылбады')

9. a:=765 болсо, **Str**(a,s); процедурасы аткарылгандан соң, s:='765' болот.

10. s:='123' болсо, **Val**(s,a,c); аткарылгандан соң, a:=123 жана c:=0 болот; s:='34BMA5' болсо, Val(s,a,c); аткарылгандан соң a:= 0 жана c ≠ 0 болот.

Эми көрүп чыгылган функция жана процедураларды иш жүзүндө колдонууга мисалдар келтирилет:

1- мисал. Берилген 'эгемендүү ',' Өзбекстан ', 'мамлекет' сөздөрүнөн 'Өзбекстан эгемендүү мамлекет' сөзүн пайда кылуучу программа түз.

Чыгаруу. Берилген сөздөр программа иштеп жатканда өзгөртүлбөйт. Ошондуктан алар константа түрүндө туюнтулат.

```
Program Сөз_жасоо;
Const a='эгемендүү '; b='Өзбекстан '; c='мамлекет';
Var d : String;
Begin
        d:=Concat(b, '_' a,c); WriteLn(d);
End.
```

2-мисал. Киритилген сөздү тескерисине оодарып берүүчү программа түз. Мисалы, 'сары' сөзүнөн 'ырас' сөзү пайда болушу керек.

Чыгаруу. Киритилген сөз *a*, пайда боло турган сөз *b* менен белгиленет. *b* нын мааниси бош жолчого теңделет (b:= ''). *a* нын узундугу аныкталат жана анын сол жагынан баштап бирден белгисин алып *b* га сол жактан бириктирип барылат.

```
Program Тескери;
Var a, b, белги: String;  i, len : Integer;
Begin
        Write('Сөздү киргиз : '); ReadLn(a);  Len:=Length(a); b:='';
        For i:=1 To len Do   begin
```

```
        белги:=Copy(a,i,1);  {a нын i-белгисинин үлгүсү алынды}
        b:=белги+b;          {a дан үлгүсү алынган белги b нын сол
                              жагына  кошулду}
                              end;
        Write(b); ReadLn;
    End.
```

3-мисал. Берилген сөздө берилген белги бар же жоктугун аныктоочу программа түз.

```
    Program Издөө;
    Var сөз : String; белги : Char;
    Begin
        Write('Сөздү кирит : '); ReadLn(сөз);
        Write('изделип жаткан белгини кирит : '); ReadLn(белги);
        If Pos(белги,сөз)>0 Then WriteLn('БАР') Else
    WriteLn('ЖОК');
    End.
```

Программа аткарылгандан соң компьютер экранында 'БАР' же 'ЖОК' сөздөрүнөн бири көрүнөт.

4-мисал. 'saodat' жана 'qadoq' сөздөрүнөн 'sadoqat' сөзүн жасоочу программа түз.

```
    Program Сөз_жасоо;
    Var a,b : String;
    Begin
        a:='saodat'; b:='qadoq';
        Delete(a,3,2);   {a='saat' bo'ldi}
        Delete(b,1,2);   {b='doq' bo'ldi}
        Insert(b,a,3);   {a='sadoqat'}
        WriteLn(a);
    End.
```

5-мисал. Берилген 'Улуу акын Алишер Навоий', 'жылда туулган' сөздөрү жана 1441 сандан 'Улуу акын Алишер Навоий 1441-жылда туулган' сүйлөмүн пайда кылуучу программа түз.

```
    Program Алишер Навоий;
    Const a=' Алишер Навоий '; b=' – жылда туулган'; c=1441; d='Улуу
акын';
    Var жыл, s : String;
    Begin
        Str(c, жыл);  {c=1441 ден жыл='1441' пайда кылынды}
        s:=Concat(d,a,жыл,b);
    WriteLn(s);
    End.
```

6-мисал. Берилген бүтүн сандын цифраларынын суммасын эсептөөчү программа түз.

```
Program Цифралар;
Var сан, цифра, len, i, c, натыйжа : Integer; _сан, _цифра : String;
Begin   Write('Бүтүн сан киргиз: '); ReadLn(сан);
        Str(сан,_сан);   len:=Length(_сан); натыйжа:=0;
For i:=1 to len Do  begin
        _цифра:=Copy(сан, i, 1);  Val(_цифра,цифра,c);
        натыйжа:=натыйжа+цифра; end;
 WriteLn(сан, ' нын цифраларынын суммасы=', натыйжа);
End.
```

Ушул программа киритилген сан 32767 ден чоң болсо туура эмес натыйжа берет. Мунун себебин өзүң ойлоп көр. Программага кандай өзгөртүү киритилсе, 2147483647 ге чейин болгон сандарды да колдонуу мүмкүн болушун ойлоп көр!

Паскалда жолчолор «массив касиети»не ээ. Башкача айтканда жолчо – массив деп, жолчодогу белгилер болсо массивдин элементтери деп каралат. Мисалы, s – жолчолуу өзгөрүүчү болсо, s[3] – ушул жолчонун 3-белгисин билдирет. Башкача айтканда, s:='компьютер' болсо, s[3]= 'м'.

7-мисал. Берилген жолчодогу белгилердин ASCII коддорун экранга чыгаруучу программа түз.

```
Program Код;
Var s : String;
i, L, cod : Integer;
Begin
        Write('Жолчо киргиз : '); ReadLn(s);
L := Length(s); { киритилген жолчонун узундугу эсептелип,
L ге өздөштүрүлдү }
        For i:=1 to L Do WriteLn(Ord(s[i]));
End.
```

Жолчолордун «массив касиети» аркылуу көптөгөн маселелерди чечүү ыңгайлуу. Мисалы, 6-мисалды жолчолордун ушул касиетинен пайдаланып өзүң чечип көр.

Суроо жана тапшырмалар

1. *Concat функциясынын милдети эмнеден турат? Мисалдар келтир.*
2. *Concat функциясынын ордуна башка амалдан пайдалануу мүмкүнбү?*
3. *Кайсы функция жолчонун узундугун аныктайт? Мисалдар келтир.*
4. *Pos функциясы кандай милдетти аткарат?*
5. *Pos функциясынын мааниси качан нөлгө тең болот? Мисалдар келтир.*
6. *Copy функциясы эмне үчүн колдонулат?*

7. Сору функциясынын мааниси сандуу болушу мүмкүнбү? Мисалдар келтир.
8. Паскалда берилген жолчонун бөлүгүн өчүрө турган кандай процедура бар?
9. Паскалда каалагандай белгилүү же жолчолуу чоңдукту сандуу чоңдукка өткөрүүгө болобу? Жообуңду түшүндүр.
10. Ord жана Chr функцияларынын милдетин айтып бер.
11. Жолчолуу өзгөрүүчүлөрдүн «массив касиети»эмнени билдирет?

Көнүгүүлөр

1. Төмөнкү функциялардын аткарылуу натыйжасын аныкта.
а) Concat('э', 'не'); б) Concat('жа', 'ша','сын');
в) a:='дүйнө'; Length(a); г) Pos('о', 'жаз');

2. Жолчолуу өзгөрүүчү S тин мааниси 'Информатика' болсун. Төмөнкү амалдар аткарылгандан соң анын маанисин аныкта:
а) Delete(s,5,7);
б) Delete(s,1,2); Delete(s,6,4);

3. Киритилген сөздөн «эне» сөзүн пайда кылуу мүмкүнчүлүгүн аныктоочу программа түз.

39-сабак. Белгилүү жана жолчолуу чоңдуктар менен иштөө темасын кайталоо

1. Берилген сөздүн белгилеринин арасына бирден пробел кошуп чыгуучу программа түз.

2. *S* жолчо берилген. Андагы «*b*» тамгаларынын санын аныктоочу программа түз.

3. A[1..N] жолчолуу массив берилген. Массив элементтеринин ичинен «*м*» тамгадан баштала тургандарын экранга чыгаруучу программа түз.

4*. *S* жолчо цифралардан гана турат. Жолчодогу цифралардан эң чоң санды пайда кыла турган программа түз.

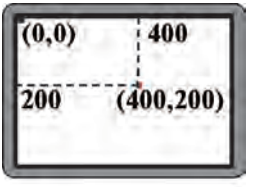
5*. *A* жолчодогу белгилерди ордун гана алмаштырып *B* жолчону пайда кылуу мүмкүн же жоктугун аныктоочу программа түз.

40-сабак. Паскалда экранды графикалык абалга өткөрүү

Бүгүнгө чейин компьютер экранына бир гана тексттүү маалыматты чыгаруу менен танышылды. Бирок компьютер экранында түрдүү сүрөттөр да пайда кылуу мүмкүн. Бул үчүн Паскалдын **Graph** (граф) модулунда чекит, түз сызык, туура төрт бурчтук, айлана сыяктуу бир канча фигураларды сызууга ылайыкташтырылган операторлор бар.

Фигура сызуу операторлорунан пайдалануу үчүн программанын башында **Uses Graph**; көрсөтмөсү берилет. Бул көрсөтмө анын курамындагы оператор жана функциялардан пайдалануу мүмкүнчүлүгүн гана берет. Ал эми бул операторлордун иштеши үчүн экран графикалык абалга өткөрүлүшү зарыл.

Графикалык абалда компьютер экраны майда чекит(пиксел) терден түзүлөт. Графикалык абалда да тексттүү абалдагы сыяктуу **жүргүч** бар болуп, ал чекиттен турат. Экранда пайда боло турган бардык схемалар жүргүчтүн экранда из калтырып же из калтырбастан жылышы натыйжасында пайда болот. Жүргүчтүн экранда турган орду анын координатасы менен аныкталат. Координатанын башы болгон (0,0) чекити экрандын жогорку сол бурчунда жайгашкан. Экран графикалык абалга өткөрүлгөндө жүргүч координата башында жайгашат. Координата октору X жана Y координата башынан тиешелүү түрдө оңго жана ылдыйга карап багытталган, б. а. маанилер ушул багыттарда өсүп барат. Экрандын жүргүч турган чекити **учурдагы чекит** дейилет. Экрандагы чекиттердин саны көбү менен **640x480** даана (0..639x0..479) болот.



Экранды графикалык абалга өткөрүү үчүн Graph модулунун **InitGraph(GD,GM,<жол>);** көрсөтмөсү пайдаланылат, мында **GD** (GraphDriver) жана **GM** (GraphMode) – бүтүн сандуу өзгөрүүчүлөр. Алардын мааниси компьютердин графикалык мүмкүнчүлүктөрүнө жана тандалган графикалык абалга байланыштуу болот. Эгерде GD:=0; (же GD:=Detect;) деп алынса, программа тарабынан эң эффективдүү графикалык абал автоматтык түрдө аныкталат. <жол> – графикалык абалда иштөөнү камсыздоочу **BGI** кеңейтмелүү атайын файл жайгашкан BGI каталогунун дареги болуп, заманбап компьютерлерде EGAVGA.BGI файлы колдонулат. Бул файл учурдагы каталогдо жайгашкан болсо <жол> ордунда бош жолчо жазылат.

Графикалык абалдан чыгуу, б.а. текст абалына кайтуу үчүн **CloseGraph** операторунан пайдаланылат. Графика менен байланыштуу программалар, негизинен, төмөнкү көрүнүштө болот:

```
Uses Graph;
Var Gd, Gm : Integer;
  {Графика менен байланыштуу маселеге мүнөздүү
  өзгөрүүчүлөр мүнөздөмөсү}
```

```
Begin
Gd := 0; {Графикалык драйверди автоматтык түрдө
аныктоо}
InitGraph(Gd,Gm, ''); {Графикалык абалга өтүү}
{Графика менен байланыштуу маселе чечими жазыла
турган бөлүк}
ReadLn; CloseGraph; {Графикалык абалдан чыгуу}
End.
```

Паскаль тилинде 16 түрдүү түстөн пайдалануу мүмкүн болот. Бул түстөр 0 дөн 15 ке чейин бүтүн сандар менен коддолгон. Graph модулунда ар бир түс үчүн атайын туруктуу (константа) ажыратылган болуп, алар текст абалында иштетиле турган түстөр менен бир түрдүү (экран менен иштөө темасындагы түстөр жадыбалына кара).

Graph модулунда 80 ге жакын оператор жана функциялар бар. Төмөндө алардын кээ бирлери менен таанышылат.

PutPixel(X,Y,Түс) процедурасы экрандын (X,Y) координаталуу чекитин «Түс» параметри менен аныкталган түскө боёйт. Мисалы, PutPixel(400,200,Red) процедурасы экрандын (400,200) координатасына тиешелүү жайда кызыл түстүү чекит жайгаштырат (86-беттеги сүрөткө кара).

GetPixel(X,Y) функциясы экрандын (X,Y) координаталуу чекити кандай түстө экендигин аныктайт. Мисалы, Түс – бүтүн типтүү өзгөрүүчү болсо, Түс:= GetPixel(40,50); оператору аткарылгандан соң Түс өзгөрүүчү (40,50) координаталуу чекит түсүнүн маанисине тең болот.

GetMaxX жана **GetMaxY** функциялары тиешелүү түрдө экрандын горизонтал жана вертикал багыт боюнча максималдуу координатасын аныктайт. Бул функциялар компьютердин графикалык адаптери жана иштетиле турган графикалык абалга байланыштуу болбогон программалар түзүүдө пайдалуу.

1-мисал. Чекиттердин жардамында экрандын (0,0) чекитин (639,479) чекити менен туташтыруучу кызыл түстүү кесинди сызуу программасын түз.

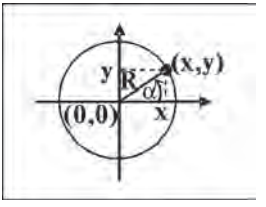
Чыгаруу. Төмөнкүчө бүтүн өзгөрүүчүлөрдүн мааниси менен алынат: bx:=0; ox:= GetMaxX; (ox:=639; сыяктуу алуу мүмкүн). Текшерип көрүшүң мүмкүн, [bx, ox] аралыкта y=[GetMaxY·x/GetMaxX] бүтүн маанилүү сызыктуу функциянын башталгыч мааниси 0 жана акыркы мааниси 479 болот (GetMaxY ордуна 479 жазуу да мүмкүн). Эми параметрлүү кайталоо операторунун жардамында түзүлгөн төмөнкү программа мисалдын чечимин берет.

```
Uses Graph;
Var   gd, gm:integer;
        bx, y, ox: Integer; x: LongInt;
Begin
        Gd := 0; InitGraph(Gd, Gm, '');
        bx:=0; ox:= GetMaxX;
For x:= bx to ox do                    begin
y:= trunc(GetMaxY*x/GetMaxX); putpixel(x, y, red);          end;
        Readln; CloseGraph;
End.
```

Эмне үчүн *x* өзгөрүүчү LongInt түрүндө алынгандыгын ойлоп көр!

2-мисал. Чекиттердин жардамында экрандын борборунда *R* радиустуу сары түстүү айлана сызуу программасын түз.

Чыгаруу. Экрандын борборун аныктай турган бүтүн өзгөрүүчүлөр киритип алынат: борборX:=trunc(GetMaxX/2); борборY:=trunc(GetMaxY/2); (же борборX:=320; жана борборY:=240). Математикадан белгилүү болгондой, мүнөздөө боюнча $\sin\alpha = \frac{y}{R}$ жана $\cos\alpha = \frac{x}{R}$. Мындан, борбору (0,0) чекитте болгон R радиустуу айлананын чекиттерин төмөнкү формулалардын жуптугу аркылуу аныктоо мүмкүн экендиги келип чыгат: $x = r \cdot \cos(\alpha)$, $y = r \cdot \sin(\alpha)$, мында α бурч 0 дөн 2 ге чейин өзгөрөт. Программада айлананын чекиттерин тыгызырак сызуу үчүн α бурч өзгөрүшү 0,01 кадам менен алынат. Ушулардын негизинде төмөнкү программа түзүлөт.




```
Uses Graph;
Var   gd, gm:integer;
        x, y, борборX, борборY: Integer; R, alfa: real;
Begin
 Gd := 0; InitGraph(Gd, Gm, '');
 Write('R= '); ReadLn(R);
борборX:=trunc(GetMaxX/2); борборY:=trunc(GetMaxY/ 2); alfa:=0;
 while alfa<=2*pi do                    begin
 x:= борборX +trunc(R*cos(alfa));
 y:= борборY +trunc(R*sin(alfa));
 putpixel(x, y, 14); alfa:= alfa+0.01;
                        end;
Readln; CloseGraph;
End.
```

3-мисал. Экрандын борборун координатанын башы эсептеп чекиттердин жардамында $x \in [-5, 5]$ аралыкта көк түстө $y=x^2$ функция графигин сызуу программасын түз.

Чыгаруу. Бул милдет төмөнкүчө чечилет:

```
uses Graph;
var   gd,gm: Integer;
        борборX, бо рборY: integer; x, y: real;
begin
gd := 0; InitGraph(gd, gm, '');
борборX:=trunc(getmaxx/2); борборY:=trunc(getmaxy/2); x:=−5;
  while x<=5 do   begin
    y:=x*x; putpixel(trunc(10*x+борборX), trunc(−5*y+борборY),blue);
     x:=x+0.01;      end;
   Readln;     CloseGraph;
End.
```

Программада масштабды чоңойтуу үчүн *x* ти 10 го, у ти 5 ке көбөйтүлдү. Параболанын бутактары төмөндөн жогоруга багытталган болушу үчүн «−» белги коюлган.

Масштабды чоңойтуу жана белгини алып таштап программаны иштетип көрүү өзүнө жүктөлөт.

Суроо жана тапшырмалар

1. *Паскалдын Graph модулу кандай максатта колдонулат?*
2. *Графикалык абалда экрандагы чекиттин орду эмнеси менен аныкталат?*
3. *Экранды графикалык абалга өткөрүү үчүн Паскалда кандай көрсөтмө берилет?*
4. *Графикалык абалдан «чыгуу» үчүн кайсы оператор колдонулат?*
5. *PutPixel, GetPixel, GetMaxX жана GetMaxY функцияларынын милдетин айтып бер.*

Көнүгүүлөр

1. Экранды графикалык абалга өткөрүүчү жана Enter клавиши басылганда дагы кайра текст абалына кайтаруучу программа түз.

2. Экрандын төрт бурчунда сары түстүү чекит пайда кылуучу программа түз.

3. Чекиттердин жардамында экранды ортосунан бөлүүчү горизонтал сызык пайда кыл.

4. Random функциясынан пайдаланып түрдүү түстүү чекиттерди пайда кылуу программасын түз.

5. $x \in [-10, 10]$ аралыкта $y=3x+5$ функциясынын графигин сызуучу программа түз.

41-сабак. Паскалда экранды графикалык абалга өткөрүү темасын кайталоо

1. Жактарынын түсү түрдүүчө болгон туура төрт бурчтук сызуу программасын түз.

2. Графикалык координатасы менен берилген чекиттин учтарынын графикалык координаталары аркылуу берилген түз сызыкка тиешелүү же тиешелүү эместигин аныктоочу программа түз (көмөк: чекиттин түсү түз сызыктын түсүнө тендигин аныктоо үчүн GetPixel функциясынан пайдалан).

3. Түрдүү түстө 15 даана параллель кесиндилер сызуучу программа түз (көмөк: кесинди координаталарын жана түстү ашыруу үчүн кайталоо операторунан пайдалан).

4. Random функциясынын жардамында «жылдыздуу асман» көрүнүшүн пайда кылуу программасын түз.

5. Экрандын борборунан өтүүчү координаталар огу, тиешелүү жайда координаталар огунун атын жазуучу жана $x \in [-7, 7]$ аралыкта $y = |x|$ функциясынын графигин сызуучу программа түз.

6*. Ичи-ичине жайгашкан 7 даана айлана сызуучу программа түз (көмөк: радиусту узартуу үчүн кайталоо операторунан пайдалан).

7*. 7 жолу өчүп-жана турган айлана сызуучу программа түз (көмөк: айлана сыз жана жараянды акырындатуу үчүн бош кайталоо жаса, мурдагы айлананы фон түсүндө сыз жана жараянды акырындатуу үчүн бош кайталоо жаса, кайталоону 7 жолу аткар).

42-сабак. Паскалда фигуралар сызуу операторлору

Буга чейин көргөнүбүздөй, чекиттердин жардамында да, оңой болбосо да, түрдүү фигуралар пайда кылуу мүмкүн. Бирок Паскалдын даяр фигуралар пайда кылуучу операторлору да бар. Бул операторлордун жардамында мурдатан тандалган кандайдыр бир түс менен түрдүү фигураларды сызуу мүмкүн.

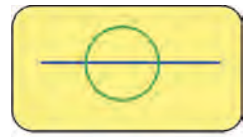
Паскалда сызуу түсүн тандоо үчүн **SetColor(түс)**, ал эми фон түсүн тандоо үчүн **SetBkColor** операторлорунан пайдаланылат. Мында **түс** – бүтүн сандуу өзгөрүүчү же туруктуу болуп, ал тандалган түстүн кодун ага тиешелүү константанын атын туюнтат. Тандалган түс **учурдагы түс** дейилет. Эгерде баштап кандайдыр бир түс тандалбаган болсо, анда ак түс учурдагы деп эсептелет.

Line(X1,Y1,X2,Y2) оператору экрандын (X1,Y1) координаталуу чекити менен (X2,Y2) координаталуу чекитин бириктирүүчү учурдагы түстүү кесинди чиет.

Circle(X,Y,R) оператору борбору (X,Y) чекитте жана радиусу R ге тең айлана чиет. Эми айланалар менен байланышкан маселелерди оңой эле чечүү мүмкүн.

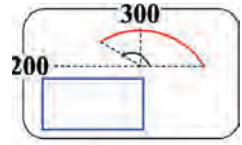
1-мисал. Учтарынын координаталары (10,200) жана (630,200) болгон көк түстүү кесинди жана борбору (300,200) координаталуу чекитте жана радиусу 100 болгон жашыл түстүү айлана сыз. Фон сары түстө болсун.

```
Uses Graph;
Var gd, gm : Integer;
Begin  Gd:=Detect; InitGraph(gd, gm,'');
       SetBkColor(Yellow);
       Setcolor(Blue);     Line(10,200,630,200);
       Setcolor(Green);    Circle(300,200,100);
ReadLn;    CloseGraph;
End.
```



Ellipse(X,Y,BB,OB,XR,YR) оператору борбору (*XY*) чекитте, *x* жана *y* октору боюнча радиустары тиешелүү түрдө *XR* жана *YR* ге тең эллипстин *BB* бурчунан *OB* бурчуна чейин болгон жааны чиет. Бурч градус чен бирдигинде берилет. *XR*=*YR* болсо, айлананын жаасы чийилет.

Rectangle(X1,Y1,X2,Y2) оператору экранда жогору сол бурчу (X1,Y1) координаталуу жана ылдыйкы оң бурчу (X2,Y2) координаталуу чекитте болгон туура төрт бурчтук чиет.



2-мисал. Борбору (300,200) координаталуу чекитте, *x* огу боюнча радиусу 100, у огу боюнча радиусу 50, башталгыч бурчу 0°, акыркы бурчу 135° болгон кызыл түстүү жаа жана сол жогорку жана оң ылдыйкы учтары тиешелүү түрдө (10,220) жана (300,400) координаталуу чекиттерде болгон көк түстүү туура төрт бурчтук сыз.

```
Uses Graph;
Var gd, gm : Integer;
Begin
      Gd:=Detect; InitGraph(gd,gm, ''); Setcolor(4);
      Ellipse(300,200,0,135,100,50);
   Setcolor(1);  Rectangle(10,220,300,400);
      Readln;    CloseGraph;
End.
```

DrawPoly(BS,KM) оператору сынык сызык чиет. BS – сынык сызыктын сынуу чекиттеринин саны, KM – сынык сызыктын сынуу чекиттеринин координаталары берилген массивдин аты. Эгерде сынык сызыктын башталгыч чекити менен акыркы чекитинин координатасы үстү-үстүнө түшсө, жабык тармак – көп бурчтук пайда болот.

Паскалдын түрдүү түс жана усулда боёлгон фигураларды чийүү операторлору да бар. Бул фигуралардын чек ара сызыктарынын түсү биз жогоруда

көргөн **SetColor** оператору менен тандалат. Алардын ичин боёо үчүн түс жана усул Паскалдын **SetFillStyle(усул,түс)**; операторунун жардамында белгиленет, мында **түс** – тандалып жаткан түстүн коду, **усул** – боёо усулу.

Паскалда фигуралардын ичин тандалган түс менен бир канча усулда боёо мүмкүн. Боёо усулдары кудум түстөр сыяктуу бүтүн сандар менен коддолгон. Graph модулунда ар бир боёо усулуна өзүнчө константалар да ажыратылган. Төмөнкү жадыбалда боёо усулдары, алардын коддору жана тиешелүү константалардын аттары келтирилген:

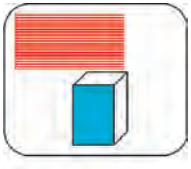
Боёо усулу	Коду	Константа аты
Фондун түсү менен боёо	0	EmptyFill
Берилген түс менен боёо	1	SolidFill
Калың горизонтал сызыктар менен толтуруу	2	LineFill
Ичке оңго жантайган сызыктар менен толтуруу	3	LtSlashFill
Калың оңго жантайган сызыктар менен толтуруу	4	SlashFill
Калың солго жантайган сызыктар менен толтуруу	5	BkSlashFill
Ичке солго жантайган сызыктар менен толтуруу	6	LtBkSlashFill
Чакмак сызыктар менен толтуруу	7	HatchFill
Жантайган чакмак сызыктар менен толтуруу	8	XHatchFill
Коюу жантайган сызыктар менен толтуруу	9	InterLeaveFill
Сейрек чекиттер менен толтуруу	10	WideDotFill
Коюу чекиттер менен толтуруу	11	CloseDotFill
Пайдалануучу белгилеген усулда боёо	12	UserFill

Bar(X1,Y1,X2,Y2) оператору экранда жогорку сол бурчу (X1,Y1) жана төмөнкү оң бурчу (X2,Y2) координаталуу чекиттерде болгон, ичи учурдагы түс жана усулда боёлгон туура төрт бурчтук сызылат.

Bar3D(X1,Y1,X2,Y2,a,b) процедурасы учурдагы түс жана усулда боёлгон параллелепипед сызат. Бул жерде **a** – параллелепипеддин капталынын узундугу, **b** болсо логикалык туюнтма болуп, анын мааниси «чын» болсо параллелепипеддин жогорку чокусу сызылат, «жалган» болсо сызылбайт.

3-мисал. Ичи кызыл түстүү калың горизонтал сызыктар менен толтурулган туура төрт бурчтук жана көгүлтүр параллелепипед сыз.

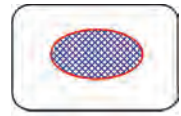
```
Uses Graph;
Var gd, gm : Integer;
Begin
        Gd:=Detect; InitGraph(gd,gm, '');
        SetFillStyle(2, 4);    Bar(10,10,400,200);
```



```
        SetFillStyle(1,9);    Bar3D(100,200,350,400,50,True);
        ReadLn;    CloseGraph;
    End.
```

FillEllipse(X,Y,XR,YR) операторy борбору (X,Y) координаталуу чекитте, X жана Y октору боюнча радиустары (узуну жана туурасы) тиешелүү түрдө XR жана YR ге тең, учурдагы түс жана усулда боёлгон эллипс сызат.

4-мисал. Ичи көк түстүү жантайган чакмак сызыктар менен толтурулган эллипс сыз.

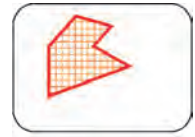


```
Uses Graph;
Var gd, gm : Integer;
Begin
  Gd:=Detect;InitGraph(gd,gm, '); SetColor(Red);
  SetFillStyle(8,1);  FillЭллипс(320,240,200,100);
  ReadLn;    CloseGraph;
End.
```

FillPoly(BS,KM) оператору учурдагы түс жана усулда боёлгон көп бурчтук сызат. BS – көп бурчтуктун бурчтарынын саны, KM – көп бурчтуктун чокуларынын координаталары берилген массив. Бул оператор DrawPoly операторунан айырмалуу түрдө, көп бурчтун биринчи чокусу менен акыркы чокусун өзү бириктирип алат.

5-мисал. Чокуларынын координаталары *KM* массив аркылуу берилген жана ичи кызыл түстүү чакмак сызыктар менен толтурулган алты бурчтук сыз.

Чыгаруу. Чокуларынын саны 7 даана болгон сынык сызык чийүү операторунан пайдаланып, 7-учунун координатасы 1-учунун координатасы менен дал келгендей берилсе, алты бурчтук пайда болот.



```
Uses Graph;
Const bs=6;
Var  gd, gm : Integer;
  km : Array[1..bs,1..2] of Integer;
Begin gd:=0; InitGraph(gd,gm, ');  Setcolor(4);
 SetFillStyle(7,4); {усул жана түс тандалды}
 км[1,1]:=300;        км[1,2]:=10;
 км[4,1]:=400;        км[4,2]:=190;
 км[2,1]:=200;        км[2,2]:=80;
 км[5,1]:=300;        км[5,2]:=80;
 км[3,1]:=200;        км[3,2]:=200;   км[6,1]:=400;   км[6,2]:=40;
FillPoly(bs,км); {эгерде DrawPoly(bs,км); жазылса ичи
 бош алты бурчтук чийилет}
End.
```

Суроо жана тапшырмалар

1. Экранда кандайдыр бир фигура чийүүдө анын түсү кайсы оператор жардамында тандалат?

2. *Паскалда кесинди чийүү мүмкүнчүлүгүн иш жүзүндө көрсөтүп бер.*
3. *Айлана кайсы оператордун жардамында чийилет?*
4. *Эллипс операторунун жардамында кандай фигуралар чийүү мүмкүн?*
5. *Туура төрт бурчтук чийүү операторунда x1, y1, x2 жана y2 лер эмнени билдирет?*
6. *DrawPoly операторунун жардамында кандай фигуралар чийүү мүмкүн?*
7. *Фондун түсүн өзгөртүүнү иш жүзүндө көрсөт.*
8. *SetFillStyle жардамында кандай фигуралар чийүү мүмкүн?*
9. *Ичи боёлбогон жана боёлгон көп бурчтук чийүүнүн усулдарын түшүндүр.*

Көнүгүүлөр

1. Экрандын ортосунан өтүүчү горизонтал жана вертикал сызык сызуучу программа түз.
2. Экрандын ортосунда радиусу 100 дөн кичине сары түстүү 4 даана айлана чийүүчү программа түз.
3. Экранды сары горизонтал сызыктар менен толтур.
4. Светофордун сүрөтүн чийүүчү программа түз.
5. Кызыл түстүү туюк беш бурчтук чийүүчү программа түз.

43-сабак. Паскалда фигуралар сызуу операторлору темасын кайталоо

1. Өзбекстандын желегин сызуучу программа түз.
2. Экрандын төрт бурчунда туурасы 60 жана узуну 40 ка тең кызыл түстүү төрт бурчтуктар сызуучу программа түз.
3. Экранды тең төрт бөлүккө бөлүп, аларды ылайыктуу түрдө кызыл, сары, жашыл жана көк түстөргө боёочу программа түз.
4. Экрандын ортосунда радиусу 100 гө тең болгон сары түстүү тегерек сызуучу программа түз.
5. Ай жана жылдыздар сүрөттөлгөн түнкү асман көрүнүшүн сызуучу программа түз.
6. Деңиз бойлой нур таратып жаткан Күндүн сүрөтүн сызуучу программа түз. Деңизди сызууда жаа сызуу операторунан пайдалан.
7. Ичи сейрек кызыл чекиттер менен толтурулган туюк алты бурчтук сызуучу программа түз.
8*. 12 түрдүү боёо усулун көрсөтүүчү 40x40 өлчөмдүү 12 даана квадрат сызуучу программа түз.
9*. Светофор жарыктарын удаалаш күйгүзө турган светофордун сүрөтүн сызуучу программа түз.

44-сабак. Файлдар менен иштөө

Биз киритүү жана чыгаруу операторлору менен тааныштык. Киритүү оператору жардамында маалыматтарды клавиатура аркылуу киритүү мүмкүн болсо, чыгаруу оператору маалыматтарды экранга чыгарат экен. Кээде чыгарылышы зарыл болгон маалыматтар экранга батпай кала турган маселелер да учурайт. Ал эми кээ бир маселелердин натыйжаларын кийинчерээк пайдалануу үчүн сактап коюу зарыл болот. Мындай абалдарда керектүү маалыматтарды файл көрүнүшүндө сактоо максатка ылайык болот. Сен файлдар жана алардын форматтары (тексттүү, графикалык ж. б.) жөнүндө жетиштүү маалыматка ээсиң. Паскаль программалоо тили түрдүү форматтуу файлдар менен иштөө мүмкүнчүлүгүн берет. Паскалда файлдар менен иштөө үчүн атайын **файл түрүндөгү өзгөрүүчүлөр** (файлдуу өзгөрүүчүлөр) киритилген. Файлдуу өзгөрүүчүлөр аркылуу сырткы эстутумда жайгашкан файлдар туюнтулат. Сандуу өзгөрүүчүнүн мааниси сан, жолчолуу өзгөрүүчүнүн мааниси жолчо болгону сыяктуу, файлдуу өзгөрүүчүнүн мааниси файл болот. Файлдуу өзгөрүүчүлөр да мүнөздөлүшү шарт.

Биз төмөндө бир гана тексттүү файлдар менен иштөөнү көрөбүз. Тексттүү файлдар ар түрдүү узундуктагы жолчолордон турган болот. Мында жолчонун узундугу 256 даана белгиден ашпастыгы зарыл. Тексттүү файлдарды туюнтуучу өзгөрүүчүлөр Паскалдын **text** кызматчы сөзү аркылуу мүнөздөлөт.

Мисалы,

var fтекст : Text; {fтекст – тексттүү файл}

Файлдар менен иштөө үчүн Паскалда төмөнкү амалдарды аткаруу зарыл:

1. Файл түрүндөгү өзгөрүүчүнү сырткы эстутумдагы файл менен байланыштыруу;
2. Файлды «окуу» же «жазуу» үчүн ачуу;
3. Файлдагы маалыматтарды окуу же файлга маалыматтарды жазуу;
4. Файлды жабуу.

Файлдуу өзгөрүүчү сырткы эстутумдагы файл менен **Assign(f, <файл аты>);** операторунун жардамында байланышат, мында **f** – файл түрүндөгү өзгөрүүчү; **<файл аты>** – сырткы эстутумда жайгашкан файлдын атын туюнтуучу жолчолуу өзгөрүүчү же туруктуу. Эгерде файл бул каталогдо болбосо, анын **толук аты**

көрсөтүлөт. Мисалы, «Hooруз.txt» файлы «D» дисктин «Майрам» аттуу каталогунда жайгашкан болсо, анын толук аты төмөнкү көрүнүштө болот:

d:\ Майрам \ Hooруз.txt

Ушул файлды f – файлдуу өзгөрүүчү менен байланыштыруу үчүн Assign оператору төмөнкүчө жазылат:

Assign(f, 'd:\ Майрам \ Hooруз.txt');

Assign операторy файлдуу өзгөрүүчүнүн маанисин, б.а. сырткы эстутумдагы аныk бир файлды белгилейт. Бул файлга кайрылуу (андагы маалыматтарды окуу же ага маалыматтар жазуу) үчүн аны «ачуу» керек. Тексттүү файлды бир убакыттын өзүндө окуу үчүн да, жазуу үчүн да ачып болбойт. Окуу үчүн ачылган файлдан маалыматтарды окуу гана мүмкүн. Жазуу үчүн ачылган файлга болсо маалыматтарды жазуу гана мүмкүн.

Паскалда файлдарды «жазуу үчүн» төмөнкү эки түрдүү усулда ачуу мүмкүн:

1) жаңы файл пайда кылуу жана аны жазуу үчүн ачуу;

2) бар болгон файлды ага маалыматтар киритүүнү улантуу үчүн ачуу.

Rewrite(f) операторy сырткы эстутумда жаңы файл пайда кылат жана аны жазуу үчүн ачат. Ал Assign операторунун жардамында кандайдыр бир файл менен байланышкан болушу керек. Мисалы, Assign(f, 'сүмөлөк.txt'); Rewrite(f); операторлору аткарылгандан соң, бул каталогдо «сүмөлөк.txt» аттуу жаңы тексттүү файл пайда болот жана ага маалыматтар жазуу үчүн ачылат. Эгерде бул каталогдо кудум ушундай аттуу файл мурдатан бар болсо, эми ал ачылып ордуна жаңы файл жазылат.

Файлга керектүү маалыматтар жазып болгондон соң, сөзсүз жабылышы керек. Паскалда ачылган файл **Close(f)** операторунун жардамында жабылат. Бул оператор окуу үчүн ачылган файлдарды да, жазуу үчүн ачылган файлдарды да жабат. Сырткы эстутумдагы файлдуу өзгөрүүчүгө ылайыктуу файл Close процедурасы колдонулушунан баштап сөзсүз ачылган болушу керек.

Файлга маалыматтарды жазуу (маалыматтарды файлга чыгаруу) үчүн төмөнкү операторлор колдонулат:

Write(f, <чыгаруу тизмеси>); жана
WriteLn(f, <чыгаруу тизмеси>);

мында **f** – файлдуу өзгөрүүчү, **<чыгаруу тизмеси>** өз ара үтүр менен бөлүнгөн бир же бир канча өзгөрүүчү же туруктуу. Бул

операторлор <чыгаруу тизмеси>нде көрсөтүлгөн өзгөрүүчү жана туруктуулардын маанилерин файлга жазат.

1-мисал. Бул катологдо «Гимн.txt» аттуу файл пайда кылып, клавиатурадан киритилген гимнибиздин 4 катарын файлдын өзүнчө жолчолоруна жазып коюучу программа түз.

Чыгаруу. Киритилген маалыматтарды файлдын өзүнчө жолчолоруна жазуу үчүн WriteLn операторунан пайдаланылат.

```
Program Гимн1;
Var _гимн : Text;      жолчо : String; m:integer;
Begin
        Assign(_гимн, 'Гимн.txt'); Rewrite(_гимн);
        For m:=1 to 4 do              begin
        Write('Гимнибиздин ', m, 'катарын кирит: ');
ReadLn(жолчо);
        WriteLn(_гимн,  жолчо); end;
        Close(_гимн);
End.
```

Биз сырткы эстутумда жаңы файл пайда кылууну көрдүк. Кээде сырткы эстутумда сакталып жаткан кандайдыр бир тексттүү файлды улантуу б.а., ага жаңы маалыматтар кошуу зарыл болуп калат. Мында Rewrite операторунун ордуна **Append(f)** оператору колдонулат. Бул оператор көрсөтүлгөн сырткы эстутумдагы файлды «жазуу үчүн» ачат. Көрсөтүлгөн файл сырткы эстутумда болбосо, каталык болот. Демек, Append операторунун жардамында ачылышы зарыл болгон файл сырткы эстутумда сөзсүз бар болушу шарт.

2-мисал. 1-мисалда пайда кылынган «Гимн.txt» файлын ач жана гимнибиздин 4 катарын уландысына кайырманы кошуп коюучу программа түз.

Чыгаруу. Файлды улантуу үчүн аны Append оператор менен ачылат.

```
Program Гимн2;
Var  f : Text;       кайырма: String; m:integer;
Begin
        Assign(f, 'Гимн.txt'); Append(f);
        For m:=1 to 4 do              begin
        Write('Кайырманын ', m, 'катарын кирит: ');
ReadLn(кайырма);
        WriteLn(f, кайырма);              end;
        Close(f);
End.
```

Reset(f) оператору дисктеги файлдарды окуу үчүн ачат. Ал Assign оператору жардамында сырткы эстутумдагы кандайдыр бир файлга байланышкан болушу керек. Көрсөтүлгөн файл сырткы эстутумда болбосо, каталык болот.

Окуу үчүн ачылган файлдан маалыматтар төмөнкү операторлордун жардамында окулат:

Read(f, <өзгөрүүчүлөр тизмеси>); жана
ReadLn(f, <өзгөрүүчүлөр тизмеси>);

Мында **f** – файлдуу өзгөрүүчү. <өзгөрүүчүлөр тизмеси> – бир же өз ара үтүр менен ажыратылган бир канча өзгөрүүчүлөр. Бул операторлор көрсөтүлгөн өзгөрүүчүлөрдүн маанилерин файлдан окуп алат.

Read оператору негизинен файлда сандуу маалыматтар жазылган болсо, б.а. файлдын жолчосу өз ара пробел менен ажыратылган сандардан турган болсо колдонулат. Read оператору бул сандардын ар бирин өзүнчө окуп алат. Бир жолчодогу маалыматтар бүткөндөн соң кийинки жолчого өтөт.

3-мисал. Жактарынын узундуктары «үч бурч.in» файлында берилген үч бурчтуктун аянтын эсептөө программасын түз. «үч бурч.in» файлы бир жолчодон туруп, анда үч бурчтуктун жактарынын узундуктарын туюнтуучу үч сан өз ара пробел менен ажыратып жазылган.

Чыгаруу. «үч бурч.in» файлындагы маалыматтарды окуу үчүн Read операторунан пайдаланылат. Үч бурчтуктун аянты Герондун формуласынан пайдаланып эсептелет.

```
Program Үч_бурчтуктун_аянты;
Var f : Text;      a, b, c, yp, s : Real;
Begin
        Assign(f, 'үч бурч.in');
        Reset(f);  {«үч бурч.in» файлы окуу үчүн ачылды}
        Read(f, a); Read(f, b); Read(f, c);
        {a,b,c лардын мааниле ри «үч бурч.in» файлынан окуп алынды}
        Close(f);  {«үч бурч.in» файлы жабылды}
        yp:=(a+b+c)/2; s:=sqr(yp*(yp–a)*(yp–b)*(yp–c));
        WriteLn('Үч бурчтуктун аянты=', s);
End.
```

ReadLn оператору файлдагы жолчону толук бойдон окуйт. Тексттүү файлдардан жолчолорун тартип менен кезекме-кезек окуу мүмкүн. Мисалы, файлдын 10-жолчосун окуу үчүн андан алдыңкы 9 жолчо сөзсүз окуп алынышы керек.

4-мисал. «класс.txt» файлында 9-класс окуучуларынын тизмеси (ар бир жолчодо бирден окуучунун фамилиясы) берилген. Ушул тизмедеги 12-окуучунун фамилиясын экранга чыгаруучу программа түз.

Чыгаруу: 12-окуучунун фамилиясы «класс.txt» файлынын 12-жолчосунда жазылган. Аны окуп алуу үчүн баштапкы 11 даана жолчону окуп алуу керек. Ал үчүн параметрлүү кайталоо операторунан пайдаланылат.

```
Program Класс;
var fio : Text;     i : Integer; фам : String;
Begin
        Assign(фио, 'класс.txt'); Reset(фио);
        For i:=1 to 11 Do ReadLn(фио, фам);
        ReadLn(фио, фам);  Close(фио);
```

```
        WriteLn('12-окуучунун фамилясы:', фам);
    End.
```

Ушул мисалда «класс.txt» файлында берилген бардык окуучулардын фамилияларын экранга чыгаруу талап кылынса, көйгөй келип чыгат. Анткени «класс.txt» файлы канча жолчодон тургандыгы белгисиз. Мындай абалдарда Паскалдын **Eof(f)** функциясы колдонулат. Eof – логикалык функция болуп, файлда окуу үчүн маалыматтар калбаган болсо «Чын», антпесе «Жалган» маанисин кабыл кылат.

5-мисал. «класс.txt» файлында 9-класс окуучуларынын тизмеси берилген. Тизмедеги бардык окуучулардын фамилияларын экранга чыгаруучу программа түз.

Чыгаруу. Шарт боюнча кайталоо оператору – While дан пайдаланылат.

```
Program Класс;
var фио : Text;     фам : String;
Begin
        Assign(фио, 'класс.txt'); Reset(фио);
        While Not(eof(фио)) Do  begin
        ReadLn(фио, фам);
        WriteLn(фам);           end;
        Close(фио);
End.
```

Программадагы кайталоо операторунун шарты – Not(eof(фио)) «жалган» маани кабыл алмайынча, б.а. eof(фио) функциясы «чын» маани кабыл алмайынча кайталануу уланат. Eof(фио) функциясы «чын» маани кабыл алышы менен, б.а. «класс.txt» файлында окуу үчүн маалыматтар аякташы менен кайталануу токтотулат. Зарыл болсо, бир программада бир канча файлды ачуу мүмкүн. Эгерде алар кезек менен, б.а. бириси жабылгандан кийин экинчиси ачыла турган болсо, бир файлдуу өзгөрүүчүдөн пайдалануу мүмкүн. Антпесе алардын ар бирине өзүнчө файлдуу өзгөрүүчү мүнөздөлүшү керек.

Суроо жана тапшырмалар

1. *Файл түрүндөгү өзгөрүүчү дегенде эмне түшүнүлөт?*
2. *Файлдуу өзгөрүүчү сырткы эстутумдагы файл менен кайсы оператордун жардамында байланышат?*
3. *Rewrite операторунун милдетин айтып бер.*
4. *Ачылган файлды жабуу үчүн кайсы оператор колдонулат?*
5. *Тексттүү файлга маалыматтарды жазуу үчүн кайсы операторлордон пайдаланылат?*
6. *Файлга жаңы маалыматтар кошуу үчүн аны кайсы операторлордун жардамында ачуу керек?*
7. *Маалыматтарды окуу үчүн файл кайсы операторлордун жардамында ачылат?*

8. *Тексттүү файлдан маалыматтарды окуу үчүн кайсы операторлордон пайдаланылат?*
9. *Read операторү менен ReadLn операторлорунун айырмасын айтып бер.*
10. *ReadLn операторунда бир канча өзгөрүүчү катышса, алар кандай жазылат?*
11. *Eof функциясынын милдетин айтып бер.*

Көнүгүүлөр

1. Апта күндөрүнүн аттарын киритип, аларды «АПТА.TXT» файлында сактап коё турган программа түз.

2. 1-көнүгүүдөгү «АПТА.TXT» файлын ачып, уландысынан апта күндөрүнүн орус тилиндеги аттарын жазуучу программа түз.

3. «АПТА.TXT» файлында берилген апта күндөрүнүн аттарын экранга чыгаруучу программа түз.

45-сабак. Файлдар менен иштөө темасын кайталоо

1. Классташтарындын фамилия жана аттарынан түзүлгөн «КЛАСС.TXT» аттуу тексттүү файл пайда кылуучу программа түз.

2. «класс.txt» файлында берилген 9-класс окуучуларынын фамилиялары ичинен «М» тамгасы менен баштала тургандарын экранга чыгаруучу программа түз.

3. «класс.txt» файлында берилген 9-класс окуучулары фамилиялары ичинен «Б» тамгасы менен баштала тургандарын бөлүп алып, алардан «бкласс.txt» файлын пайда кылуучу программа түз.

4*. $y = \sin^2 x$ функциясынын $[-\pi,\pi]$ аралыктагы маанилерин 0,01 кадам менен эсепте. Натыйжаларын «sinus.out» файлында сактап кой.

5*. «sinus.out» файлына түшүндүрмө кошуучу программа түз.

46-сабак. Процедура жана функциялар

Көбүнчө белгилүү бир амалдар жыйнагын программанын түрдүү бөлүгүндө кайталоого туура келет. Паскалда көп кайталана турган амалдар жыйнагын негизги программадан бөлүп алып, алардан өзүнчө блоктор – **процедура** жана **функциялар** түзүү мүмкүн. Ар бир ушундай түзүлгөн процедура жана функцияга сөзсүз **ат** берилет. Керектүү процедура же функцияга анын аты аркылуу кайрылуу мүмкүн. Процедура жана функциялардан туура пайдаланып түзүлгөн программа, адатта, жөнөкөй жана түшүнүктүү болот.

Процедуралар да, функциялар да белгилүү амалдар жыйнагынан турган болсо, алар бири-биринен эмнеси менен айырмаланат?

Функциялар, жалпысынан алганда, кандайдыр бир маанини эсептөөгө ылайыкташтырып түзүлөт. Жыйынтык натыйжада эсептелген маани функциянын атына өздөштүрүлөт.

Процедуралар болсо белгилүү бир амалдардын удаалаштыгын аткаруу максатында түзүлөт. Мында кандайдыр бир натыйжалык маани эсептелиши эмес, о.э. берилген амалдар удаалаштыгы аткарылышынын өзү мааниге ээ.

Мисалы, программада жактары берилген үч бурчтуктун аянтын бир канча жолу эсептөө керек болсо, функциядан пайдалануу максатка ылайык. Анткени мында ар жолу бир гана маани – үч бурчтуктун аянты эсептелет. Эгерде чокуларынын берилген жактары боюнча экранда бир канча үч бурчтук чийүү талап кылынган болсо, табигый түрдө, бир гана маанини эсептөөгө ылайыкташтырылган **функция** эмес, экранда үч бурчтук чийүү жараянын уюштуруучу процедура түзүү зарыл.

Процедура жана функцияларды аларга кайрылууда бериле турган белгилүү бир маанилерге байланыштуу кылып уюштуруу да мүмкүн. Бул маанилер параметрлер, ушундай түрдө уюштурулган процедура жана функциялар болсо **параметрлүү процедура** жана **функциялар** дейилет.

Процедура жана функциялар темалар менен башталат. Процедуранын темасы төмөнкү жалпы көрүнүшкө ээ:

Procedure <процедуранын аты> (параметрлер);

Функциянын темасы төмөнкү жалпы көрүнүшкө ээ:

Function <функциянын аты> (параметрлер) : <функциянын маанисинин түрү>;

Параметрлүү процедура жана функцияларда параметрлердин түрлөрү да мүнөздөлөт. Мисалы,

Function даража (a, n : Intеrer):Integer; же Procedure шифр (жолчо:String);

Көрүп турганындай, процедура жана функциялар окшош түзүлүшкө ээ. Алардын түзүлүшү программанын түзүлүшүнөн дээрлик айырмаланбайт. Негизги программада мүнөздөлгөн туруктуу жана өзгөрүүчүлөр **жалпы (глобал)** туруктуу жана өзгөрүүчүлөр дейилет. Алардан программанын каалаган бөлүгүндө, алсак, процедура жана функциялардын ичинде да пайдаланса болот. Кандайдыр бир процедура же функциянын ичинде

Процедуранын жалпы түзүлүшү:

```
Procedure <процедуранын аты>
      (параметрлер);
Label
    <белгилер>;
Const
  <константалар мүнөздөмөсү>;
Var
  <өзгөрүүчүлөр мүнөздөмөсү>;
«Ички» процедуралар менен
функциялар
begin
  <процедура денеси>
{программа}
end;
```

Функциянын жалпы түзүлүшү:

```
Function <функциянын  аты>
   (параметрлер):түрү;
Label
   <белгилер>;
Const
  <константалар мүнөздөмөсү>;
Var
  <өзгөрүүчүлөр мүнөздөмөсү>;
«Ички» процедуралар менен
функциялар;
begin
  <функция денеси>
{программа}
end;
```

мүнөздөлгөн туруктуу жана өзгөрүүчүлөр **жергиликтүү (локал)** туруктуу жана өзгөрүүчүлөр дейилет. Алардан бир гана өзү мүнөздөлгөн процедура же функциянын ичинде пайдалануу мүмкүн. Паскалда жалпы жана жергиликтүү өзгөрүүчү бир түрдүү атка ээ болушу да мүмкүн. Мында жергиликтүү өзгөрүүчү мүнөздөлгөн процедура же функциянын ичинде жергиликтүү өзгөрүүчүнүн мааниси, башка процедура жана функцияларда жана негизги программада жалпы өзгөрүүчүнүн мааниси эсепке алынат.

1-мисал. [20, 83], [178, 391], [211, 746] аралыктардагы бардык бүтүн сандардын суммасын табуу программасын түз.

Чыгаруу. Берилген үч аралыктагы бүтүн сандардын суммасын эсептөө үчүн параметрлүү кайталоо операторунан үч жолу пайдалануу туура келет. Мисалы берилген аралыктын башталгыч жана акыркы маанилерин параметр катары алып, ушул аралыктагы бүтүн сандардын суммасын эсептөөчү функциядан пайдаланып да чыгаруу мүмкүн.

```
Program Сумма;
Var  i, s, s1, s2, s3 : Integer;
Function Сум(n1, n2:Integer):Integer;
Begin
        s:=0;   For i:=n1 To n2 Do s:=s+i;   Сум:=s;
End;
Begin
        S1:= Сум(20,83); s2 := Сум(178,391); s3:=Сум(211,746);
        S:=s1+ s2 + s3;   WriteLn('S= ', s)
End.
```

2-мисал. Үч бурчтуктун берилген жактары боюнча, анын бийиктиктерин аныктоочу программа түз.

```
Program Үч бурчтуктун_бийиктиги;
```

```
        Var a, b, c, ha, hb, hc: real;
        Function  H_UB(a, b, c: real): real;    { a, b, c – Үч бурчтуктун
        жактары }
        Var yp, s: real;
    Begin
        yp:= (a+b+c)/2;   {Периметрдин жарымын эсептөө }
        s:= Sqrt(yp*(yp–a)*(yp–b)*(yp–c));   {Аянтты эсептөө}
        H_UB:= 2*s/a;   {Функцияга маани берилди }
    End;
    Begin
        Write('Үч бурчтук жактары (a,b,c) киритилсин '); ReadLn(a,b,c);
        ooба:= H_UB(a, b, c);   hb:= H_UB(b, a, c);   hc:= H_UB(c,
        b, a);
        WriteLn('Үч бурчтуктун бийиктиктери: ');
        WriteLn('ooба= ', ooба:10:4, 'hb= ', hb:10:4, 'hc= ', hc:10:4);
        ReadLn
    End.
```

3-мисал. Үч үч бурчтук чокуларынын төмөнкү координаталары берилген:

1) (120,20), (80,170), (140,150); 2) (200,97), (500,156), (210,180);
3) (300,190), (200,390), (415,222).

Ушул үч бурчтуктарды тиешелүү түрдө кызыл, сары жана жашыл түстөрдө чийүү программасын түз.

```
    Uses Graph;
        var  gd, gm:Integer;
    Procedure Үч бурчтук(x1,y1,x2,y2,x3,y3,col:Integer);
    begin
    SetColor(col);   Line(x1,y1,x2,y2);   Line(x2,y2,x3,y3);
    Line(x3,y3,x1,y1);
    end;
    Begin  gd:=0;  InitGraph(gd,gm, '');
    Үч бурчтук(120,20,80,170,140,150,4);
Үч бурчтук(200,97,500,156,210,180,14);
     Үч бурчтук(300,190,200,390,415,222,2); ReadLn;
  CloseGraph;
  End.
```

Жогоруда келтирилген программаларга көңүл буруп, төмөнкү жыйынтыкка келүү мүмкүн: процедура жана функцияларга кайрылуу убагында функциянын аты сөзсүз кандайдыр бир оператордун курамында, ал эми процедуранын аты өзүнчө (өз алдынча) жазылат экен.

Суроо жана тапшырмалар

1. Программада процедура жана функциялар кандай максатта колдонулат?

2. *Процедура жана функциялардын айырмасы эмнеде?*
3. *Параметрлүү процедура жана функциялар жөнүндө айтып бер.*
4. *Процедуранын жалпы түзүлүшүн туюнтуп бер.*
5. *Функциянын жалпы түзүлүшүн туюнтуп бер.*
6. *Программадагы жалпы жана жергиликтүү өзгөрүүчүлөр жөнүндө айтып бер.*
7. *Функциянын ордуна качан процедураны колдонуу мүмкүн?*

Көнүгүүлөр

1. Үч туура төрт бурчтуктун ар биринин бирден диагоналынын чокуларынын координаталары берилген: 1) 20,20 жана 80,200; 2) 200,97 жана 500,156; 3) 300,120 жана 400,420. Ушул туура төрт бурчтуктарды тиешелүү түрдө кызыл, сары жана жашыл түстөрдө чийүү программасын түз.

2. Берилген n натуралдык сан үчүн $S=1\cdot5+2\cdot6+3\cdot7+...+n\cdot(n+4)$ сумманы эсептөө программасын процедуранын жардамында түз.

3. Үч сандан чоңун табуу программасын түз. Ал үчүн эки сандан чоңун табуу функциясын түзүп, андан пайдалан.

47-сабак. Процедура жана функциялар темасын кайталоо

1. Берилген тексттеги 'a' белгини 'g' белгиге, 'm' белгини 's' белгиге, 'f' белгини 'h' белгиге алмаштыруучу программаны процедураны жардамында түз.

2. $y = x^5 + 3x$ функциянын маанисин x тин -9, -5, -2, 2, 5, 7 маанилеринде эсептөө программасын түз. Даражаны көбөйтүү аркылуу эсептөө үчүн функция түзүп ал.

4. Процедуранын жардамында экрандын ортосунда ичи-ичине жайгашкан түрдүү түстүү 15 даана айлана чийүүчү программа түз.

48–49-сабак. Кайталоого тиешелүү тапшырмалар

1. Жагынын узундугу a га тең квадрат жана ага ичтен сызылган айлана чийүү программасы түзүлсүн. a нын мааниси клавиатурадан киритилсин.

2. Радиусу R ге тең айлана жана ага сырттан сызылган квадрат чийүү программасы түзүлсүн. R дин мааниси клавиатурадан киритилсин.

3. Жактарынын узундугу a жана b болгон туура төрт бурчтук жана ага ичтен сызылган эллипс чийүү программасы түзүлсүн. a жана b нын мааниси клавиатурадан киритилсин.

4. Көк фондуу экранды аралыгы 20 пиксел болгон вертикал ак сызыктар менен толтур.

5. Ар түрдүү түстүү 15 даана тегерек сыз.

6. Столдун сүрөтүн сыз.

7. Үйдүн сүрөтүн сыз.

III БӨЛҮМ WEB-БАРАКЧА ДАЯРДОО

50-сабак. HTML жөнүндө түшүнүк

Интернеттин WWW кызматы, негизинен, web-сайт же web-баракчаларга байланыштуу экен, «Web-баракчалар кандай даярдалат?» деп суроо беришиң табигый. Web-баракчалар атайын программалардын негизинде даярдалат. Мисалы, Microsoft FrontPage, Macromedia HomeSite, Adobe Dreamweaver сыяктуу редакторлор, PHP, ASP, JavaScript сыяктуу сервер скрипттери (сценарийлер тили), XML, HTML жана башкалар. Бул программалардын бардыгы HTML (Hypertext Markup Language – Гипертексттүү маркерлөө тили) тилине негизделет. HTML программалоо тили эсептелбейт. Бул тилде документ (web-баракча) даярдоо үчүн Windowstун «Блокнот» сыяктуу жөнөкөй текст редактору жетиштүү.

HTML тилинин буйруктары «<» жана «>» белгилери арасына жазылат жана **дескриптор** (англ. мүнөздөөчү) же **тег** (англ. tag – жарлык, белги) деп аталат. Мисалы, <HTML> жазуусу HTML тилиндеги документтин башталышын билдирет. Тегдер латын алфавити тамгалары жардамында жазылат, мында тамгалар жогорку же төмөнкү регистрде жазылышы айырмаланбайт, б.а. <HTML> менен <html> бирдей тег болушат. Жалпысынан, тегдер эки түргө бөлүнөт:

1. Жуп тегдер (же контейнер-тегдер): <B> теги үчүн </B> теги бар болуп, биринчиси кайсы бир амалдын башталышын билдирсе, экинчиси ушул амал аяктаганын билдирет.

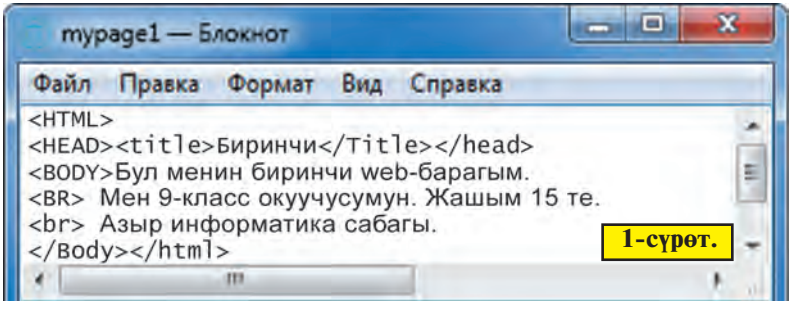
2. Жуп эмес тегдер: <D> көрүнүштөгү тег ачылат, жабылышы шарт эмес, мисалы, өзүнөн кийинки текстти жаңы жолчого өткөрүүчү
 теги сыяктуу.

HTML-документ – «html» же **«htm»** кеңейтмелүү тексттүү файл болуп, ал жөнөкөй текст редакторунда тегдерди колдонуп жазылган текст. HTML-документ <HTML> жана </HTML> тегдеринин арасында жазылган болот. HTML-документ эстутумга жүктөлсө, ал экранда web-браузер жардамында web-баракча көрүнүшүндө көрүнөт.

HTML-документ, адатта, эки бөлүмдөн турат. Биринчиси **HEAD** (англ. Башкы бөлүк же тема) бөлүмү болуп ал <HEAD> жана </HEAD> тегдери арасында жайгашат. Экинчиси **BODY** (дене) бөлүмүндө документтин мазмуну көрүнөт жана ал <BODY> жана </BODY> тегдери арасында жайгашат. Эгерде HTML-документ фрейм-структураны туюнтушу (маалыматтар web-браузер терезесинде өзүнчө областтарда көрүнүшү) зарыл болсо, анда BODY бөлүмү ордуна FRAMESET (FRAME SET – структуралар (рамкалар) түзүлмөсү (жыйнагы), **<FRAMESET>** жуп теги жардамында) бөлүмү иштетилет. Адатта, HTML-документте <HEAD> жана <BODY> жуп тегдерин жазуу сунуш кылынат, бирок милдеттүү эмес.

Web-баракчага киритилиши зарыл болгон дагы бир элемент – web-баракчанын аты болуп, ат киритүү үчүн **<TITLE>** жуп теги колдонулат. Web-баракчада бул тег бир жолу иштетилет. Web-баракчанын аты web-браузердин тема жолчосунда берилип, web-баракчанын ичинде көрүнбөйт. Ошол себептүү аны web-баракчанын каалаган жайына (адатта, HEAD бөлүмүнө) жазылат. Web-баракчага каалаган ат, мисалы, өз атынды беришиң мүмкүн.

HTML тили тынымсыз өнүгүп барууда. Өз кезегинде web-браузерлер да жаңыланып турат. Азыркы күндө web-баракча даярдоо үчүн негизинен HTML-4 тилинен пайдаланылат. Анын кээ бир буйруктарын «эски» web-браузерлер (Internet Explorer-4 же Интернет Explorer-6) аткара албайт. Белгилүү болгондой, түрдүү web-браузерлер, мисалы, Internet Explorer, Opera, FireFox, Mozilla жана Netscape бир-биринен айырмаланат. Ошондуктан бир HTML-документ түрдүү web-браузерлерде айырмаланып көрүнүшү мүмкүн.



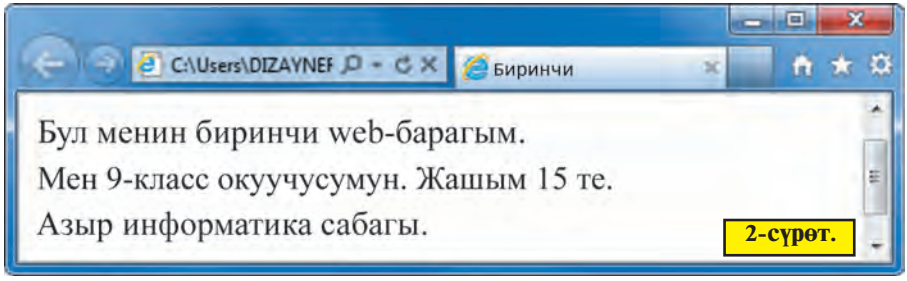
1-сүрөт.

Эң жөнөкөй web-баракча тексттен гана турган болот. Биз да web-баракча даярдоону текст жайгаштыруудан баштайбыз. Ал

үчүн Windowstун Блокнот текст редакторун ишке түшүрөбүз (башка текст редакторунан пайдаланса да болот). Web-баракча, адатта, тексттер сыяктуу темадан башталат. Бул биздин биринчи web-баракчабыз болгондуктан, ага «Бул менин биринчи web-баракчам» – деп, тема жазабыз. Ал үчүн Блокнот программасынын ишчи столуна 1-сүрөттөгү текст киритилет.

Бул тексте <HTML>, </HTML>, <HEAd>, </head>, <title>, </Title>, <BODY>, </Body> жана
 HTML тилинин тегдери болуп, <HTML> – web-баракчанын башталышын, </HTML> – web-баракча аяктаганын, <HEAd> – теманын бөлүмү башталганын, <HEAd> – тема бөлүмү аяктаганын, <title> – ат киритүү башталганын, </Title> – ат киритүү аяктаганын, <BODY> – маалыматтар бөлүмү башталганын, </BODY> – маалыматтар бөлүмү аяктаганын,
 – тексттин калганы жаңы жолчодон жазылышы керектигин билдирет.

Жогоруда киритилген текстти **«mypage1.html»** аты менен кайсы бир катологдо, мисалы, «Мои документы» папкасындагы «Гулноза» папкасында сактап коёбуз. Эми бул файлдын белгиси web-браузерге ылайыкташканын көрүү мүмкүн. Пайда кылынган web-баракча Internet Explorer браузеринде ачылганда 2-сүрөттөгү сыяктуу көрүнүштө болот:



2-сүрөт.

Киритилген текстүү файл жана web-баракчаны салыштырып көрүп, төмөнкү жыйынтыкка келебиз:

1) файлдын аты (жогорудагы мисалда, **mypage1**) web-баракчада көрүнбөйт;

2) web-баракчанын аты (жогорудагы мисалда, **Биринчи**) web-браузердин тема жолчосунда көрүнөт;

3) web-баракчанын текстинде атайын буйруксуз жазылган жолчо (жогорудагы мисалда, «web-барачым.» тексти да, «Жашым 15 те.» тексти да) өзүнөн мурдагы жолчого уланып кетет.

HTMLде киритилген тексттин web-браузердеги көрүнүшү дайыма эле биз күткөн натыйжаны бере бербейт. Аны биз каалаган көрүнүшкө келтирүү үчүн бир канча жолу редакциялоого туура келет. Ал үчүн Блокнот текст редакторун ишке түшүрүү, керектүү HTML-документти дисктен издеп табуу жана эстутумга жүктөө, аны редакциялоо жана кайра сактоо, web-браузерди ишке түшүрүү, редакцияланган HTML-документти дагы дисктен таап web-браузерде көрүү керек болот. Адатта, web-баракча биз ойлогон көрүнүшкө келиши үчүн жогоруда санап өтүлгөн иштерди бир канча жолу аткарууга туура келет. Бул иштерди кыйла ыңгайлуу усулдар менен да аткаруу мүмкүн:

1-усул. Internet Explorerде ачылган web-баракчаны редакциялоо зарыл болсо, чычканчаны ушул web-баракчанын үстүнө алып келип, оң топчусу басылат. Ачылган контекст-менюдан «HTML-кодун көрүү» Просмотр HTML-кода бөлүмү тандалса, web-баракчанын HTML-документи жүктөлгөн Блокнот текст редактору экранга чыгат. HTML-документти редакциялап, сактап коюлат. Internet Explorer инструменттер панелиндеги **«Жаңылануу»** ( ) топчусу басылса, маалыматтар майданындагы web-баракча жаңыланат (редакцияланган web-баракчага алмашат).

2-усул. Көрүнүш (Вид) менюсунан «HTML-кодун көрүү» бөлүмү тандалат жана 1-усулдагы сыяктуу редакцияланат.

Суроо жана тапшырмалар

1. *HTML тилинин буйруктары кандай аталат?*
2. *HTML-документ деген эмне? HTML-документ файлы атынын кеңейтмеси кандай?*
3. *Тегдердин түрлөрү жөнүндө айтып бер.*
4. *HTML-документ кандай тег менен башталат?*
5. *Web-баракчага ат берүүнү мисалдар аркылуу көрсөтүп бер.*
6. *Web-баракчанын аты web-браузердин каеринде көрүнөт?*
7. *Web-баракчанын аты HTML-документтин кайсы бөлүгүндө жазылышы мүмкүн?*

Көнүгүүлөр

1. «Менин Ата журтум» аттуу web-баракча үчүн текст даярда. Аны текст редакторунда пайда кылып, «Менин мекеним» аты менен сакта.

2. «Менин Мекеним» аттуу тексти web-документ көрүнүшүнө өткөр. Web-баракчаны web-браузерде ачып редакциялa.

3. «Биздин мектеп» темасында тексттүү жөнөкөй web-баракча даярда.

51-сабак. Web-баракчага текст киритүү

Web-баракчадагы текстте бир канча тема жарытылышы мүмкүн. Мында web-баракчада бир канча тема иштетүүгө туура келет. Мисалы, өздүк web-баракчаңа өзүң жөнүндө, кызыгууларың, жактырган фильмдериң жөнүндө маалымат киритсең, тиешелүү түрдө «Өзүм жөнүндө», «Кызыгууларым», «Жактырган кинофильмдерим» сыяктуу темалар коюшуң мүмкүн. Кээде, бир теманын текстин чоң тамгалар менен жазуу керек болсо, башкасын майдараак шрифтте жазуу зарыл болуп калат.

HTML тили 6 баскычтуу тема коюу мүмкүнчүлүгүн берет. Ал үчүн HTML тилинде төмөнкү жуп тегдер бар: **<H1>, <H2>, <H3>, <H4>, <H5>, <H6>.** Демек, бул тегдердин ар бири үчүн тиешелүү түрдө жабылуучу (аяктоочу) тегдер (</H1>, ...,</H6>) да бар («H» белгиси «Heading», б.а. англисче тема сөзүнүн биринчи тамгасы).

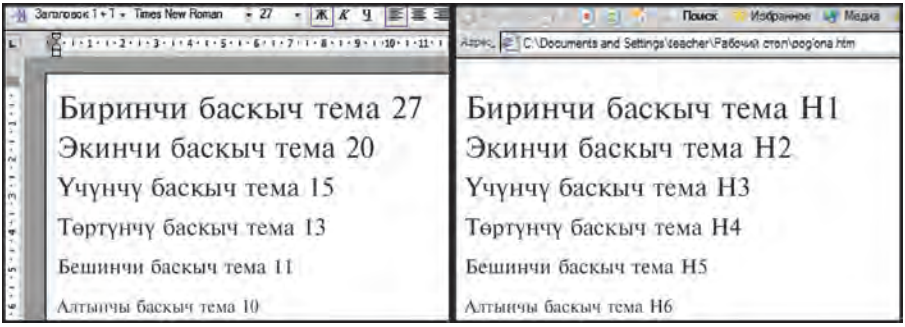
Файл Правка Формат Вид Справка

```
<HTML>
<h1> Биринчи баскыч тема H1 </h1>
<h2> Экинчи баскыч тема H2 </h2>
<h3> Үчүнчү баскыч тема H3 </h3>
<h4> Төртүнчү баскыч тема H4 </h>
<h5> Бешинчи баскыч тема H5 </h5>
<h6> Алтынчы баскыч тема H6 </h6>
</html>
```

MS Word программасында иштегенинде шрифт өлчөмү (мисалы, 27), шрифттин жазуу түрү (мисалы, Times New Roman), жазуу стили (мисалы, Тема1), түскө каныккандыгы (мис: калың) сыяктуу түшүнүктөр менен таанышкан элең. Ушуларды эсепке алып MS Word до даярдалган тексттеги темаларды HTML тилинде даярдалган web-баракчадагы темалар менен төмөнкүчө салыштыруу мүмкүн:

Web-баракчага текст киритүү текст редакторундагы сыяктуу ишке ашырылышы мүмкүн. HTML-документке киритилген текстти web-браузер маалыматтар майданынын өлчөмүнө ылайыкташтырып форматтап алат. Ушул себеп киритилген текст браузерде бир аз түрү өзгөргөн абалда көрүнүшү мүмкүн. Текст web-барак-

чада дал биз каалагандай жайгашышы үчүн HTML тилинде текст форматына таасир этүүчү атайын тегдер бар.



Web-баракчада абзацтарды белгилөө үчүн **<P>** жуп теги иштетилет. Бул тег абзац башталышында жазылат жана өзүнөн кийин жазылган текст алдында бош жолчо калтырат. Баштап айтып өтүлгөндөй, тексттин уландысын жаңы жолчого өткөрүү үчүн жуп эмес **
** тегинен пайдаланылат. Бул тегди текстте бош жолчо калтыруу үчүн да иштетүү мүмкүн.

HTML тегдери параметрлери менен жазылышы мүмкүн. Параметрлер тег атынан кийин пробел менен ажыратылат. Параметрлерге маани берилиши керек болсо, ал барабардык белгисинен кийин тырмакчанын ичинде же тырмакчасыз жазылат.

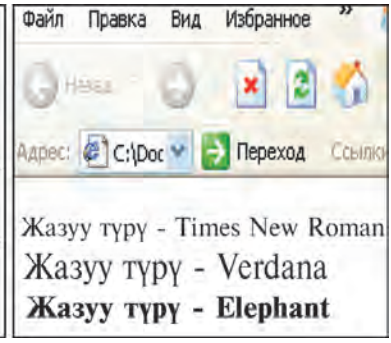
Шрифттин жазуу түрүн белгилөө үчүн **<FONT>** (шрифт) жуп теги **FACE** параметри менен колдонулат: **<FONT FACE= "жазуу түрүнүн аты">** текст </**FONT**>. Шрифттин жазуу түрүнө **Times New Roman, Verdana, Elephant** сыяктуулары кириши сага белгилүү. Бир web-баракчада бир канча жазуу түрүн колдонуу мүмкүн. Бирок кайсы бир web-браузер шрифттин керектүү жазуу түрүн көрсөтө албастыгы мүмкүн. Ошондуктан этияттан бул тег параметринин бир канча маанилерин үтүр менен бөлүп жазуу максатка ылайык болот:

<FONT FACE= "Times New Roman", "Arial Black", "Elephant">.

Web-браузер тег параметри маанилерин солдон баштап окуйт жана бири болбосо кийинкисин издейт. Төмөндө шрифттин жазуу түрүн колдонуу боюнча мисал келтирилген.

Кээде, тексттин кайсы бир бөлүгүнө көңүлдү буруу максатында кээ бир сөздөр кандайдыр бир өзгөчөлүк менен ажыратып көрсөтүлөт. MS Word документинде буга тексттеги сөздөрдү

```
<HTML>
<title>Шрифттин жазуу түрү</title>
<FONT FACE="Times New Roman">
Жазуу түрү - Times New Roman </Font>
<br><FONT FACE="Verdana","Arial Narrow">
Жазуу түрү - Verdana</Font>
<br><FONT FACE="Elephant","Bodoni MT Black">
Жазуу түрү - Elephant </Font>
</html>
```



калың, жантайма же асты сызылган көрүнүштө жазуу аркылуу жетишүү мүмкүн эле. HTML тилинде да ушундай мүмкүнчүлүк болуп, төмөнкү жуп тегдерден пайдаланылат:

<B> – калың шрифт (Bold)	**же ордуна**	**<Strong>**
<I> – жантайма шрифт (Italik)		**<Em>** же **<Cite>**
<U> – асты сызылган шрифт (Underline)		–

Жогоруда келтирилген тегдерди чогуу иштетип, web-баракчадагы текстти калың жана жантайма, жантайма жана асты сызылган жана башка көрүнүшкө келтирүү мүмкүн:

```
<HTML><H1>Тексттеги сөздөрдү ажыратып көрсөтүү</H1>
<p>Тексттин кайсы бир бөлүгүнө көңүлдү буруу максатында кээ бир
сөздөр ажыратып көрсөтүлөт.
<BR>Буга тексттеги белгилүү бир сөздөрдү <B>калың,</B> <I> жантайма
</I> же <U>астына сызылган </U>көрүнүштө жазуу аркылуу жетишүү
мүмкүн.
<BR>Тегдерди чогуу колдонуп, текстти <B><I>калың,</B><U> жантайма
жана астына сызылган,</I></U>жана башка көрүнүшкө келтирүү мүмкүн.
</HTML>
```

Тексттеги сөздөрдү ажыратып көрсөтүү

Тексттин кайсы бир бөлүгүнө көңүлдү буруу максатында кээ бир сөздөр ажыратып көрсөтүлөт.
Буга тексттеги белгилүү бир сөздөрдү **калың,** *жантайма* же астына сызылган көрүнүштө жазуу аркылуу жетишүү мүмкүн.
Тегдерди чогуу колдонуп, текстти ***калың жана жантайма,*** *жантайма жана астына сызылган* жана башка көрүнүшкө келтирүү мүмкүн.

MS Word программасында инструменттери менен текстти баракчада түрдүүчө (б.а, баракчанын сол, орто, оң

бөлүгүндө же баракча кендиги бойлоп) тегиздөө мүмкүн эле. HTML тилинде да текстти web-баракчада түрдүүчө жайгаштыруу мүмкүнчүлүгү бар жана бул милдет **<P>** жуп эмес тегине (же **<H1>, <H2>, <H3>, <H4>, <H5>, <H6>** жуп тегдерине) ALIGN (англ., тегиздөө) параметрин кошуу аркылуу ишке ашырылат:

<P ALIGN= ″жайгаштыруу параметри мааниси″>.

Жайгаштыруу параметринин мааниси ордуна ″**Left**″ (сол), ″**Right**″ (оң), ″**Center**″ (борбор) жана ″**Justify**″ (кеңдик) сөздөрүнөн бири жазылат. Абзацтарды web-баракчада жайгаштырууга мисал көрөбүз:

```
<html>
<h2>Текстти web-бараkта жайгаштыруу</h2>
<p align="left">Бул сап баракtын сол жагында жайгашkан.
<h5 align="right">Бул сап баракtын оң жагында жайгашkан.</h5>
<p align="center">Бул сап баракtын ортосунда жайгашkан.
<h4 align="justify">Бул абзац эки жакtан тегизделген. Сен текстти баракка жай-
гаштыруу <br>усулдары менен Microsoft Word текст процессору темасында
толук <br>таанышkансың. HTMLде бул аракеттерди аткаруу бир аз
<br>башкачараак ишке ашырылат. </html>
```

Текстти web-бараkта жайгаштыруу

Бул сап баракtын сол жагында жайгашkан.

Бул сап баракtын оң жагында жайгашkан.

Бул сап баракtын ортосунда жайгашkан.

Бул абзац эки жакtан тегизделген. Сен текстти баракка жайгаштыруу усулдары менен Microsoft Word текст процессору темасында толук таанышkансың. HTMLде бул аракеттерди аткаруу бир аз башкачараак ишке ашырылат.

Киритилген текст менен анын web-браузерде көрүнүшү бир аз айырмаланат. Мунун себеби, адатта web-браузерлер артыкча бош жайларды (пробелдерди) танат (таштап жиберет). Кээде, текст кандай киритилсе, web-баракчада да ушундай жайгашышы шарт болот. Мисалы, web-баракчага ыр жайгаштыруу же жөнөкөй белгилердин жардамында сүрөт чийүү керек болсо, текстти форматтоону web-браузерге тапшырып болбойт. Мындай абалдарда HTMLдин **<PRE>** жуп тегинен пайдаланылат. Бул тегдин жардамында текст web-баракчага HTML-документте кандай жазылган болсо, ушундай көрүнүштө чыгарылат. <PRE> теги кандай иштешин төмөнкү мисал туюнтат:

```
<html>
<pre>
Жаз!        Кандай сонун  мезгил!
  Жаз!        Кандай сонун  мезгил!
    Жаз!        Кандай сонун  мезгил!
</pre>
<P>
<br>Жаз!        Кандай сонун  мезгил!
<br>  Жаз!        Кандай сонун  мезгил!
        Жаз!        Кандай сонун  мезгил!
</html>
```

<PRE> жана </PRE> тегдеринин арасындагы текстке <P>,
 сыяктуу тегдерди колдонуу пайдасыз. Бул аралыкта бул тегдер web-браузер тарабынан кабыл алынбайт. Web-баракчага текстти <PRE> тегинен пайдаланып киритүү өтө ыңгайлуу, бирок бул тегди эч арга калбаган абалда гана иштетүү максатка ылайык. Анткени, web-браузер текстти маалыматтар майданынын өлчөмүнө ылайыктап форматтап алат. <PRE> теги колдонулган текст болсо web-браузер тарабынан форматталбайт жана анын маалыматтар майданына батпаган бөлүгү көрүнбөй калат.

Суроо жана тапшырмалар

1. *HTML-4 төкканча баскычтуу темаларды иштетүү мүмкүн?*
2. *Теманын баскычтары бири-биринен эмнеси менен айырмаланат?*
3. *HTML да абзац кайсы тегдин жардамында белгиленет?*
4. *HTML да кайсы тег текстти жаңы жолчого өткөрөт?*
5. *Текст калың, жантайма, асты сызылган шрифтте жазылышына мисалдар жаз.*
6. *Web-баракчада текстти тегиздөө усулдарына мисалдар жаз.*
7. *Web-браузер текстти форматтабасын үчүн кандай тег иштетилет?*

Көнүгүүлөр

1. Республикабыздын гимни тексти чагылдырылган web-баракча даярда. Анда жолчолордун шрифт жазуу түрүн түрдүүчө танда.

2. Мектебиң жайгашкан республика, облус, шаар, район, көчө аттары, тиешелүү түрдө, 1-, 2-, 3-, 4-, 5-баскыч темалары көрүнүшүндө чагылдырылган web-баракча даярда.

3*. «Биздин класс» аттуу web-баракча даярда. Анда түрдүү форматтоо тегдеринен пайдалан.

52-сабак. Web-баракчага текст киритүү темасын кайталоо

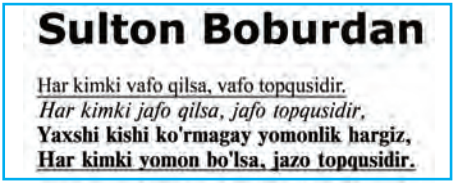
1. «Менин үй-бүлөм» аттуу web-баракча даярда. Анда түрдүү форматтоо тегдеринен сырткары белгилердин жардамында пайда кылынган жөнөкөй сүрөт (үй, арча жана башка) да болсун.

2. Предметтердин аталышы кызыгууна карай түрдүү баскыч темалары көрүнүшүндө берилген web-баракча даярда.

3*. «Биздин класс» аттуу web-баракчандагы бардык сөздөрдө ээ калың жана жантайма, баяндооч жантайма жана асты сызылган көрүнүштө болсун.

4*. «Улуу бабаларыбыз» аттуу web-баракча даярда. Анда баракчанын ортосунда жайгашкан бабаларыбыздын аты 1-баскыч темасында кылган иштери ондон тегизделип 3-баскыч темасында көрүнсүн.

5*. Көрүнүшү төмөнкүдөй «Рубаий» аттуу web-баракча даярда.



53-сабак. Шрифттин өлчөмү, түсү жана web-баракчанын фону

Мурдагы темаларда түзгөн web-баракчабызда бир түрдүү шрифттен пайдаландык. Шрифт өлчөмү да бир гана тема тегдеринин жардамында өзгөртүлөт. Бирок интернеттеги web-баракчаларда түрдүү шрифт жана өлчөмдө, ар түрдүү түстөр менен жазылган тексттерди кезиктирүү мүмкүн.

Шрифт өлчөмүн белгилөө үчүн HTML тилинде **<FONT>** жуп теги **SIZE** (өлчөм) параметри менен чогуу иштетилет. Бул тег иштетилгенде киритилип жаткан тексттин шрифт өлчөмү өзгөрбөйт. Шрифт өлчөмү өзгөргөнүн бир гана web-браузерде көрүү мүмкүн.

Web-баракчада 7 түрдүү өлчөмдөгү шрифттерди иштетүү мүмкүн. Алар 1 ден 7 ге чейин параметр мааниси менен белгиленип, 1 эң кичине, 7 болсо эң чоң шрифт өлчөмүн билдирет. Мисалы, шрифт өлчөмүн 5 ке тең кылып алуу үчүн **<FONT SIZE=5>** көрүнүшүндөгү жуп тег жазылат. Негизги шрифт өлчөмүнө кайтуу үчүн **</FONT>** теги колдонулат.

```
<html>
<h1>Шрифттин өлчөмү</h1>
<p>Тексттин бул сабы негизги шрифтте жазылган<br>
<font size=7>7-шрифт<font size=6> 6-шрифт<br>
<font size=5>5-шрифт<font size=4> 4-шрифт</font>
</font></font></font>
<br>Тексттин бул сабы негизги шрифтте жазылган<br>
<font size=3>3-шрифт</font>
<font size=2> 2-шрифт<font size=1> 1-шрифт
</font></font></font>
<br>Тексттин бул сабы негизги шрифтте жазылган
</html>
```

HTML-код жана web-баракчага көңүл бурган болсоң, шрифт өлчөмү канча жолу тандалган болсо, ошончо жолу аякталган.

Web-баракчада түрдүү түстөрдү иштетүү аны андан да кооз жана көркүү көрсөтөт. Web-баракчада шрифт жана текст түсү менен бирге фондун түсүн да өзгөртүү мүмкүн. Шрифт түсүн тандоо үчүн **<FONT>** жуп теги **COLOR** параметри менен бирге колдонот. Бул көрсөтмөдөн кийин ″=″ белгиси жана тырмакчанын ичинде **түс коду** жазылат. Түс коду ″#″ белгиси менен башталат. Текстте анын жалпы көрүнүшү төмөнкүчө:

<FONT COLOR= ″# түс коду″>.

Түс коду RGB (Red-кызыл, Green-жашыл, Blue-көк) түстөр системасына негизделген. Бул системада керектүү түс үч негизги түстөрдүн түрдүү өлчөмдөгү аралашмасынан пайда кылынат. Үч негизги түстүн ар бири 16 негиздүү эсептөө системасындагы 00 дөн FF ке чейин (256 даана) сандар

Ак	#FFFFFF	White
Кара	#000000	Black
Кызыл	#FF0000	Red
Жашыл	#00FF00	Green
Көк	#0000FF	Blue
Сары	#FFFF00	Yellow
Кызгылт	#FF00FF	Magenta

менен белгиленет. Негизги түстөрдү колдонууда алардын англис тилиндеги айтылышынан да пайдалануу мүмкүн. Жадыбалда кээ бир түстөрдүн коддору жана англис тилиндеги айтылышы келтирилген.

Шрифт түсүн тандоону төмөнкү мисалда көрүү мүмкүн:

```
<html>
<font color="red"> Бул</font>
<font color="blue"> web-барагынын</font>
<font color="magenta"> бардык</font>
<font color="green"> сөздөрү</font>
<font color="yellow"> түрдүү</font>
<font color="black"> түстө</font>
</html>
```

Бул
web-барагынын
бардык сөздөрү
түрдүү түстө

Web-баракчадагы текст же фондун түсүн өзгөртүү үчүн **<BODY>** жуп теги, тиешелүү түрдө, **Text** (текст) же **Bgcolor** (background color, б.а. фон түсү) параметрлери менен бирге колдонулат. Бул параметрлерден кийин "=" белгиси жана тырмакча ичинде түс коду же түстүн англис тилиндеги аты жазылат. Түс коду "#" белгиси менен башталат. Текст түсүн өзгөртүү теги шрифт түсү башкача тандалган бөлүккө таасир кылбайт. Төмөнкү мисалда web-баракчада текстке жана фонго түс берүү көрсөтүлгөн:

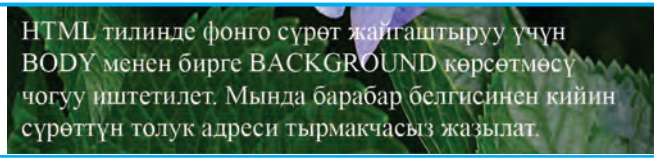
Бул {Бул бөлүк текст түсү боюнча кызгылт түстө көрүнөт}
web-барагынын бардык сөздөрү {Бул бөлүк текст түсү боюнча кызгылт түстө көрүнөт} түрдүү түстө {Бул бөлүк текст түсү боюнча кызгылт түстө көрүнөт}

```
<html><body bgcolor="#ffddd0" text="#ff00ff"><font color="red"> Бул
</font>{Бул бөлүк текст түсү боюнча кызгылт түстө көрүнөт}
<font color="blue"> web-барагынын </font>
<font color="magenta"> бардык</font>
<font color="green"> сөздөрү  </font>
{Бул бөлүк текст түсү боюнча кызгылт түстө көрүнөт}
<font color="yellow"> түрдүү</font>
<font color="black"> түстө</font>
{Бул бөлүк текст түсү боюнча кызгылт түстө көрүнөт}
</body></html>
```

MS Word программасында фонго түс берүү же фонго сүрөт жайгаштыруу мүмкүн эле. HTML тилинде фонго сүрөт жайгаштыруу үчүн **<BODY>** жуп теги **Background** параметри менен колдонулат. Мында барабардык белгисинен кийин сүрөттүн толук

адреси тырмакчасыз жазылат. Эгерде сүрөт web-документ жайгашкан катологдо болсо, анда ушул сүрөттүн аты (мисалы, Гүл.jpg) жазылышы жетиштүү.

```
<html>
<body background=Nastarin.jpg>
<body text="#ffffff">
<p>HTML тилинде фонго сүрөт жайгаштыруу үчүн
<br>BODY менен бирге BACKGROUND көрсөтмөсү
<br>чогуу иштетилет. Мында барабар белгисинен кийин
<br>сүрөттүн толук адреси тырмакчасыз жазылат.
</body></html>
```



Web-баракчага жүктөлүү тез жүрүшү үчүн сүрөттүн кеңейтмеси, адатта, JPEG же GIF форматына ылайык болушу керек, бирок BMP форматтуу сүрөт да иштетилиши мүмкүн. Бул жөнүндө кийинки сабактарда маалымат берилет.

Суроо жана тапшырмалар

1. *HTML де канча түстөн пайдалануу мүмкүн?*
2. *Web-баракчадагы шрифт өлчөмү кайсы тег жардамында белгиленет?*
3. *Web-баракчадагы шрифт түсү кайсы тег жардамында белгиленет?*
4. *Web-баракчадагы текст түсү кайсы тегдин жардамында белгиленет?*
5. *<BODY> теги параметрлеринин иши жөнүндө мисалдар келтир.*
6. *Web-баракча фонун түрдүү түстө белгилөөнү мисалдар менен көрсөт.*
7. *Web-баракча фонуна сүрөт жайгаштыруу жөнүндө айтып бер.*

Көнүгүүлөр

1. «Улуу бабаларыбыз» аттуу web-баракчада ар бир сөз башкасынан түсү менен айырмалансын.

2. «Менин үй-бүлөм» аттуу web-баракчада сөздөр, текст жана фон түсүн өзгөрт.

3. «Биздин класс» аттуу web-баракча фонуна сүрөт жайгаштыр.

54-сабак. Шрифт өлчөмү, түсү жана web-баракча фону темасын кайталоо

1. Web-баракчада фамилияң, атың жана атаңдын аты түрдүү өлчөмдүү шрифтте көрүнүшү үчүн Html-код жаз.

2. Web-баракчада район, мектеп, класс, фамилия жана атың түрдүү түс жана өлчөмдөрдө көрүнүшү үчүн Html-код жаз.

3. Html-коддо баштап web-баракча фонуна түс берүү, соң сүрөт жайгаштыруу тегин жаз. Пайда болгон web-баракча көрүнүшүн түшүндүр.

4. «Мекенимдин тарыхый шаарлары» аттуу web-баракча даярда. Анда шаар аттары түрдүү түс жана шрифт өлчөмдөрү менен айырмалансын. Шаарлар жөнүндө жазган маалыматтарың абзацтан, ортодон тегизделип чыксын.

5. «Мен сүйгөн кесип» аттуу web-баракча даярда. Анын фонуна сүйгөн кесибиңе таандык сүрөттү MS Word коллекциясынан ал (көмөк: баштап тиешелүү сүрөттү MS Word ишчи майданына жайгаштыр, кийин сүрөт нускасын Paint программасы ишчи майданына жайгаштыр, сүрөттүн бөлүгүн өзүнчө файлга сакта).

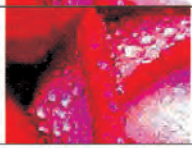
55-сабак. Web-баракчада графика

Web-баракчаны «жандандыруу»нун эң натыйжалуу усулу – ага түрдүү сүрөттөр жайгаштыруу болуп саналат. Жалаң гана тексттен турган web-баракча маалыматка бай болушу мүмкүн, бирок эригиштүү болот.

Бирок web-баракчага керегинен ашыкча сүрөт жайгаштыруу web-баракчаны одоно кылып жиберет жана web-баракча файлынын көлөмүн чоңойтуп жиберет. Файл канчалык чоң көлөмгө ээ болсо, аны Интернет тармагынан окуп алуу ошончо көп убакыт талап кылат.

Демек, web-баракчага файл көлөмү кичине болгон сүрөттөрдү жайгаштырган оң. Интернет тармагында, негизинен, **jpeg** жана **gif** форматтуу сүрөт файлдары колдонулат. Анткени, 1-ден, web-баракчанын көлөмү кичине болушу үчүн, 2-ден, web-браузерлер бул файлдарды кошумча программаларсыз көрсөтө алат.

Чындыгында, **bmp** форматтуу сүрөт файлы **jpeg** форматка өткөрүлсө, көлөмү бир нече эсе кичиреет. Сүрөт файлын бир форматтан башкасына өткөрүү атайын программалар (ACDSee, Photoshop,...) жардамында ишке ашырылат. Салыштыруу үчүн төмөнкү мисалды келтиребиз:

Сүрөттөр	Форматы	Сүрөттөрдүн өлчөмү	Графикалык файлдын көлөмү
	BMP	130 x 100 пиксел	38,3 килобайт
	JPEG	130 x 100 пиксел	4,44 килобайт
	GIF	130 x 100 пиксел	6,5 килобайт

Web-баракчага сүрөт жайгаштыруу үчүн **<IMG>** жуп эмес теги (image англ. сүрөттөлүш) колдонулат. Сүрөт файлын көрсөтүү үчүн бул тегге **SRC** (source - булак) параметри кошулушу шарт. Мисалы, web-баракчага жайгаштырылып жаткан сүрөт файлынын аты ″mypic.jpg″ болсо, HTML-документке төмөнкү жолчо кошулат:

<IMG SRC = ″mypic.jpg″>,

мында **mypic.jpg** SRC параметринин мааниси.

Өткөн темаларда текстти web-баракчада **ALIGN** параметри жардамында жайгаштыруу менен тааныштың. Бул параметрди web-баракчада сүрөттү сол же оң жакка жайгаштыруу үчүн да колдонуу мүмкүн. Чындыгында бул параметр сүрөттүн жанына текстти түрдүү абалдарда жайгаштыруу үчүн колдонулат, бирок сүрөт жайгашкан жолчодо текст болбосо, ал сүрөттүн жайгашуусуна таасир этет. Мисалы, <IMG ALIGN=″right″ SRC=″mypic.jpg″> же <IMG SRC=″mypic.jpg″ ALIGN=″right″> жазуусу ″mypic.jpg″ сүрөттү web-баракчанын оң жагына жайгаштырат.

Web-баракчада текст жана сүрөттү ALIGN параметринин төмөнкү жадыбалдагы маанилерине ылайыктуу абалдардан биринде жайгаштыруу мүмкүн:

Мисалы:

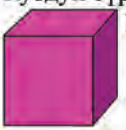
1. <IMG ALIGN=″Top″ SRC=″kub.bmp″ > жазуусу web-баракчага ″kub.bmp″ сүрөтүн тексттин бул жолчосун эң чоң элементине тегиздейт:

MIDDLE	Сүрөттүн ортосу учурдагы жолчонун астына тегизделет
ABSMIDDLE	Сүрөттүн ортосу учурдагы жолчонун ортосуна тегизделет
BOTTOM	Сүрөттүн ылдыйкы чек арасы учурдагы жолчонун астына тегизделет
TOP	Сүрөттүн жогорку чек арасы учурдагы жолчонун эң чоң элементине тегизделет
LEFT	Сүрөт сол жээкке тийип турат, текст сүрөттүн оң жагына жазылат
RIGHT	Сүрөт оң жээкке тийип турат, текст сүрөттүн сол жагына жазылат

2. <IMG ALIGN=″Top″ SRC=″kub.bmp″ > жазуусу web-баракчага ″kub.bmp″ сүрөтүн оң жээгине тегиздеп, сүрөттүн үстүнкү чек арасын ушул жолчонун эң чоң элементине тегиздейт жана ушул жолчо сүрөттүн сол жагына жазылат:

```
<html>
Кубдун сүрөтү<br>
<img src="kub.bmp" align="top"> Кубдун 6 жагы, 12 кыры, 8 чокусу
бар.
</html>
```

Кубдун сүрөтү

 Кубдун 6 жагы, 12 кыры, 8 чокусу бар.

Web-баракчада текст жана сүрөттүн жайгашуу абалдары текст форматына ылайык тегдердин иштетилишине да байланыштуу, муну көрүү үчүн жогорудагы мисалдардан экинчисинде
 тегин алып таштоонун өзү жетиштүү.

```
<html>
Кубдун сүрөтү<br>
<img src="kub.bmp" align="right"> Кубдун 6 жагы, 12 кыры, 8 чокусу
бар.
</html>
```

Кубдун сүрөтү
Кубдун 6 жагы, 12 кыры, 8 чокусу бар.

Web-баракчага сүрөт жайгаштырууда анын өлчөмдөрүн да тандоо мүмкүн. Ал үчүн WIDTH (туурасы, узуну) жана HEIGHT (эни, бийиктиги) параметрлеринен пайдаланылат. Бул буйруктардын жардамында web-баракчага жайгаштырылып жаткан сүрөттүн узуну жана туурасы пикселдерде же сүрөттүн чыныгы өлчөмүнө салыштырмалуу пайыз эсебинде берилет. Сүрөттүн өлчөмдөрүнүн пайыз эсебинде берилиши ыңгайсыз (кээ бир браузерлер пайызды кабыл албайт). Мисалы,

<IMG SRC="mypic.jpg" WIDTH=50 HEIGHT=100>

жазуусу **mypic.jpg** файлындагы сүрөттүн (жогорку жадыбалдагы 130x100 пикселдүү сүрөт) чыныгы өлчөмдөрү кандай болушуна карабай аны web-баракчага 50x100 пиксел өлчөмдүү кылып жайгаштырат:

```
<html>
<p align= "justify">
Web-баракчагы сүрөттүн айланасында бош жай калтыруу (сүрөттөн
чегинүү) үчүн HSPACE жана VSPACE буйруктары колдонулат.
Калтырыла турган жай пикселдерде берилет.<br>
<img src="lola.jpg" align=left width=50 height=100> HSPACE —
сүрөттүн оң жана сол жагынан бош жай калтырат.
<br>VSPACE — сүрөттүн жогору жана ылдый жагынан бош жай
<br>калтырат.
</html>
```

Web-баракчагы сүрөттүн айланасында бош жай калтыруу (сүрөттөн чегинүү) үчүн HSPACE жана VSPACE буйруктары колдонулат. Калтырыла турган жай пикселдерде берилет.

HSPACE – сүрөттүн оң жана сол жагынан бош жай калтырат.
VSPACE – сүрөттүн жогору жана ылдый жагынан бош жай калтырат.

Сүрөт өлчөмдөрүн чоңойтуу анын сапатынын начарлоосуна алып келээрин унутпа! Сүрөттү кайсы бир графикалык редактор жардамында керектүү өлчөмгө келтирип алып, кийин web-баракчага жайгаштырган оң!

Web-баракчага сүрөт жайгаштырганда анын айланасында бош жай болбостугу, б.а., экранда бул сүрөткө текст же башка сүрөт «тийип» чыгышы мүмкүн (жогорудагы 1-мисал). Web-баракчадагы сүрөт айланасында бош жай (боштук – space) калтыруу (сүрөттөн чегинүү) үчүн **HSPACE** (сүрөттүн сол жана оң жагы-

нан бош жай калтырат) жана **VSPACE** (сүрөттүн үстү жана астынан бош жай калтырат) буйруктары колдонулат. Калтырыла турган бош жай пикселдерде берилет. Мисалы,

<IMG SRC="жоогазын.jpg" ALIGN=left HSPACE=15 VSPACE=15>

жазуусу web-баракчадагы (жоогазын.jpg) сүрөттүн оң, сол, үстү жана астынкы жактарынан кендиги 15 пикселге тең (рамка сымал) бош жай калтырат:

```
<html>
<p align= "justify">
Web-баракчадагы сүрөттүн айланасында бош жай калтыруу (сүрөттөн
чегинүү) үчүн HSPACE жана VSPACE буйруктары колдонулат.
Калтырыла турган жай пикселдерде берилет.<br>
<img src="lola.jpg" align=left hspace=15 vspace=15> HSPACE —
сүрөттүн оң жана сол жагынан бош жай <br> калтырат.
<br>VSPACE — сүрөттүн жогору жана ылдый жагынан бош <br>жай
калтырат.</html>
```

Web-баракчадагы сүрөттүн айланасында бош жай калтыруу (сүрөттөн чегинүү) үчүн HSPACE жана VSPACE буйруктары колдонулат. Калтырыла турган жай пикселдерде берилет.

HSPACE – сүрөттүн оң жана сол жагынан бош жай калтырат.
VSPACE – сүрөттүн жогору жана ылдый жагынан бош жай калтырат.

Сүрөт айланасында (кара) рамка пайда кылуу үчүн BORDER (чек ара) параметринен пайдаланылат. Мында параметрдин мааниси катарындагы рамканын калындыгы пикселдерде алынат, б.а. мисалы:

<IMG SRC="жоогазын.jpg" BORDER=9 >.

Сүрөттүн айланасында рамка пайда кылууну өз алдынча аткарып көр.

Суроо жана тапшырмалар

1. *Кандай графикалык форматтарды билесиң?*
2. *BMP форматтуу сүрөттү JPEG жана GIF форматына PAINT программасы жардамында өткөр.*
3. *Web-баракчага сүрөт жайгаштыруу кандай аткарылат?*
4. *<IMG> тегинин кандай параметрлери бар?*
5. *Сүрөттү web-баракчанын сол же оң жагына жайгаштыруу кандай аткарылат?*
6. *Web-баракчага сүрөт өлчөмдөрүн өзгөрткөн абалда кандай жайгаштыруу мүмкүн?*

7. Кайсы параметрлердин жардамында web-баракчада сүрөттүн айланасында бош жай калтырылат?

Көнүгүүлөр

1. «Улуу бабаларыбыз» аттуу web-баракчага бабаларыбыз ийгиликке жетишкен тармакка тиешелүү сүрөттөрдү түрдүү өлчөмдөрдө жайгаштыр.

2. «Мен сүйгөн кесип» аттуу web-баракчага кесипке тиешелүү сүрөттөр жайгаштырып, сүрөттөр айланасында 15 пикселдүү рамка пайда кыл.

3. «Менин үй-бүлөм» аттуу web-баракчага үй-бүлө мүчөлөрүндүн кесибине таандык сүрөттөрдү жайгаштыр. Мында жайгаштыруунун түрдүү усулдарынан пайдалан.

56–57-сабак. Web-баракчада графика темасын кайталоо

1. «Биздин класс» аттуу web-баракчага 3 даана сүрөт жайгаштыр.

2. «Мекенибиздин тарыхый шаарлары» аттуу web-баракчага сүрөттөр жайгаштыр. Сүрөттөр жайгашуусу түрдүү абалдарда жана өлчөмдөрдө болсун (көмөк: сүрөттөрдү интернет тармагынан алуу мүмкүн).

3. «Үй жаныбарлары» аттуу web-баракча даярда. Анда жаныбарлардын сүрөттөрү жайгаштырылсын жана сүрөттүн жанында жаныбардын аты жазылган болсун (көмөк: сүрөттөрдү MS Word коллекциясынан алуу мүмкүн).

4. «Компьютер курулмалары» аттуу web-баракча даярда. Ага керектүү сүрөттөрдү графикалык редактордун жардамында өзүң жаса.

5. «Менин досторум» аттуу web-баракча даярда. Анда досторуң кызыккан кесип же тармакка тиешелүү сүрөт жана маалыматтарды жайгаштыр.

58-сабак. Web-баракчага тизме жайгаштыруу

MS Word текст процессорунун жардамында даярдалган документте тизме пайда кылуунун 2 түрдүү усулу болуп, бири маркерлүү (топчусу менен экинчиси тартиптелген (топчусу менен) тизме болуп саналат. Мисалы:

Маркерлүү тизме	Маркерлүү тизме	Маркерлүү тизме	Иреттелген тизме	Иреттелген тизме
• Информатика	o Информатика	♦ Информатика	1. Информатика	A. Информатика
• Математика	o Математика	♦ Математика	2. Математика	B. Математика
• Тарых	o Тарых	♦ Тарых	3. Тарых	C. Тарых

Web-баракчага тизме киритүү үчүн <UL> (unordered list – иреттелбеген, б.а. маркерлүү тизме) же <OL> (ordered list – тар-

типтелген тизме) жуп тегдеринен пайдаланылат. Көрүнүп тургандай, HTML-документте <OL> жана <UL> тегдеринен кийин жайгашкан маалыматтар web-браузер тарабынан оң жакка белгилүү бир аралыкта чегинүү менен көрүнөт. Тизме элементтерин белгилөө үчүн <LI> (list item – тизме элементи) жуп эмес теги колдонулат. <LI> теги менен башталган тизме элементи дайыма жаңы жолчодо көрүнөт. Мисалы, жадыбалдагы биринчи мамычасындагы маркерлүү тизмени туюнтуучу HTML-документ үзүндүсү төмөнкүчө жазылат:

```
<html><UL>
<LI>Информатика<LI> Математика<LI> Тарых
</UL></html>
```

Кантип маркердин көрүнүшүн өзгөртүү мүмкүн деген суроо туулат. Маркерлүү тизмени белгилөөчү <UL> теги **TYPE** параметри менен жазылышы мүмкүн. Бул параметр **disk** (тегерек), **circle** (айлана), **square** (боёлгон квадрат) чондугунда болушу мүмкүн, мисалы <UL TYPE = square>. Эгерде <UL> теги жогорудагы сыяктуу параметрсиз жазылса web-браузер тизме маркерин диск чондугунда деп эсептейт. Тизме элементин көрсөтүүчү <LI> теги да TYPE параметри менен жазылышы мүмкүн:

```
<html><ul>
<li type=disk> Информатика
<li type=circle> Математика
<li type=square> Тарых
</ul></html>
```

- Информатика
- Математика
- Тарых

Жадыбалдагы тартиптелген тизмени web-баракчада көрсөтүү керек болсо, HTML-документ үзүндүсү жана web-браузерде көрүнүшү төмөнкүчө болот:

```
<html><ol>
<li> Информатика<li> Математика<li> Тарых
</ol></html>
```

1. Информатика
2. Математика
3. Тарых

Эгерде тизменин тартип катары 1 ден айырмалуу сандан (мисалы, 3 төн) башталышы керек болсо, <OL> теги **START** параметри менен бирге иштетилет. Мисалы: <OL start = 3>. Эгерде тизме латын тамгалары же рим цифралары менен иреттелиши

керек болсо, анда <OL> теги TYPE параметри менен бирге иштетилет. Дал ушундай <LI> теги да TYPE жана VALUE параметрлери менен бирге иштетилиши мүмкүн.

Төмөнкү мисал web-баракчада түрдүү көрүнүштөгү тартиптелген тизме кандай жайгаштырылышын айкын көрсөтүп берет:

```
<html>
<ol start=3><li> Информатика<li> Математика<br>...
<li value=17> Тарых</ol>
<ol type=A><li> Информатика<li> Математика</ol>
<ol type=a><li> Информатика<li> Математика</ol>
<ol type=I start=5><li> Информатика <li> Математика</ol>
<ol type=i><li> Информатика<li type=A> Математика</ol>
</html>
```

Кээде web-баракчалар кооз жана көркөм болушу үчүн маркерлүү тизмеде маркер катары графикалык сүрөттөрдөн пайдаланылганын көрүү мүмкүн. Бирок, web-баракчада графикалык элементтердин бар экени узатылып жаткан маалымат көлөмүн көбөйтүп жиберет. Сүрөттүү тизмени түзүүдө <LI> тегинин зарылдыгы болбойт. Сүрөттүү тизме элементтерин бир-биринен айырмалоо үчүн <P> же
 тегдеринен пайдалануу мүмкүн. Графикалык маркерлүү тизме кандай пайда кылынышын төмөнкү мисалда көрүү мүмкүн:

```
<html><ul>
<h2>Даректар</h2><br>
<h3><img src="Barglar.gif > Арча <br>
<img src="Barglar.gif> Мажүрүмтал<br>
<img src="Barglar.gif"> Чие</h3>
</ul></html>
```

Даректар

- Арча
- Мажүрүмтал
- Чие

Бул жерде «Жалбыректар.gif» графикалык файл болуп, ал ушул web-баракча сакталып жаткан каталогдо жайгашкандыгын эстетип өтөбүз.

Интернет тармагында өтө көп үйрөтүүчү web-сайттар жайгаштырылган болуп, алардын жардамында кайсы бир илимди, анын бөлүгүн же программалык каражаттардан пайдалануу үйрөнүү мүмкүн. Белгилүү болгондой, кайсы бир илим же программалык каражат зарыл термин жана түшүнүктөрдү өз ичине алат. Бул термин жана түшүнүктөрдү web-баракчада мүнөздөө үчүн **<DL>** (definition list – мүнөздөө тизмеси) жуп теги иштетилет. Бул жуп тегдин ичинде **<DT>** (definition term – терминди мүнөздөө) жана **<DD>** (definition description – мүнөздөө баяны) жуп эмес тегдери иштетилет. Төмөнкү мисал аркылуу бул тегдердин милдетин түшүнүп алуу оңой:

```
<html><dl><dt>Информатика
<dd>Компьютер техникасын колдонууга негизделип адамдын
ишинин түрдүү тармактарында маалыматтарды издөө, топтоо,
сактоо, кайра иштөө жана андан пайдалануу маселелери менен
алектенүүчү предмет.
<dt>Дескриптор же тег<dd>HTML тилинин буйруктары болуп,"<"
жана ">" белгилеринин арасына жазылат, англисчеден tag-жарлык,
белги сыяктуу которулат.
</dl></html>
```

Информатика
: Компьютер техникасын колдонууга негизделип адамдын ишинин түрдүү тармактарында маалыматтарды издөө, топтоо, сактоо, кайра иштөө жана андан пайдалануу маселелери менен алектенүүчү предмет.

Дескриптор же тег
: HTML тилинин буйруктары болуп,"<" жана ">" белгилеринин арасына жазылат, англисчеден tag-жарлык, белги сыяктуу которулат.

Тизменин кээ бир элементине тизме кошуу керек болгон абалдар да учурайт. Мында ичи-ичине жайгашкан, б.а. катмарлашкан тизме пайда кылынат. Төмөнкү мисалда сага белгилүү болгон тегдердин жардамында катмарлашкан тизме пайда кылуу көрсөтүлгөн:

```
<html><ul>
<b>Кээ бир планеталардын жолдоштору</b>
<li> Жер
<ol><li> Ай </ol>
<li> Марс
<ol><li> Фобос <li> Деймос </ol>
</ul></html>
```

Кээ бир планеталардын жолдоштору

- Жер
  1. Ай
- Марс
  1. Фобос
  2. Деймос

Мындай тизмелерди түзүү татаал эмес, бирок тегдерди тартиптүү жазууга көп көңүл буруу зарыл болот.

Суроо жана тапшырмалар

1. *MS Worddo кандай тизме түрлөрү бар?*
2. *Web-баракчада канча түрдүү тизме иштетүү мүмкүн? Мисал келтир.*
3. *Тартип катарлуу тизме кандай түзүлөт? Мисалдар келтир.*
4. *Маркерлүү тизме кандай түзүлөт? Мисалдар келтир.*
5. *Графикалык сүрөттөлүштүү тизмени түзүү жараянын түшүндүр.*
6. *Графикалык маркерлүү тизме кандай түзүлөт?*
7. *Мүнөздөөлөр тизмеси жөнүндө айтып бер.*

Көнүгүүлөр

1. Рим цифралары катышкан тизме түз.
2. Тартиби 9 дан баштала турган тизме түз.
3. Катмарлашкан тизме түз жананы түшүндүр.

59-сабак. Web-баракчага жадыбал жайгаштыруу

Web-баракчага жадыбал киритүү үчүн **<TABLE>** (table – жадыбал), **<TR>** (table row – жадыбал жолчосу), **<TH>** (table header – жадыбалдагы тема) жана **<TD>** (table data – жадыбал маалыматтары) жуп тегдери колдонулат. <TABLE> теги жадыбал башталышын, </TABLE> теги болсо жадыбал аягын, <TR> жуп теги жадыбал жолчосун жана <TD> жуп теги болсо жадыбал мамычасын белгилейт. <TH> жуп теги темалуу чакмактарды билдирип, бул чакмактардагы маалыматтар web-баракчада калың шрифт менен чакмактын ортосунда тегизделген абалда (б.а., ALIGN=Center жана VALIGN=Middle, мында V – вертикал багытты билдирет) көрүнөт. <TH> теги <TR> тегисиз иштетилбейт. Эгерде web-баракчада жадыбалдын чек ара сызыктары да көрүнүшү керек болсо, анда BORDER параметри колдонулат. BORDER параметринин маанилери пикселдерде өлчөнөт жана бир гана жадыбал рамкасынын сырткы сызыктарынын калыңдыгына өзүнө мүнөздүү таасир этет. Жадыбал жолчолордон турат. Өз кезегинде ар бир жолчо мамычаларга бөлүнөт. <TR> жана <TD> тегдери тиешелүү түрдө </TR> жана </TD> тегдери менен жабылат. Мисалы, жадыбалды туюнтуучу HTML-документ үзүндүсү төмөнкүчө жазылат:

Ай аты	Мезгил	Айдын тартиби
Январь	Кыштын 2-айы	Жылдын биринчи айы
Декабрь	Кыштын 1-айы	Жылдын акыркы айы

```
<TABLE>
<TABLE BORDER>
<TR> <TH> Айдын аты </TH><TH> Мезгил </TH><TH>
Айдын тартиби </TH></TR>
<TR> <TD>Январь</TD><TD>Кыштын 2-айы</TD><TD>
Жылдын биринчи айы</TD> </TR>
<TR> <TD>Декабрь</TD><TD>Кыштын 1-айы</TD><TD>
Жылдын акыркы айы</TD> </TR>
</TABLE>
```

Жадыбалдын чек ара сызыктарысыз жана чек ара сызыктары менен көрүнүүчү HTML-документ жана ага тиешелүү web-баракчада жадыбалдын жайгашуусун төмөнкү мисалда көрүү мүмкүн:

```
<HTML><TABLE>
<TR><TH>Айдын аты</TH><TH>Мезгил</TH><TH>Айдын тартиби
</TH></TR><TR><TD>Январь</TD><TD>Кыштын 2-айы </TD>
<TD>Жылдын биринчи айы </TD></TR>
<TR><TD>Декабрь</TD><TD>Кыштын 1-айы </TD>
<TD>Жылдын акыркы айы</TD></TR></TABLE>
<TABLE><TABLE BORDER=7>
<TR><TH>Айдын аты</TH><TH>Мезгил</TH>
<TH>Айдын тартиби </TH></TR><TR><TD>Январь</TD>
<TD>Кыштын 2-айы </TD><TD>Жылдын биринчи айы</TD></TR>
<TR><TD>Декабрь</TD><TD>Кыштын 1-айы </TD >
<TD>Жылдын акыркы айы</TD></TR>
</TABLE></HTML>
```

Айдын аты	Мезгил	Айдын тартиби
Январь	Кыштын 2-айы	Жылдын биринчи айы
Декабрь	Кыштын 1-айы	Жылдын акыркы айы

Айдын аты	Мезгил	Айдын тартиби
Январь	Кыштын 2-айы	Жылдын биринчи айы
Декабрь	Кыштын 1-айы	Жылдын акыркы айы

Web-баракчада берилип жаткан жадыбалга тема берүү зарыл болсо **<CAPTION>** (тема) жуп теги иштетилип, ал биринчи <TR> тегинен мурда жазылышы шарт. Бул тегдин ALIGN жана VALIGN параметрлери болуп, параметр маанилери кандай иштетилиши мүмкүндүгү төмөнкү жадыбалда келтирилген:

ALIGN	VALIGN	Түшүндүрмө
TOP	жазылбайт	Тема жадыбалдан жогоруда жадыбалдын ортосуна карай тегизделет
BOTTOM	жазылбайт	Тема жадыбалдын астында жадыбалдын ортосуна карай тегизделет
LEFT	TOP	Тема жадыбалдан жогоруда жадыбалдын сол чек арасына карай тегизделет

LEFT	BOTTOM	Тема жадыбалдын астында жадыбалдын сол чек арасына карай тегизделет
CENTER	TOP	Тема жадыбалдын үстүндө жадыбалдын ортосуна карай тегизделет
CENTER	BOTTOM	Тема жадыбалдын астында жадыбалдын ортосуна карай тегизделет
RIGHT	TOP	Тема жадыбалдан жогоруда жадыбалдын оң чек арасына карай тегизделет
RIGHT	BOTTOM	Тема жадыбалдын астында жадыбалдын оң чек арасына карай тегизделет

Төмөнкү мисалда ALIGN жана VALIGN параметрлеринин иштетилишин көрүү мүмкүн:

```
<html><table><table border=7>
<caption align=right valign=bottom>ЖАДЫБАЛ</caption>
<tr><th>Айдын аты</th><th>Мезгил</th><th>Айдын тартиби
</th></tr>
<tr><td>Январь</td><td>Кыштын 2-айы </td><td>Жылдын биринчи
айы </td></tr> <tr><td>Декабрь</td><td>Кыштын 1-айы </td><td>
Жылдын акыркы айы</td></tr>
</table></html>
```

Айдын аты	Мезгил	Айдын тартиби
Январь	Кыштын 2-айы	Жылдын биринчи айы
Декабрь	Кыштын 1-айы	Жылдын акыркы айы

ЖАДЫБАЛ

Жадыбалдын ички сызыктары калыңдыгын (эки чакмакты ажыратуучу параллел сызыктардын арасындагы аралыкты) **CELLSPACING** параметри менен (мисалы, CELLSPACING =5) пикселдерде тандоо мүмкүн. Чакмакка киритилген маалыматтар менен чакмак арасындагы аралык болсо **CELLPADDING** параметри менен (мисалы, CELLPADDING=9) пикселдерде аныкталат.

Мурдараак, web-баракчага сүрөт жайгаштырганда анын айланасына тексттин жайгашышы жөнүндө маалымат берген элек. Кудум ушундай, жадыбалдын да айланасына текст жайгаштыруу мүмкүн болуп, текст жадыбалдын жалаң гана сол же

оң жагына жайгаштырылышы мүмкүн. Ал үчүн <TABLE> теги ALIGN параметри менен бирге иштетилет, мисалы: <TABLE ALIGN=LEFT>. Эгерде жадыбал web-баракчанын сол жагына тегизделсе, текст жадыбалдын оң жагында болот жана тескерисинче.

Чакмактын ичиндеги маалыматтарды форматтоо үчүн тема, текст жана сүрөттөрдү форматтоо үчүн иштетилген бардык тегдерден пайдалануу мүмкүн. Жадыбал, жолчо жана чакмактардын чек ара сызыктарынын түсүн өзгөртүү BARDERCOLOR параметри аркылуу ишке ашырылат.

Жогоруда келтирилген маалыматтардын негизинде төмөнкү HTML-документ жана web-баракчаны пайда кылуу мүмкүн:

```
<html>
<table><table align = right Border=7 bordercolor=RED
CELLSPACING=5 CELLPADDING=3>
<TR bordercolor=BLUE> <TH>Айдын<BR>аты</TH><TH>Мезгил
</TH> <TH>Айдын тартиби </TH></TR>
<TR><TD>Январь</TD><TD bordercolor=MAGENTA>Кыштын 2-айы
</TD><TD>Жылдын биринчи<BR>айы </TD>
<TR bordercolor=black><TD>Декабрь</TD><TD>Кыштын 1-айы
</TD><TD bordercolor=green>Жылдын акыркы<BR>айы</TD></TR>
</table>
<BR>Ячейканын ичиндеги маалыматтарды форматтоо үчүн
тема,текст жана сүрөттөрдү форматтоо үчүн иштетилүүчү бардык
тегдерден пайдалануу мүмкүн. Жадыбал,сап жана ячейкалардын
чектеш сызыктарынын түсүн өзгөртүү BORDERCOLOR параметри
аркылуу ишке ашырылат.
</html>
```

Ячейканын ичиндеги маалыматтарды форматтоо үчүн тема,текст жана сүрөттөрдү форматтоо үчүн иштетилүүчү бардык тегдерден пайдалануу мүмкүн. Жадыбал,сап жана ячейкалардын чектеш сызыктарынын түсүн өзгөртүү BORDERCOLOR параметри аркылуу ишке ашырылат.

Айдын аты	Мезгил	Айдын тартиби
Январь	Кыштын 2-айы	Жылдын биринчи айы
Декабрь	Кыштын 1-айы	Жылдын акыркы айы

MS Word программасында документке жайгаштырылган жадыбалдын чакмактарын бириктирүү же чакмакты дагы чакмактарга бөлүү мүмкүн, мисалы:

HTML тили да ушундай мүмкүнчүлүк берет. Ал үчүн <TH> же <TD> тегдери **COLSPAN** (column spanning – мамычаны бириктирүү) жана **ROWSPAN** (row spanning – жолчону бириктирүү) параметрлери менен бирге иштетилет. Мындай алып караганда бөлүнгөн чакмактуу жадыбалды пайда кылуу үчүн чакмакты бөлүүнүн ордуна башка чакмактарды бириктирүү аркылуу жетишүү мүмкүн. Жогорудагы жадыбалдын HTML-документи төмөнкүчө болот:

```
<HTML><TABLE><TABLE BORDER CELLPADDING=2><TR>
<TD align=middle><font color=red>1-ячейка</font></TD>
<TD COLSPAN=2 bgcolor=#00ffD0><B><font color=white>2
мамыча бириккен ячейка </font></B></TD>
<TR><TD ROWSPAN=2 bgcolor=yellow><B><I><U>Я2 катар
бириккен ячейка</U></I></B></TD>
<TD bgcolor=magenta><U>4-ячейка</U></TD>
<TD ROWSPAN=2 bgcolor=black><B><font color=white>2 катар
бириккен ячейка</font></B></TD></TR><TR><TD
bgcolor=#808080><I>6-ячейка</I></TD>
</TABLE></HTML>
```

Жадыбалдардын үстүндө дагы бир канча амалдарды аткаруу мүмкүн, аларды өз алдынча үйрөнүүнү сунуш кылабыз.

Суроо жана тапшырмалар

1. *Web-баракчада жадыбал кандай түзүлөт?*
2. *Жадыбалдын чек ара сызыктарын кандай форматтоого болот?*
3. *Жадыбалдагы тема жөнүндө маалымат бер.*
4. *Жадыбалдын темасы жөнүндө маалымат бер.*
5. *Жадыбал менен текст web-баракчада кандай жайгашышы мүмкүн?*
6. *Жадыбалдагы маалыматтарды форматтоо боюнча мисал түз.*
7. *Жадыбалдын чек ара сызыктарын форматтоо боюнча мисал түз.*

Көнүгүүлөр

1. Жадыбалга тиешелүү тегдер жана аларды мүнөздөөнүн мамычалуу web-баракчасын пайда кыл.
2. «Биздин класс» аттуу web-баракчага жадыбал жайгаштыр.
3. «Менин үй-бүлөм» аттуу web-баракчандагы үй-бүлө мүчөлөрүн жөнүндөгү маалыматтарды жадыбал көрүнүшүндө туюнт.

60-сабак. Web-баракчага тизме жана жадыбал жайгаштыруу темасын кайталоо

1. Мектепке байланышкан маркерлүү жана тартиптелген тизмени өз ичине алган катмарлашкан тизме түз.

2. «Компьютер курулмалары» аттуу web-баракчада курулмаларды жадыбалга маркерлүү тизме аркылуу жайгаштыр.

3. «Мен сүйгөн адабий каарман» аттуу web баракча даярда. Анда чыгармanын аты тема катары, автору жана адабий каармандын сапаттары жадыбал көрүнүшүндө берилсин.

4. «Биздин класс» аттуу web-баракчага 5 мыкты окуган классташыңдын фамилиясын класс журналындагы катар тартиби менен жадыбалга жайгаштыр.

5. «Мен сүйгөн кесип» аттуу web-баракчада кесибиңе байланыштуу тармактарды жадыбал ичиндеги тизме көрүнүшүндө жаз.

61-сабак. Web-баракчада «өтүү» (гиперкайрылуу)

Web-баракчадагы маалымат бир канча бөлүмдөн турган болушу мүмкүн. Бул бөлүмдөргө тез «өтүү» мүмкүнчүлүгү web-баракчаны көрүп чыгууну оңойлотот. Web-баракчада мындай өтүүлөр гипертексттүү байланышты түзөт. Гипертекст технологиясы өз мезгилинде WWW кызматынын кыска убакытта кең жайылуусуна себеп болгон эле.

Гиперкайрылуу, б.а. документтин бир жайынан башка жайына же бир документтен башка документке өтүү, **<A>** жуп теги жардамында ишке ашырылат. Бул тегдин **HREF** параметри бар болуп, анын маани өтүү жайынын адреси болот. Web-баракчанын бул тег жазылган жайы **өтүү чекити** дейилет. <A> теги параметри менен жалпы абалда төмөнкүчө жазылат:

<A HREF = ″#адрес″> текст </A>,

мында «текст» – каалагандай текст болуп, браузер аны экранда белгилеп көрсөтөт, «адрес» – өтүү керек болгон жайдын (бөлүм) адреси. Адатта, өтүү жайын аныктоочу текст экранга көк түстө жана астына сызылган көрүнүштө чыгат. Адрес да каалагандай текст болушу мүмкүн.

Web-баракчанын өтүү керек болгон жайына <A> тегинин **NAME** параметри жардамында «адрес» киритилет. Ал өтүү чекитиндеги «адрес» менен бирдей болушу керек. Бул тег NAME параметри менен бирге жалпы абалда төмөнкү көрүнүштө жазылат:

<A NAME = ″адрес″> текст </A>,

мында «текст» – каалагандай текст болуп, адатта, «текст» катары web-баракчанын ушул жеринен башталган бөлүмүнүн аты жазылат, <A NAME> деги «адрес» <A HREF> теги «адрес»тен көрүп турганындай «#» белгиси менен айырмаланат.

```
<HTML>
<H2 ALIGN="center">Web-баракчага гиперкайрылуу</H2>
<UL>
<LI><A HREF="#1-bob">I БӨЛҮМ</A>
<LI><A HREF="#2-bob">II БӨЛҮМ</A>
<LI><A HREF="#3-bob">III БӨЛҮМ</A>
</UL>
<P><H2><A NAME="1-bob">I БӨЛҮМ</A></H2>
<P>Бул жерге I БӨЛҮМ гө таандык маалымат жазылат
<P><H2><A NAME="2-bob">II БӨЛҮМ</A></H2>
<P>Бул жерге II БӨЛҮМ гө таандык маалымат жазылат
<P><H2><A NAME="3-bob">III БӨЛҮМ</A></H2>
<P>Бул жерге III БӨЛҮМ гө таандык маалымат жазылат
</HTML>
```

Web-баракчада өтүү чекити башка тексттерден «айырмаланып» турушу үчүн, аны тизменин курамына киритүү да мүмкүн. Гиперкайрылуу web-баракчада өтүү чекитинде берилген текстке чычканчанын параметрин алып келип, чычкандын сол топчусун басуу аркылуу ишке ашырылат. Мында өтүү чекитиндеги текст түсү өзгөрөт (адатта, кызгыш түстө көрүнөт).

Келтирилген мисалда өтүү чекиттери маркерлүү тизме курамына киритилген. Өтүү чекити менен өтүү жайы экранда көрүнүп турган болсо, өтүү аткарылганы билинбейт.

Гиперкайрылуу өтүү чекитиндеги тексттин ордуна же текст менен бирге сүрөт да коюу мүмкүн. Ал үчүн өтүү чекитиндеги тексттин ордуна <IMG> теги иштетилет. Бул тегдин жардамында коюлган сүрөттү браузер автоматтык түрдө өтүү буйругу менен байланыштырат. Эми web-баракчада «өтүү» үчүн текстten да, сүрөттөн да пайдаланса болот. Мисалы, HTML-документке

<A HREF=″#1-бөлүм″><IMG SRC=″mypic.jpg″></A>

жазуусу киритилсе, web-баракчада «mypic.jpg» сүрөт аркылуу «1-бөлүм»гө өтүү мүмкүн болот.

<A> теги жардамында бир web-баракчанын ичинде гана эмес, о.э. бир web-баракчадан башка web-баракчаларга өтүүнү да ишке ашыруу мүмкүн. Ал үчүн бул тегдеги «адрес» катары Интернет системасындагы кайсы бир web-баракчанын адресин, б.а. URL-адрести жазуу жетиштүү. Мисалы,

<A HREF="http//:www.rambler.ru"//Rambler.ru га өтүү</A>

Өтүүдөгү «адрес» катары дисктеги web-баракча файлынын атын жазса да болот жана мында дисктеги web-баракча ачылат. Бул өзгөчөлүк курамдык web-баракчалар (web-сайттар) жаратуу мүмкүнчүлүгүн берет.

Курамдык web-баракча – бир темага арналган, бир-бирине байланыштуу жана бир-бирине «өтүү» мүмкүнчүлүгү болгон web-баракчалар комплекси.

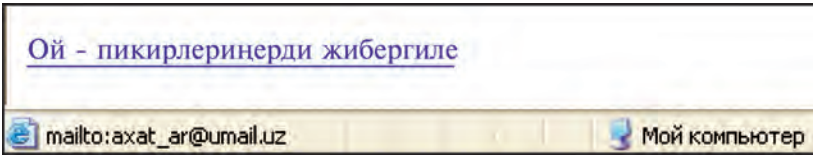
Бир теманы бөлүктөргө бөлүп, ар бир бөлүк үчүн өзүнчө web-баракча даярдоо, алардын ар биринде бир-бирине өтүүнү уюштуруу ашыкча иштей туюлушу мүмкүн. Бирок мунун төмөнкү (негизги) артыкчылыктары бар:

- Web-баракчадагы маалымат көлөмү канчалык кичине болсо, аны редакциялоо ошончо оңой болот;
- Web-баракчадагы маалымат көлөмү канчалык кичине болсо, аны окуу (көрүп чыгуу) ошончо ыңгайлуу болот;
- Web-баракча файлынын көлөмү канчалык кичине болсо, аны Интернет тармагынан «алуу» ошончо тез ишке ашырылат.

<A> теги жардамында электрон почтага да маалымат жиберүүнү уюштуруу мүмкүн. Ал үчүн почта адресинен баштап mailto (почтага) сөзү жазылат, мисалы:

<A HREF="mailto:axat_ar@mail.ru0">Ой-пикирлериңди жөнөт</A>.

Бул гиперкайрылуу көрүнүшүнөн башкалардан айырмаланбайт. Эгерде чычкандын көрсөткүчү гиперкайрылууга багытталса, абал жолчосунда электрон почтанын адреси көрүнөт:



Бирок, бул адреске кайрылганда (Интернет тармагында иштеп жаткан болсоң) web-браузер электрон почта менен иштөө

үчүн өз терезесин ачат. Бул терезенин интерфейси ар түрдүү web-браузерлерде түрдүүчө болушу мүмкүн, бирок көп программалар адрес менен пайдалануучунун аты жөнүндө маалымат чыгарат жана «Subjekt» жолчону толтурууну эстетет.

Суроо жана тапшырмалар

1. *URL-адрес дегенде эмнени түшүнөсүң?*
2. *Web-баракчада гиперкайрылуу кандай түзүлөт?*
3. *Web-баракчадагы өтүү чекити өз ичине эмнелерди алышын түшүндүр.*
4. *Курамдык web-баракча дегенде эмнени түшүнөсүң?*
5. *Электрон почтага кайрылуу кандай түзүлөт?*

Көнүгүүлөр

1. «Менин үй-бүлөм» аттуу web-баракчада үй-бүлө мүчөлөрүндүн тизмеси аркылуу алар жөнүндөгү маалыматтарга өтүүнү уюштур.

2. «Үй жаныбарлары» аттуу web-баракчада үй жаныбарларынын сүрөттөрү аркылуу алар жөнүндөгү маалыматтарга өтүүнү уюштур.

3. «Менин web-баракчаларым» аттуу web-баракча даярда жана андан өзүң даярдаган web-баракчаларга жана дагы негизги баракчага кайта турган гиперкайрылууларды уюштур.

62-сабак. Формалар

Web-баракчада формалар суроолор өткөрүү, web-сервер жана клиент ортосунда байланыш түзүү же берилген тизмеден керектүү документти тандоо максатында колдонулат. Мисалы, web-баракчанын рейтингин аныктоо, кайсы бир ишкананын продукциялары жөнүндөгү пикирлерди чогултуу, Интернет аркылуу таанышуу жана башка максаттарда web-баракчада формалар түзүлөт. Максатка карап формадагы суроолор да түрдүүчө болот. Бирок, web-баракчада суроолорду түзүү бирдей тегдер жардамында ишке ашырылат. Бул тегдер web-баракчада суроонун тексти менен бирге жооп терезесин да пайда кылат. Суроо тексти web-баракча даярдалып жатканда киритилет. Ал түзүлүшүнө карай, негизинен, эки түргө бөлүнөт:

1. Каалагандай жооп киритүү үчүн ылайыкташкан.

2. Сунуш кылынган жооптордон бирин тандоо үчүн ылайыкташкан.

Суроолорду web-баракчанын каалаган жерине жайгаштыруу мүмкүн, б.а., форманын түзүлүшүн өзүнө жаккан түрдө уюштуруун мүмкүн.

Байланышты башкача уюштуруу да мүмкүн. Мисалы, баштап көрүлгөндөй, web-баракчада керектүү суроолорду берип (жөнөкөй текст катары), бул суроолорго электрон почта аркылуу жооп жиберүүнү суранып электрон почтандын адресин көрсөтүшүң мүмкүн. Бирок мында көп (дээрлик) жооп ала албайсың. Себеби, биринчиден, бардыгы эле тааныбаган адамга кат жазууга кирише албайт. Экинчиден, формадагы суроолорго жооп берүү көп ойлонууну талап кылбайт, бирок катты болсо ойлоп жазуу керек болот.

Форма пайда кылуу үчүн <FORM> жуп теги колдонулат. Анын ACTION жана METHOD сыяктуу параметрлери бар. ACTION параметри милдеттүү болуп, анын мааниси URL-адреси болуп саналат.

Форманы жөнөтүүнү бир канча усулда ишке ашыруу мүмкүн. Форманы жөнөтүү усулун көрсөтүү үчүн METHOD параметри иштетилет. Көп абалдарда форманы жөнөтүү үчүн электрон почта көп колдонулат. Ал үчүн **<FORM>** тегине METHOD=POST жана ACTION= ″mailto: электрон почта системасындагы адрес″ параметрлеринин мааниси менен кошулат. Мисалы:

<FORM METHOD=POST
ACTION=″mailto:rtm@umail.uz″>

Формада суроо-жооп уюштуруу үчүн **<INPUT>** так теги NAME параметри менен колдонулат. Бул тег жардамында берилген суроого жооп киритүү үчүн текст майданы (жооп жолчосу) пайда кылынат. Жооп жолчосу белгилеринин саны SIZE (өлчөм) параметри менен аныкталышы мүмкүн. Суроо болсо жөнөкөй текст катары киритилет. Мисалы:

<P> Сенин атың:
<INPUT NAME = ″ат киритиле турган жай″ SIZE=25>

Бул тегдер формада «Сенин атың:» текстин жана 25 ке чейин белги киритүү мүмкүн болгон «ат киритиле турган жай» аттуу текст майданын пайда кылат.

Форманын кээ бир бөлүмдөрүнө жазыла турган жооп бир жолчого батпастыгы мүмкүн. Мисалы, формада "Түшүндүрмө" бөлүмү болсо, адатта, бул бөлүмгө бир канча жолчодон турган текст жазыла турган майдан ажыратылат. Ал үчүн **<TEXTAREA>** жуп тегинен пайдаланылат. Бул тегдин курамында текст майда-

нынын атын **(NAME)**, жолчолор **(ROWS)** жана мамычалар (COLS) санын белгилөөчү параметрлер катышат. Мисалы,

<P> Түшүндүрмө:
<TEXTAREA> NAME="Түшүндүрмө" ROWS=4 COLS=40>
</TEXTAREA>

Бул тегдер формада «Түшүндүрмө:» сөзү, жана 4 жолчо жана 40 мамычалуу (б.а., ар бири 40 белгилүү 4 жолчо) «Түшүндүрмө» аттуу текст майданын пайда кылат.

Кээ бир суроолорго анык жооптордон бири тандалат. Мисалы, маалыматын жөнүндөгү суроого сөзсүз, «Башталгыч», «Орто», «Атайын-орто» же «Жогорку» жоопторунан бирин тандайсын. Мындай суроолорго жоопторду формага алдынала киритип коюу мүмкүн. Адатта, мындай түзүлгөн жооптордун алдына тегерекче жайгаштырылып, кайсы жооп тандалса, ушул жооптун алдындагы тегерекче (чычкан жардамында) белгиленет. Формада мындай суроо-жоопту түзүү үчүн <INPUT> тегинде NAME параметри менен TYPE (tip) параметри RADIO (багыттоо) мааниси менен бирге иштетилет. Белгиленген тегерекчеге ылайык сага (же web-серверге) келе турган маалымат VALUE (маани) параметринин мааниси болот. Мисалы:

<P>Маалыматын:

<INPUT TYPE = radio NAME = "Маалыматы" value = "Башталгыч">Башталгыч

<INPUT TYPE = radio NAME = "Маалыматы" value = "Орто"> Орто

<INPUT TYPE=radio NAME="Маалыматы" value="Атайын-орто"> Атайын-орто

<INPUT TYPE=radio NAME="Маалыматы" value="Жогорку"> Жогорку

Мында TYPE=radio параметри экранда тегерекче пайда кылат; NAME= ден кийинки тырмакчанын ичинде жазылган «Маалыматы» сөзү майдандын аты болуп, экранга чыгарылбайт; VALUE= ден кийинки «Башталгыч» сөзү web-серверге жиберилет; андан кийинки Башталгыч сөзү экранга тегерекченин жанынан чыгарылат.
 теги кийинки текст жаңы жолчодон чыгуусун камсыздайт.

Кээде, сунуш кылынган жооптордон бир канчасын тандоо зарыл болуп калат. Мында TYPE параметринин RADIO маанисинин ордуна CHECKBOX (тандоо жайы) мааниси иштетилет.

Мында формада тегерекченин ордуна чакмак пайда болот. Мисалы, өздөштүрүлгөн тилдер жөнүндөгү суроону HTML-документте төмөнкүчө берүү мүмкүн:

<P>Сүйлөшө ала турган тилдериң

<INPUT TYPE = checkbox NAME = "Тил" value= "Uzbekish">Өзбек

<INPUT TYPE=checkbox NAME="Тил" value="Russian"> Орус

<INPUT TYPE=checkbox NAME="Тил" value="English"> Англис

<INPUT TYPE=checkbox NAME="Тил" value="German"> Немец

Көрүп чыгылган тегдерден пайдаланып татаал болбогон формаларды түзүү мүмкүн. Пайда кылган формаңды Интернет тармагында жайгаштырсаң, аны миллиондогон адамдар көрөт. Бирок андагы суроолорго берилген жооптор сага жетип келбейт. Пайда кылынган форма жооптору менен кайтып келиши үчүн <INPUT> теги **SUBMIT** (кошулуу) маанисиндеги TYPE параметринен пайдаланылат. Формадагы текст майдандарын тазалоо үчүн болсо <INPUT> теги **RESET** (кайра тандоо) маанисиндеги TYPE параметринен пайдаланылат. Бул тегде **VALUE** параметри иштетилсе, браузер экранда топчу пайда кылат. Мисалы,

<INPUT TYPE="submit" VALUE="Форманы жөнөтүү">

теги экранга ичинде «Форманы жөнөтүү» сөзү жазылган топчу чыгарат жана бул топчу тандалса формадагы маалыматтар керектүү адреске жөнөтүлөт,

<INPUT TYPE="reset" VALUE="Форманы тазалоо">

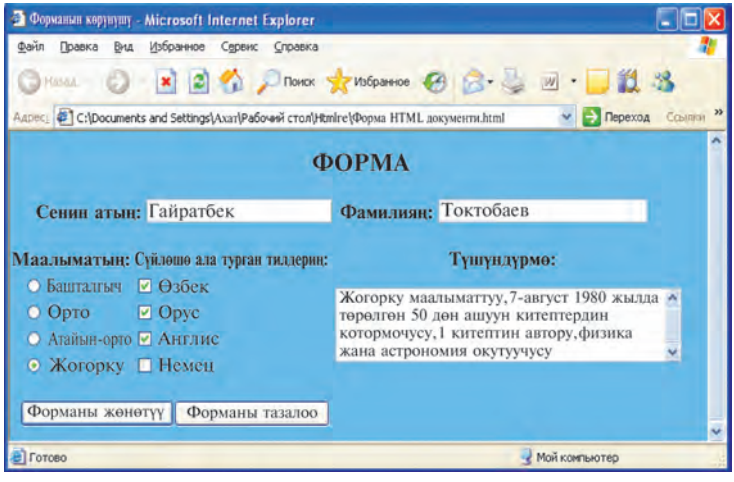
теги болсо экранга ичинде «Форманы тазалоо» сөзү жазылган топчу чыгарат жана бул топчу тандалса формадагы бардык маалыматтар өчүп, маалыматтарды жаңылоого мүмкүнчүлүк болот.

Төмөндө форма үчүн HTML-документ жана ага ылайык web-баракча келтирилген:

Жогорудагы форманы пайда кылууда web-баракчага жадыбал жайгаштыруу мүмкүнчүлүктөрүнөн пайдаландык. Ошондуктан эки түрдөгү тандоо топчулары катарлаш берилген.

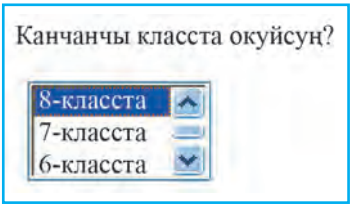
Форма пайда кылууда тандоо мүмкүнчүлүгүн <SELECT> жуп теги жардамында да ишке ашыруу мүмкүн. Мында тандалышы

```
<HTML>
<title>Вид формы</title>
<BODY BGCOLOR="#55AAFF"><H2 ALIGN="center">ФОРМА</H2>
<FORM METHOD=POST ACTION="mailto:rtm@umail.uz">
<FONT SIZE=4>Сенин атың: <INPUT NAME="Гайратбек" SIZE=26>
Фамилияң: <INPUT NAME="Токтобаев" SIZE=30><P>
<table>
<TR><TH>Маалыматың:<BR></TH><TH>Сүйлөшө ала турган
тилдериң:<BR></TH><TH>Түшүндүрмө:</TH></TR>
<TR><TD><INPUT TYPE=radio NAME="Маалыматы"
value="Башталгыч">
Башталгыч</TD>
<TD><INPUT TYPE=checkbox NAME="Тил" value="Uzbekish">
Өзбек</TD>
<TD ROWSPAN=4><TEXTAREA ROWS=4
COLS=40></TEXTAREA></TD> </TR>
<TR><TD><INPUT TYPE=radio NAME="Маалыматы" value="Орто">
Орто</TD>
<TD><INPUT TYPE=checkbox NAME="Тил" value="Russian">Орус
</TD></TR>
<TR><TD><INPUT TYPE=radio NAME="Маалыматы"
value="Атайын-орто">Атайын-орто</TD>
<TD><INPUT TYPE=checkbox NAME="Тил" value="English">Англис
</TD></TR>
<TR><TD><INPUT TYPE=radio NAME="Маалыматы"
value="Жогорку">Жогорку</TD>
<TD><INPUT TYPE=checkbox NAME="Тил" value="German">Немец
</TD></TR>
</table>
<P><INPUT TYPE="submit" value="Форманы жөнөтүү "><INPUT
TYPE="reset" value="Форманы тазалоо"></FONT></FORM>
</BODY></HTML>
```



керек болгон маалыматтар сүрүлүүчү тизме сыяктуу көрүнөт. Тизме элементтери <SELECT> теги ичинде иштетиле турган <OPTION> так теги жардамында киритилет. SELECT тегинин NAME, SIZE, MULTIPLE сыяктуу параметрлери бар. Параметрлер атын төмөндө келтирилген мисалдан түшүнүп алуу мүмкүн.

```
<html>
<title>Select</title>Канчанчы класста окуйсуң?<br>
<SELECT name="" size=3>
<OPTION value="9"> 9-класста  <OPTION value="8"> 8-класста
<OPTION value="7"> 7-класста  <OPTION value="6"> 6-класста
<OPTION value="5"> 5-класста  </select>
</html>
```



Көрүп турганыңдай, форманын тандоо бөлүгү өтө кичине жайды ээлеп турат. Бул мисалдагы тандоонун натыйжасында керектүү адреске бир гана маани – «8» жөнөтүлөт. Эгерде бир канча маанини тандоо мүмкүнчүлүгү берилиши зарыл болсо, маани талап кылбай турган MULTIPLE параметрин кошуу жетиштүү.

Суроо жана тапшырмалар

1. *Өзүң көргөн формалар жөнүндө айтып бер.*
2. *Формалар эмне үчүн иштетилет?*
3. *Web-баракчада форма кандай түзүлөт?*
4. *Форманы жөнөтүү жөнүндө айтып бер.*
5. *Формада текст майданы кандай белгиленет?*
6. *Формада бир канча жолчолуу текст майданы кандай түзүлөт?*
7. *Формада алдын-ала берилген жоопторду тандоону уюштуруу жолдорун түшүндүр.*

Көнүгүүлөр

1. «Мен неге сүйөмүн Өзбекстанды?» аттуу темадагы web-баракчалар тандоосуна катышуу үчүн суроо формасын даярда.
2. «Достошууга сунуш» аттуу форма даярда. Анда билишиң керек болгон маалыматтар берилсин.
3. «Китепкана анкетасы» аттуу форма даярда.

63-сабак. Web-баракчада «өтүү» жана формалар темасын кайталоо

1. «Мен даярдаган web-баракчалар» аттуу web-баракча даярда. Негизги баракчадан өзүң даярдаган web-баракчаларга өтүү жана артка кайтуу сүрөт жана текст аркылуу болсун.

2. «Информатика окуу китептери жөнүндөгү пикирлер» аттуу форма даярда. Ага окуу китеби жөнүндөгү пикирлерди чогултуу максат кылып алынсын.

3. 5-класстан 9-класска чейин компьютерде аткарган практикалык иштеринди ачып берүүчү «Мектеп, информатика жана мен» аттуу web-баракча даярда.

64-сабак. Интерактивдүү web-сайттар

Web-сайттар гиперкайрылуулар (hyperlink) аркылуу байланышкан бир нече web-баракчалардын жыйындысы болуп, аларды шарттуу түрдө төмөнкү эки түргө ажыратуу мүмкүн.

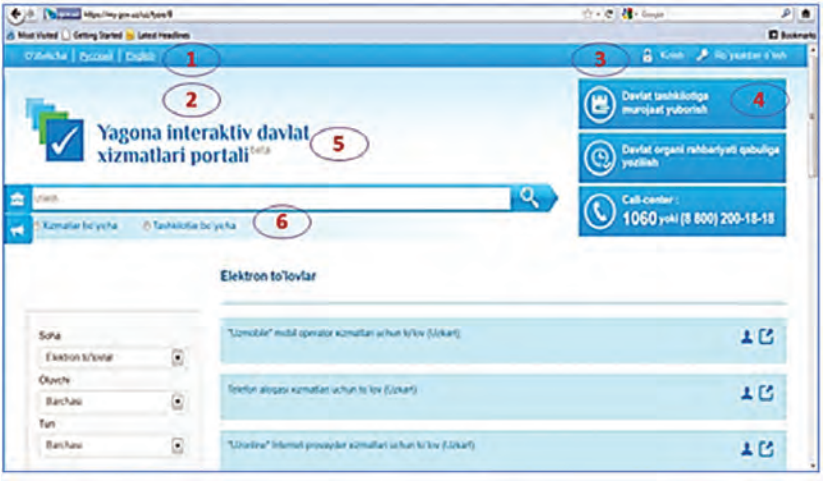
статикалык	динамикалык

Статикалык web-сайттар – өз ара байланыштуу түрдө коддолгон тексттүү, сүрөттүү жана башка түрдөгү маалыматтардан турган өзгөрбөс web-баракчалардын комплекси болуп саналат. Алар пайдалануучу үчүн кызыктуу болгон стандарттык абалдагы документтер жана маалыматтардан түзүлөт. Эгерде аларды жаңылоо же кошумча маалыматтарды кошуу керек болсо, анда программалык кодду ар жолу өзгөртүү керек болот. Бул болсо, көп убакыт жана эмгекти талап кылат жана web-баракчалар санынын артуусу натыйжасында web-сайтты башкаруу татаалдашып барат. Интернетке негиз салынган баштапкы мезгилдерде бардык web-сайттар статикалык түрдө болгондугун эстетип өтүү керек.

Азыркы мезгилде web-сайттар динамикалык түрдө даярдалат. Динамикалык web-сайттар – пайдалануучунун суроосун аткаруу жараянында маалыматтары бири-бирине байланышпаган түрдө өзгөрүп туруучу web-баракчалардан турат. Динамикалык web-сайттарда маалыматтар менен иштөө пайдалануучунун суроосуна ылайык серверде сакталып жаткан маалыматтар базасына кайрылуунун негизинде уюштурулат.

Статикалык жана динамикалык web-сайттардын айырмасын төмөнкү мисал аркылуу көрүү мүмкүн (**my.gov.uz** – бирдиктүү интерактивдүү мамлекеттик кызматынын порталы). Сүрөттө көрсөтүлгөн web-баракчалар бир web-сайтка тиешелүү болуп,

пайдалануучунун суроосуна ылайык биринчи web-баракчанын ордуна экинчи web-баракча ачлыган.



Эгерде бул web-сайт статикалык түрдө түзүлгөн болгондо, анда андагы ар эки web-баракчада бериле турган бирдей маалыматтар ар бир web-баракча үчүн кайтадан коддолмок (кайталанган маалыматтар сандар аркылуу туюнтулган). Бул web-сайт динамикалык түрдө болгондуктан атайын сценарийдин негизинде web-баракчанын тиешелүү бөлүгүнүн өзгөрүүсү аркылуу биринчи web-баракча экинчисине өтөт. Web-баракчанын өзгөрбөй калган бөлүгү үчүн бир жолу жазылган коддор ар эки баракча үчүн да жалпы болот. Демек, web-сайт бир-бирине жакын өтө көп web-баракчалардан турган болсо, шексиз, анын

динамикалык түрдө болушу чоң мааниге ээ. Динамикалык web-сайттардын негизги ыңгайлыктарынан бири маалымат ресурстарын администратор терезеси аркылуу башкаруунун жеңилдиги болуп саналат.

Динамикалык web-сайттар интерактивдүү (англ. interaction – өз ара таасир эте алуу) технологияларды колдоодо ыңгайлуу болуп, мындай технологиялар колдонулган web-сайттар интерактивдүү web-сайттар деп аталат. Бүгүнкү күндө web-сайттардын интерактивдүүлүк мүмкүнчүлүгүнө өзгөчө көңүл бурулууда.

Бирок, көпчүлүк учурларда флеш-анимациялуу же мультимедиалуу ресурстары бар болгон web-сайттар интерактивдүү web-сайттар түрүндө туура эмес көрсөтүлүүдө.

Интерактивдүү web-сайттарда маалыматтарды көрүү же таанышуу мүмкүнчүлүгүн берүү менен гана чектелип калбастан, ошондой эле аларда каттоодон өтүү, кабарды жөнөтүү жана кабыл алуу, онлайн (англ. online – байланышта, тармакта) сурамжылоолор өткөрүү, буюртманын негизинде маалымат алуу, түрдүү эсептегичтер жана башка элементтер аркылуу пайдалануучунун “байланыш” түзүү мүмкүнчүлүгү да берилет. Ошону менен бирге көптөгөн интерактивдүү web-сайттар пайдалануучу менен сайт администрациясынын ортосунда онлайн-маектер уюштуруу, онлайн-чаттар (англ. chatter – сүйлөшүү) аркылуу реалдуу убакыт аралыгында түздөн-түз байланышуу мүмкүнчүлүгүн берет.

Web-сайтка интерактивдүүлүк өзгөчөлүгүн киргизүү үчүн атайын программалык коддор – сервер скрипттеринен пайдаланылат. Бул скрипттер пайдалануучудан алына турган маалыматтар серверде кайра иштелгенден соң web-баракчада көрүнүшүн камсыздап берет.

Адатта, браузер html-файлды окуйт, эгерде ушул html-файлда сервер скрипти бар болсо, баштап алардагы сценарий боюнча амалдар серверде аткарылат, андан соң алынган натыйжалар браузерде көрүнөт. Скрипттер серверде аткарылгандыгы жана натыйжа браузерге жиберилгендиги үчүн сервер скриптинин баштапкы программалык коду браузерде көрүнбөйт.

Сервер скрипттер төмөнкүлөрдү аткарышы мүмкүн:

- каалагандай маалыматтарды тезинен кошуу жана өзгөртүү;
- пайдалануучунун суроосуна жооп берүү же аларга маалымат жиберүү;
- каалагандай маалымат же маалыматтар базасына кирүү мүмкүнчүлүгү;

- пайдалануучунун каалоосуна ылайык web-баракчаны өзгөртүү жана жөндөө сыяктуу.

Интерактивдүү web-сайттарда бир топ тартип-эрежелер бар болуп, алардан негизгилерин санап өтөбүз:

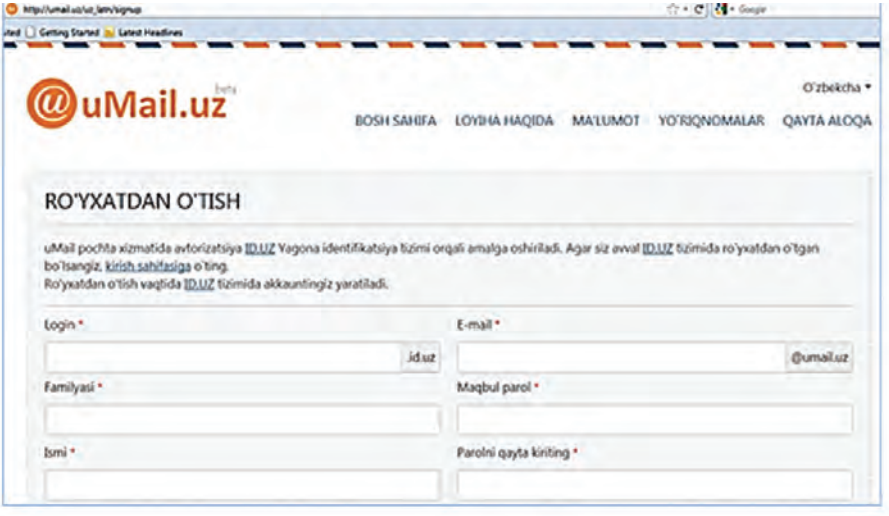
– web-сайттагы web-баракчалардын дээрлик бардык бөлүгү сервер программалары жардамында генерация (пайда) кылынат жана кайра иштелет;

– web-баракчалар генерациясы үчүн бардык маалымат тиешелүү маалыматтар базасынан алынат. Маалыматтар базасы атайын сервер компьютерлеринде түрдүү көрүнүштө сакталат;

– көбүнчө web-сайттарда уруксатты чектөө элементтеринен пайдаланылат. Бул чектөөлөр түрдүү клиенттер үчүн түрдүүчө болушу мүмкүн. Жөнөкөй клиенттер web-сайтты көрүп чыгуу гана, башкалары болсо өзгөртүүлөр киритүү мүмкүнчүлүгүнө ээ болушат.

Мында идентификациялоо (латын. identifico – тендештирүү, бирдей деп эсептөө), б.а. логин (пайдалануучунун идентификаторунун аты) жана пароль (француз. parole – сөз) элементтеринен пайдаланылат (төмөнкү сүрөттөргө кара);

– web-сайтта малыматтардын издөө системасы болот.



Динамикалык web-сайттар JavaScript, PHP, Perl жана ушул сыяктуу өтө көп таркалган жана жалпы таанылган программалоо системаларынын жардамында иштеп чыгылып, алардын жардамында сервер скрипттери жазылат. Ушул программалоо түрлөрү каалагандай татаалдыктагы сайттарды эң жогорку даражада

даярдоо мүмкүнчүлүгүн берет. Бирок, бул милдетти аткарууда чоң тажрыйба талап кылынгандыктан, программисттер тарабынан ишке ашырылат.

Суроо жана тапшырмалар

1. *Web-баракчалар түзүү технологиясына карай кандай түрлөргө бөлүнөт?*
2. *Интерактивдүү web-баракчалар жөнүндө маалымат бер жана мисалдар келтир.*
3. *Логин жана паролду колдонушу зарыл болгон web-баракчалар жөнүндө айтып бер.*
4. *Өзүңө электрон почта ачканыңда кандай web-баракчада иштегенсиң?*
5. *Компьютериңдеги кайсы электрондук окуу китеби жана колдонмолордо интерактивдүү web-баракча бар?*

65—66-сабак. Өз алдынча иштөөгө тиешелүү тапшырмалар

Төмөнкү түшүндүрүлгөн тег жана белгилерден пайдаланып web-баракчаларына өзгөртүүлөр кирит:

1. Текстгин шрифт форматын өзгөртүү:

MS Word программасында	HTML теги	HTML да мисал	Web-баракчада
б.а. надстрочный	<SUP> жуп теги	ме^{кт}еп	ме$_{кт}$еп
б.а. подстрочный	<SUB> жуп теги	ме_{кт}еп	ме$_{кт}$еп
б.а. зачеркнутый abc	<S>,<STRIKE>, <DEL> жуп тегдеринин бири	~~мектеп~~ <s>мектеп</s> <strike>мектеп</strike>	~~мектеп~~
Түшүнүк берүү	<! > так теги	<! мектеп>	көрүнбөйт

2. Атайын белгилер киритүү:

MS Word программасында	HTML жазуусу	HTML да мисал	Web-баракчада
<	&LT	&lt	<
>	&GT	&gt	>
&	&AMP	&amp	&
"	&QUOT	&quot9-класс& quot	"9-класс"
Туруктуу пробел		5 баа	5 баа
©	&COPY	&copy	©

3. Эффекттер:

HTML теги	HTML да мисал	Web-баракчада
<ACRONYM>	<acronym title= «274-мектеп»>9-класс< /acronym>	Чычкандын көрсөткүчү "9-класс" жазуусуна багытталганда 9-класс 274-мектеп
<HR>	<hr align=center size= color= red width= 60% noshade>	Беттин 60 пайызын ээлей турган ортодон тегизделген калыңдыгы 4 пикселдүү кызыл түстүү сызык сызылат, NOSHADE параметри сызыктын одонолугун жоготот.
<MARQUEE>	<marquee behavior= "alternate" width=60% height=30% bgcolor= blue> <Font size= 7 color= "white"> АРАКЕТ </marquee>	Беттин горизонтал багытында 60 пайызын, вертикал багытында 30 пайызын ээлей турган көк түстүү туура төрт бурчтун ичинде ак түстүү шрифт өлчөмү 7 ге тең "АРАКЕТ" сөзү кыймылдап турат

67–68-сабак. Кайталоого тиешелүү тапшырмалар

1. Киритилген А тексттеги эң көп учурай турган белгилер саны менен пробелдер санын салыштырып көрсөтө турган программа түз. Мисалы: «а – 21» > «пробел — 7».

2. Төмөнкү мазмунда программа түз: киритилген N ($1<N<15$) тин мааниси боюнча экранда борбору экран борборунда болгон N айлана сүрөтүн пайда кыл.

3. Берилген N ($N>21$) разряддуу сан 2 ге, 3 кө бөлүнгөндөгү калдыгын аныктоочу программа түз.

4. N санынын бардык бөлүүчүлөрүнүн санын табуучу программа түз.

5. «Кош бол, мектебим!» аттуу web-баракча даярда.

ПАЙДАЛАНЫЛГАН НЕГИЗГИ БУЛАКТАР

1. *Б. Балтаев, А. Абдукадиров, Н. Тайлаков, М. Махкамов, А. Азаматов, С. Хафизов*. Informatika va hisoblash texnikasi asoslari. 9-класс. – Т.: Cho‘lpon басмасы, 2006.

2. *Б. Балтаев, М. Махкамов, А. Азаматов*. Paskal tilida dasturlash. Методикалык колдонмо. – Т.: 2007.

3. *Б. Балтаев, А. Азаматов, Ш. Хидиров, Б. Хуррамов, Г. Ишанходжаева*. Algoritmlash va Paskal dasturlash tili bo‘yicha berilgan misol, masalalarni yechish usullari. O‘qituvchilar uchun metodik qo‘llanma. – Т.: «NIHOL» басмасы, 2012.

4. *Л.Л. Босова, А.Ю. Босова*. Информатика, 7–9. Издательство «БИНОМ», – М.: 2013.

5. *Шауцукова Л.З*. Информатика, 10–11. Издательство «Просвещение», – М.: 2000.

6. *А.Г. Кулаков, С.К. Ландо, А.Л. Семенов, А.Х. Шень*. Алгоритмика, V–VII классы. – М.: Дрофа, 1997.

7. *А.Н. Степанов*. Информатика, Учебник для вузов. Санкт-Петербург: Издательство «Питер», 2006.

Түшүндүрмө: окуу китебиндеги даталар жана терминдер боюнча пайдаланылган булактардын толук тизмеси Республикалык билим берүү борборунун алдындагы Информатика предмети боюнча Илимий-методикалык кеңештин 2015-жыл 12-марттагы токтому менен тастыкталып, сунуш кылынган.

Бул тизме Республикалык билим берүү борборунун web-сайтында (rtm.uz) жайгаштырылган.

МАЗМУНУ

I БӨЛҮМ. АЛГОРИТМДӨӨНҮН НЕГИЗДЕРИ

II БӨЛҮМ. ПРОГРАММАЛООНУН НЕГИЗДЕРИ

III БӨЛҮМ. WEB-БАРАКЧА ДАЯРДОО

Bahodir Jalolovich Boltayev
Axat Raxmatovich Azamatov
Abror Davlatmirzayevich Asqarov
Muxtor Qurbonovich Sodiqov
Gulnoza Axatovna Azamatova

INFORMATIKA
VA HISOBLASH TEXNIKASI ASOSLARI

(Qirg‘iz tilida)

Umumiy o‘rta ta’lim maktablarining 9-sinfi uchun darslik

Котормочу ***Гайратбек Токтобаев***
Редактору ***Айнура Зулпихарова***
Көркөм редактору ***Сардор Курбанов***
Техн. редактору ***Елена Толочко***
Корректору ***Айнура Зулпихарова***
Компьютерде даярдаган ***Гульчехра Азизова***

Лицензия номери AI № 163. 09.11.2009. Басууга 2015-жыл 29-июнда уруксат берилди. Форматы $60\times90^{1}/_{16}$. Times KAZ гарнитурасы. Офсеттик басма. Кегли 11. Шарттуу басма табагы 10,0. Эсеп басма табагы 8,9. Нускасы 713. Келишим № 30–2015. Заказ № 241.

Өзбекстан басма сөз жана кабар агенттигинин Cho‘lpon атындагы басма-полиграфиялык чыгармачылык үйү. 100129, Ташкент, Навоий көчөсү, 30. Телефон: (371) 244-10-45. Факс (371) 244-58-55.

Өзбекстан жана кабар агенттигинин Гафур Гулам атындагы басма-полиграфиялык чыгармачылык үйү о.э. «O‘zbekiston» басма-полиграфиялык чыгармачылык үйү басмаканаларында келишилген түрдө басылды. 100128, Ташкент, Лабзак көчөсү, 86/100129, Ташкент, Наваий көчөсү, 30.

Б 73 **Информатика жана эсептөө техникасынын негиздери**. Жалпы орто билим берүүчү мектептердин 9-классы үчүн окуу китеби /Б. Балтаев [жана б]. – Т.: Cho‘lpon атындагы БПЧҮ, 2015. – 160 б.
I. Балтаев, Б.
ISBN 978-9943-05-749-4

УЎК: 372.8:004=512.154(075)
КБК 32.81(5Кир)я721

Ижарага берилген окуу китебинин абалын көрсөтүүчү жадыбал

№	Окуучунун аты жана фамилиясы	Окуу жылы	Окуу китебинин алган кездеги абалы	Класс жетекчисинин колу	Окуу китебинин тапшырылып жаткандагы абалы	Класс жетекчисинин колу
1						
2						
3						
4						
5						
6						

Окуу китеби ижарага берилип, окуу жылынын соңунда кайтарып алынганда жогорудагы жадыбал класс жетекчиси тарабынан төмөнкү баалоо критерийлери боюнча толтурулат:

Жаңы	Окуу китебинин биринчи жолу пайдалануга берилгендеги абалы.
Жакшы	Мукабасы бүтүн, окуу китеби негизги бөлүгүнөн ажырабаган. Бардык баракгары бар, жыртылбаган, беттеринде жазуу-сызуулар жок.
Канааттандырарлуу	Мукабасы эскирген, бир аз чийилген, четтери жыртылган, окуу китеби негизги бөлүгүнөн бир аз ажыраган, пайдалануучу тарабынан канааттандырарлуу ремонттолгон. Кээ бир беттерине чийилген.
Канааттандырарлуу эмес	Мукабага чийилген, жыртылган, негизги бөлүгүнөн ажыраган же бүтүндөй жок, канааттандырарлуу эмес калыбына келтирилген. Беттери жыртылган, баракгары жетишсиз, чийип боёп ташталган. Окуу китебин калыбына келтирип болбойт.