ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Б.Б. Бөрібаев, К.С. Дүйсебекова

СИ тілінде программалау

Оқу-әдістемелік құрал

Алматы, 2007

ISBN
ББК

Пікір жазушылар: техника ғылымдарының кандидаты, доцент Рақымбергенов С.
физика-математика ғылымдарының кандидаты, доцент Жомартова Ш. Ә.

Бөрібаев Б.Б., Дүйсебекова К.С. Си тілінде программалау: оқу - әдістемелік құрал. Алматы, 2007.- 211 б.

"**Си тілінде программалау**" оқу құралы жоғарғы оқу орындарындағы техникалық бағытта білім алып жатқан студенттерге арналған, мұнда Си тілін негізге ала отырып программалау тәсілдерінен теориялық мағлұматтар беріліп, әрбір тақырып бойынша жинақталған зертханалық жұмыстарда студенттердің өз беттерімен орындауларына арналған тапсырмалар келтірілген. Ұсынылып отырған оқу құралы Си тілінде программалауды өз бетінше оқып үйренгісі келетін оқырмандардың да қажетіне жарайды деген сенімдеміз.

КІРІСПЕ

Өткен ғасырдың 80-ші жылдары басында UNIX операциялық жүйесінің басқаруымен жұмыс істейтін компьютерлер әлемінде ең белгілі тілдің бірі Си тілі болды. Содан бері ол дербес компьютерлер мен мейнфреймдер (үлкен ЭЕМ-дер) жұмыс істейтін негізгі тілдердің біріне айналды. Программалық жабдықтамалармен айналысатын көптеген фирмалар бұл тілді мәтіндік процессорларды, электрондық кестелерді, компиляторларды құру үшін кең қолданып келеді. Соңғы кездерде Си тілі ең көп таралған тілдердің біріне айналды.

Си тілінің артықшылықтарына – ондағы программалардың тез орындалуы, мәтінінің қысқа болуы, түйінді сөздерінің аз болуы, тс.с. жатады. Бұл тілдің ассемблерге ұқсас басқару мүмкіндіктері де бар. Программаларды өз қалауыңызша максималды орындау жылдамдығына немесе жадты тиімді пайдалануға үйлестіруіңізге болады. Белгілі бір операциялық жүйеге арнап жазылған программаларды басқа жүйелерге кедергісіз немесе аздаған өзгертулер енгізе отырып көшіре аласыз. Ол аппараттық жабдықтамалар жұмысына тікелей араласып, жедел жадтың жекелеген биттерімен арифметикалық және логикалық амалдар орындау мүмкіндігін туғызады. Функциялардың көлемді кітапханасы программалаушылар алдында тұрған күрделі мәселелерді жеңіл және жылдам шешуге мүмкіндік береді.

Си тілінде программалауды үйрену компьютерлік технологияларды игергісі келетін әрбір маманның алғашқы қадамдарының бірі болуы тиіс. Олай дейтініміз Си, С++ және осыларға ұқсас тілдер қазіргі кезде үздіксіз даму үстінде. Сол себепті назарларыңызға ұсынылып отырған оқу құралының алдына қойған мақсаты студенттерді тиімді де қысқа мәтінді программалар жазуға және олардың нәтижелерін алып, түрлендіру істеріне машықтандыру болып табылады. Оқу құралы тұтынушының программалау дәрежесінің кез келген деңгейіне арналған. Программалауды енді үйрене бастаған студенттер үшін ол оқулық болса, тәжірибелі программалаушылар үшін анықтамалық құрал ретінде пайдаланылуы мүмкін.

Оқу құралында теориялық мағлұматтардың көптеген нақты мысалдар мен жаттығулар арқылы түсіндіріліуі кітаптың негізгі бір ерекшелігі деуге болады.

Мемлекеттік тілімізге әлі информатика терминдері толық аударылып, олардың стандарты бекітілмегендіктен, кітап соңында қолданылған терминдердің бірсыпыра аудармасы келтірілген.

Қазақ тілінде жарық көріп отырған бұл оқу құралы төл тілімізде дәріс алып жатқан студенттеріміз үшін программалау негіздерінің қыры мен сырын меңгеру жолында көмекші құрал бола алады деген сенімдеміз.

Жаңа программалау тілін үйренудің жалғыз тәсілі – ол тілде программалар жазу
Брайэн Керниган

1. АЛГОРИТМДЕУ НЕГІЗДЕРІ

1.1. Алгоритм және программа ұғымдары

ЭЕМ-ді пайдалану істерін қарастырмас бұрын оның жұмысымен тығыз байланысты алгоритм, программа ұғымдарын білуіміз қажет. Әрбір ЭЕМ алдын ала берілген алгоритммен, яғни жоспармен жұмыс істейді. Алгоритмді заңдылық, реттелген амалдар жиыны, кезекпен орындалатын операциялар тізімі деп ұғынған жөн. Бұл ұғым қазіргі кезде кеңінен қолданылып жүр. Оның көптеген анықтамалары да бар. Соның бірін келтіре кетейік.

Алгоритм *– берілген есептің шығару жолын реттелген амалдар тізбегі түріне келтіру*. Кез келген есепті қарапайым амалдарды тізбектей орындау арқылы шығаруға болады. Алгоритмді ЭЕМ-де орындау үшін оны программа түрінде жазып шығу керек.

Программа *– алгоритмді машинаға түсінікті нұсқаулар тізімі ретінде жазу.* Программа машинаға түсінікті командалардан тұрады. Осы командалар тізбегі орындалу барысында есептің нәтижесі шығады. Әрбір ЭЕМ алдын ала жазылған программамен істейді. Процессор программаның құрамындағы командаларды кезекпен орындап отырады. Командалар тізбегін программа деп қарастыруға болады. Команда бір ғана қарапайым амалды орындау үшін берілген бұйрық ретінде беріледі. Командалар: арифметикалық немесе логикалық амал; ақпаратты тасымалдау командасы; берілген сандарды салыстыру командасы; келесі командаларға көшу тәртібін орындау, т.с.с.

Алгоритм және программа ұғымдары ұқсас екені көрініп тұр, алгоритм есептің шығару жолын қарапайым әрекеттер тізбегімен өрнектесе, программа сол өрнекті машинаға түсінікті тілмен жазып береді. Сондықтан программа жазу тәсілдері солардың қандай ережені пайдаланып, қандай компьютерлерге арналып жазылғанына байланысты алгоритмдік тілдер немесе программалау тілдері ұғымына келіп тіреледі.

Сонымен, алгоритм оны атқарушы ЭЕМ-ге жұмыс тәртібін түсіндіретін ережелер мен нұсқаулар тізбегінен тұрады. Алгоритмді атқарушының рөлін негізінен адам немесе автоматтандырылған аспап, яғни ЭЕМ, робот, т.б. атқарады. Мысалы, $y=(ax+b)(cx-d)$ функциясын есептеу төменгі іс-әрекеттерден тұрады:

1) a-ны x-ке көбейту, оны R_1 деп белгілеу;
2) оған b-ны қосу, нәтижесін R_2 деп белгілеу;
3) c-ны x-ке көбейту, оны R_3 деп белгілеу;

4) одан *d*-ны алу, оны R_4 деп белгілеу;

5) R_2-ні R_4-ке көбейту, оны у деп белгілеу.

Алгоритмнің орындалу кезінде оны орындаушыға келесі жолы қандай нұсқау бойынша орындалатыны белгілі болуы қажет. Ал орындаушының жүзеге асыра алатын командалар жиыны – командалар жүйесін құрайды.

Алгоритм мен программаға байланысты ЭЕМ-нің мынадай жұмыс ерекшеліктері болады:

1) есепті шығару жолы алгоритм түрінде өрнектелуі қажет;

2) алгоритм программаға айналдырылуы тиіс;

3) программа машина жадына енгізіліп, ретімен орындалуы керек. Алгоритм күнделікті тұрмыста да кеңінен қолданылады, Мысалы, студент болу үшін алгоритмнің мынадай қадамдарын орындау керек.

1. Орта мектепті бітіріп, аттестат алу.
2. Керекті құжаттарды аттестаттың түпнұсқасымен бірге белгілі бір оқу орнына өткізу.
3. Конкурстан өту.

Бұл көрсетілген пункттердің орнын ауыстыруға болмайды. Олар көрсетілген ретпен кезектесіп орындалуы тиіс. Сонда ғана керекті нәтижеге (студент болу) қолымыз жетеді.

Алгоритм информатиканың іргелі ұғымдарының бірі. Квадрат теңдеудің түбірін табу ережесі, үшбұрыштың ауданын есептеу жолдары алгоритмдердің мысалдары болып табылады.

1.2. Алгоритм қасиеттері

Алгоритм ұғымының мәнін ашатын негізгі қасиеттерінен немесе оған қойылатын талаптардан қысқаша мағлұматтар келтірейік. ЭЕМ-де орындалуға тиіс алгоритмдерге мынадай талаптар қойылады:

1) ол анық әрі дәл өрнектелуі тиіс – детерминділік қасиеті;
2) алгоритм шектелген уақыттан соң нәтиже беруі тиіс, яғни алгоритм қадамдарының саны шексіз болмауы керек – нәтижелілік қасиеті;
3) бір тектес есептерге жалпы бір ғана алгоритм қолданылуы тиіс – жалпылық қасиеті;
4) алгоритмді кішкене бөліктерге бөлу мүмкіндігі болуы қажет – дискреттілік немесе модульдік (бөліктік) қасиеті.

Біріншіден, алгоритм анық, әрі *дәл өрнектелуі* қажет. Онда қандай қадамдар көрсетілсе, тек соны ғана орындау керек. Есеп шығару жолына керектің бәрі біржақты анықталуы және орындаушыға түсінікті, әрі нақты болуы тиіс. Екіншіден, алгоритм *нәтижелі болуы* керек. Әрекеттердің шектелген санынан кейін белгілі бір уақыт ішінде қорытынды нәтиже алуымыз қажет. Әрбір алгоритм біршама бастапқы мәліметтердің болуын талап етеді және іздеген нәтижені алуға жеткізеді. Мысалы, сандарды

қосу алгоритмі үшін бастапқы мәліметтерге қосылғыштар мәні жатады, ал нәтижесі қосынды болады. Үшіншіден, алгоритмнің *жалпылық қасиеті* болады, яғни бастапқы мәліметтер мәнінің бір жиыны бір ғана нәтиже береді. Егер берілген мәліметтер өзгерсе, нәтиже де өзгереді. Басқаша айтқанда, бір алгоритм бір типтес есептердің әр түрлі алғашқы мәліметтері үшін әр түрлі нәтижелер беруі тиіс. Мысалы, квадрат теңдеуді шешу алгоритмі кез келген *a*, *b*, *c* мәндері үшін оның түбірін дұрыс табуы керек. Төртіншіден, алгоритмнің *үзік-үзік модульдерге бөліну* қасиеті болуы тиіс, яғни үлкен алгоритмді бірнеше кішкене алгоритмдерге жіктеуге әрқашанда мүмкіншілік болуы керек. Сондықтан алгоритмді екі-үш бөлікке бөліп, оларды өзінше құра алатын дәрежеде жұмыс істелуі қажет. Олар тек бірінің қорытындысын келесі жолы керекті мәлімет ретінде қолдануы тиіс.

1.3. Алгоритмдерді жазу жолдары

Алгоритмдерді ЭЕМ-де орындау үшін оларды алдын ала жазып алу керек, яғни ол белгілі бір заңдылықпен өрнектелуі тиіс. Жалпы алгоритмді жазып өрнектеу түрлеріне:

1) табиғи тіл арқылы жазу;
2) графика жолымен жазу;
3) алгоритмдік тілдермен жазу жолдарын жатқызуға болады.

Бірақ алгоритмді табиғи тілде өрнектеу ЭЕМ-дерде қолданылмайды, өйткені онда дәлдік, нақтылық болмайды.

Алгоритмдерді графика жолымен жазу, онан кейін оны программаға айналдыру істері мемлекеттік стандартпен бекітіліп ақпарат өңдеу жұмысында кеңінен қолданылып келеді. Алгоритмдік, яғни программалау тілдері есептерді шығару жолын баяндау-өрнектеу үлгісі, белгілі бір проблеманы шешу негізінде орындалатын әрекеттерге басшылық, ой еңбегін үнемдеуге мүмкіндік беретін әдіс, есеп шешімін табуды автоматтандыруға қажетті іс-әрекет, жаңа проблеманы шешу кезінде қолданылатын тәсілдер, күрделі процестерді өрнектеу және математикалық дәлдікпен анық етіп жазу құралы бола алады.

1.4. ЭЕМ-де есеп шығару кезеңдері

ЭЕМ-де есеп шығару күрделі процесс болып есептеледі, ол төмендегі кезеңдерден тұрады:

1. Берілген есепті математикалық түрде өрнектеу, яғни есепті мәселе ретінде қоя білу.
2. Есепті шығарудың ЭЕМ-ге ыңғайлы сандық тәсілдерін анықтау.
3. Есепті шығару жолын алгоритм түрінде бейнелеу.
4. Есепті ЭЕМ-де шығару программасын жасау және оның қателерін түзету.

5. Есепке керекті мәліметтер дайындау.
6. ЭЕМ-де есепті шығару және шыққан нәтижені іс жүзінде қолдану.

Берілген есепті математикалық түрде өрнектеу дегеніміз – есептің берілген мәндерін математикалық таңбаларды қолданып жаза білу және керекті математикалық формулаларды, өрнектерді анықтау болып саналады.

Күрделі формулаларды, теңдеулерді арифметикалық амалдар тізбегіне айналдыру есепті шығарудың сандық тәсілдерін табу не анықтау жолы болып есептеледі. Қазіргі кезде барлық есептердің шығару жолының сандық тәсілдері белгілі десе де болады, тек солардың ішінен өзімізге тиімді жолын таңдап алуымыз керек. Бұл мақсатта есепті шығару дәлдігін, нәтижені жылдам табу мүмкіндігін, мәліметтерді дайындау мен есепті шығарудың бағасын салыстыра отырып қарастыру қажет.

Есептің алгоритмін жасағанда, оның шығару жолын тізбектелген іс-әрекеттер ретінде схема түрінде өрнектеледі.

Программа жасағанда қазірде кеңінен тараған программалау тілінің бірінде алгоритм нақты түрде жазылады. Бізде кең тараған тілдерге – Паскаль, Дельфи, Си жатады. Жазылған программаның қатесін түзету ЭЕМ-нің көмегімен шешіледі, өйткені жіберілген қателерді тек ЭЕМ ғана жылдам аңғарып, түзету мүмкіндігін береді.

Есепті шығаруға керекті деректерді сұрыпталған күйінде алдын ала қағазға, әйтпесе магниттік дискіге жазып, ЭЕМ-нің жадына реттей отырып енгіземіз. Есептің нәтижесін алған соң шешім қабылдау және оны іс жүзінде қолдану – мамандардың жұмысы. Тек солар ғана белгілі бір шешім қабылдай алады. Бірақ оқып-үйрену барысында кездесетін, яғни студенттерге арналған есептерде жоғарыда көрсетілген сатылардың бірсыпырасы болмайды, өйткені олар бірден формула күйінде беріледі, шығарудың сандық тәсілі формулада айқын көрініп тұрады (интеграл, туынды болмаса), нәтижені алған соң, оны жазып алу жеткілікті. Мәселені шешудің немесе есеп шығарудың көрсетілген алты сатысы күрделі өндірістік есептерде, дипломдық немесе курстық жұмыстарда жиі кездеседі.

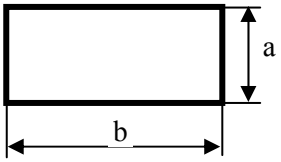
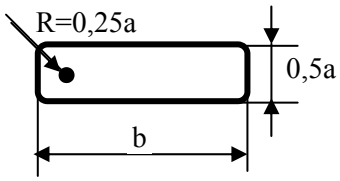
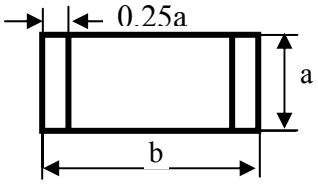
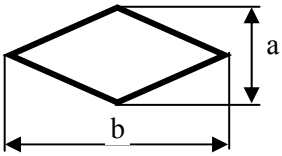
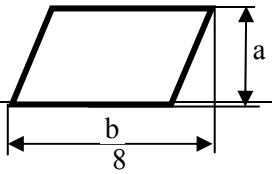
1.5. Алгоритмдерді графикалық түрде жазу

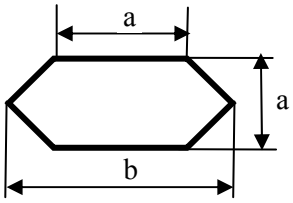
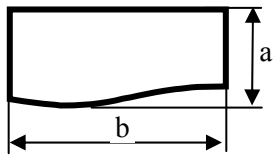
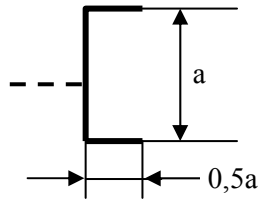
Алгоритмдерді өрнектеудің көп тараған түрі – оны график арқылы бейнелеу. Ал бұл – түсінікті, анық, көрнекті түр болып есептеледі. Тек оларды сызу көбірек еңбекті талап етеді. Графикалық жолмен алгоритмдерді жазу үшін мемлекеттік стандарт белгіленген, онда кез келген амал белгілі бір геометриялық фигурамен өрнектеледі. Ол фигуралар немесе блоктар амалдар символы деп те аталады. Блоктар бағытталған сызықтармен байланысып, бірінен соң бірі орналасады. Жиі қолданылатын амалдар, яғни мәліметтерді ЭЕМ-ге енгізу, формуламен есептеу, шарттар-

дың орындалуын тексеру, нәтижені қағазға басу символдары 1-кестеде көрсетілген. Осы суреттегі көрсетілген блоктардан (символдардан) алгоритм схемалары құрастырылады. Алгоритмдер схемасымен ақпаратты өңдеудің әрбір сатысы немесе орындалатын операциялар реті анықталады. Кейде алгоритмдер схемасын оның блок-схемасы деп те атайды.

1 кесте

Алгоритмдерді бейнелеу блоктары

Іс-әрекеттің аты	**Блоктың пішімі**	**Атқаратын жұмысы**
Процесс		Математикалық өрнектерді есептеу
Басы – соңы		Алгоритмдерді бастау, аяқтау
Алдын ала анықталған процесс (подпрограмма)		Қосалқы программаларға кіру және шығу
Шешім		**Есеп шығару** жолын таңдау
Енгізу-шығару		Мәліметтерді енгізу және шығару

Модификация		**Цикл басы**
Құжат		Нәтижені баспаға (қағазға) шығару
Түсініктеме		Схеманы, формулаларды түсіндіру

Сонымен алгоритм блоктармен немесе геометриялық көпбұрыштар түріндегі фигуралармен өрнектеледі. Әр блоктың ішіне орындалатын іс-әрекеттің (амалдың) мазмұны жазылады. Символдардың (блоктардың) бір кіру және бір шығу сызықтары болуға тиіс.

Алгоритмдер блоктардың өзара байланысуына қарай үш түрлі бірыңғай құрылымға – сызықтық, тармақтық және циклдік болып үш топқа бөлінеді. Енді соларды қарастырайық.

1.6. Алгоритмдердің бірыңғай құрылымы

Күрделі алгоритмдерді құру үшін қарапайым бірыңғайланған алгоритмдік құрылымдар қолданылады. Олар сызықтық, тармақталу және цикл құрылымдарынан тұрады.

Программалау теориясында кез келген күрделі программаны үш түрлі құрылымнан құрастыруға болатыны дәлелденген, олар: *сызықтық*, *тармақты* және *циклдік* құрылымдар. Осы үшеуі құрылымдық программалаудың негізгі конструкциялары, яғни құраушылары болып саналады.

Сызықтық құрылым бірінен кейін бірі орындалып тізбектеле орналасқан бірнеше операторлардан тұрады.

Тармақты – шартқа байланысты екі операторлардың бірінің орындалуы

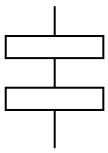
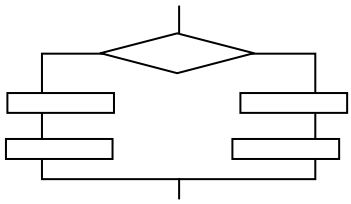
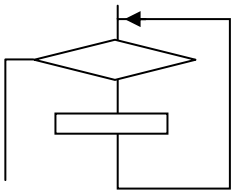
Цикл – операторлар бөлігінің бірнеше рет қайталана орындалуы.

Оператор – тілдің қарапайым сөйлемі, ол белгілі бір әрекет немесе амал орындап, **;** таңбасымен аяқталады.

Негізгі конструкцияларды пайдалану мақсаты – қарапайым құрылымды программа алу болып саналады. Мұндай программалар оңай оқылады, түзетіледі және керек болса, оңай өзгертіледі. Құрылымдық программалауда `goto` операторын қолдануға болмайды, өйткені ол программа логикасын түсінуді қиындатады. Бірақ кейде `goto` операторын қолдану қажет болатын кездер болады.

2 кесте

Алгоритмдердің бірыңғай құрылымдары

Сызықтық құрылым	Тармақты құрылым	Циклдік құрылым
		

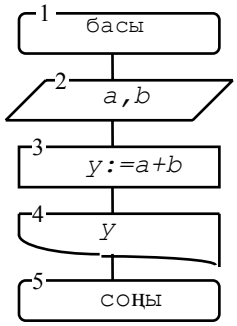
Программа жұмысын басқару операторларын программаның басқарушы конструкциясы деп атайды. Олар:

- құрама операторлар;
- таңдау операторлары;
- цикл операторлары;
- көшу операторы.

1. Сызықтық құрылымды алгоритм немесе қарапайым сызықтық алгоритм іс-әрекеттердің орындалу ретіне қарай тізбектеле орналасқан блоктардан тұрады. Амалдардың бұлай бірінен соң бірі реттеліп орындалу тәртібін табиғи атқарылу дейді.

Мысалы, $y = a+b$ формуласы бойынша есептеу тіктөртбұрыш арқылы кескінделетін есептеу блогы (3-блок) арқылы өрнектеледі. Ал нәтижені қағазға басу үшін көпбұрышты құжат алу блогын (4-блок) пайдаланып,

оның ішіне нәтиженің атауларын жазамыз.Жоғарыда көрсетілген *y=a+b* формуласымен есептеу үшін *a* және *b*-ның сандық мәндерін ЭЕМ-ге енгізіп (2-блок), содан кейін қосу амалын орындап, ақырында *y*-ті экранға (қағазға) басып шығарып, жұмысты тоқтатамыз. Осы алгоритмнің схемасы 1.1-суретте көрсетілген, ал оның жанында Си тіліндегі программасы жазылған. Программа мәтінін құру жолдарын келесі тарауларда қарастырамыз.



1.1-сурет. Алгоритм схемасы

```
#include <stdio.h>
#include <conio.h>
main()
{
  int a,b,y;
  clrscr();
  printf("a, b =");
  scanf("%i%i",&a,&b);
  y = a + b;
  printf("y = %i", y);
  getch();
}
```

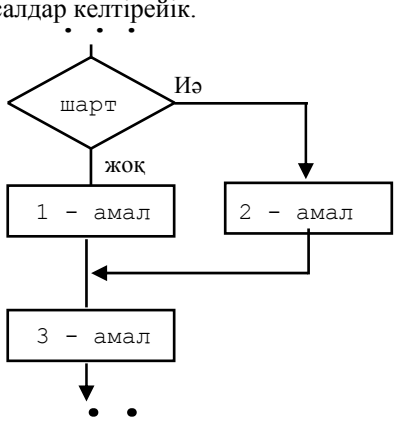
2. Тармақталу алгоритмдері. Тұрмыста кездесетін алгоритмдер әр түрлі болып келеді. Олардың жиі кездесетін түріне алгоритмнің белгілі бір шарттың орындалуына не орындалмауына байланысты тармақталып бірнеше жолдарға бөлінуі жатады.

Тармақталу алгоритмінің құрылымы қарапайым болып келеді. Мұнда арифметикалық теңсіздік (теңдік) түрінде берілген логикалық шарт тексеріледі. Егер ол орындалса, онда алгоритм бір жолмен, ал орындалмаса екінші жолмен жүзеге асырылады, яғни есепті шығару жолы тармақталып екіге бөлініп кетеді. Тармақталу алгоритмдеріне шартты тексеру блогы міндетті түрде кіреді. Ол ромб түрінде кескінделіп, басқа блоктармен 1 кіру және 2 шығу сызықтары арқылы байланысады. Көбінесе тармақталу алгоритмдері екі түрде кездеседі, олар "таңдау" және "аттап өту" мүмкіндіктерін іске асыруға көмектеседі.

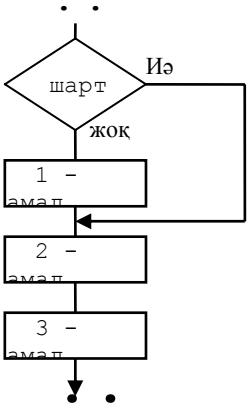
"Таңдау" жолымен тармақталуда берілген шарт тексеріледі (1.2-сурет), егер ол шарт орындалса (орындалуы ақиқат болса), онда 2-амал жүзеге асырылып, содан кейін келесі 3-амалға көшеміз. Ал, егерде шарт орындалмаса, яғни оның орындалу мүмкіндігі жалған болса, онда 1-амал атқарылып, содан кейін 3-амал атқарылады. Сонымен, шарттың ақиқат немесе жалған болуына байланысты 1-амал немесе 2-амал орындалады.

"Аттап өту" (1.3-сурет) алгоритмінде шарт орындалса, 1-амалды аттап өтіп, бірден 2-амалды, содан кейін 3-амалды орындаймыз. Ал шарт

жалған болса, онда 1-амал міндетті түрде орындалып, одан кейін 2- және 3-амалдар жүзеге асырылады. Тармақталу кезеңінде шартты тексеру блогы орындалуы барысында, алгоритмнің екі мүмкіндігінің тек біреуі ғана таңдап алынып жүзеге асырылады да, ал екінші таңдап алынбаған тармақ біріктіру нүктесіне дейін орындалмай қалады. Енді осыған нақты мысалдар келтірейік.



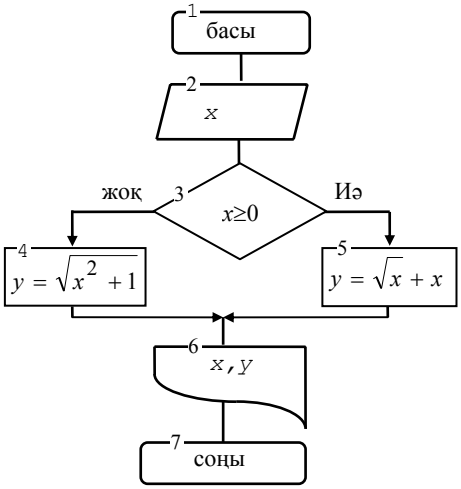
1.2 -сурет. "Таңдау" алгоритмі алгоритмі



1.3-сурет."Аттап өту"

1-мысал. y функциясын төмендегі формула бойынша есептеп шығару керек.

$$y = \begin{cases} \sqrt{x} + x, & x \geq 0 \\ \sqrt{x^2 + 1}, & x < 0 \end{cases}$$

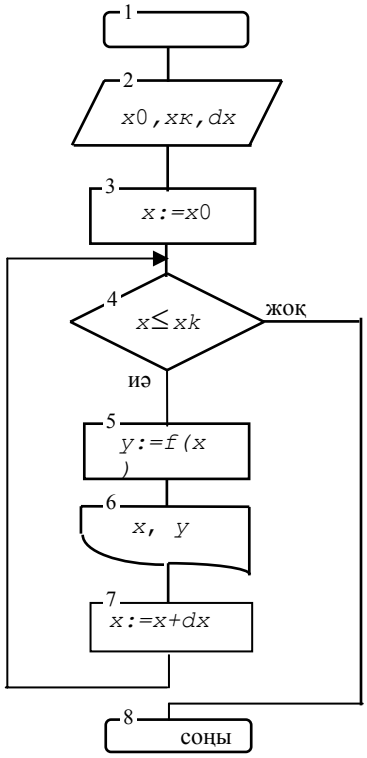


1.4-сурет. Тармақталу алгоритмі

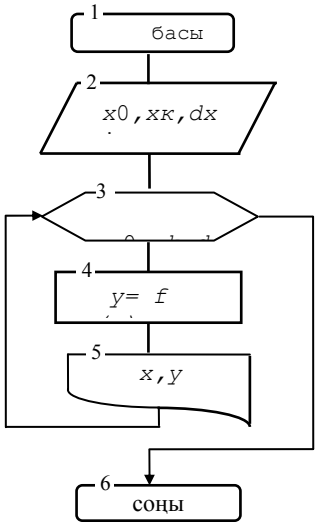
Мұнда *x* айнымалысының таңбасына (оң, теріс) байланысты не жоғарғы, не төменгі формуланы таңдап алып, сол арқылы *y* функциясының мәнін табамыз (1.4 сурет). 2-блоктың орындалу барысында *x* айнымалысына белгілі бір мән беріледі де, ол мән енгізу операторлары арқылы программаға енгізілуі тиіс. Бұдан кейін енгізілген мәннің оң немесе теріс екендігі үшінші шартты тексеру блогы

арқылы айқындалады. Шарттың "ақиқат" (иә) немесе "жалған" (жоқ) болуына байланысты 4- не 5-блоктардың бірі ғана орындалып, "таңдау" орындалады. 6-блок *x* айнымалысының және *y* функциясының сандық мәндерін экранға немесе қағазға басып шығарады.

3. Циклдік алгоритмдер. Математикада, экономикада көптеген есептерді шығару кезеңінде бір теңдеуді пайдаланып, ондағы айнымалының өзгеруіне байланысты оны бірнеше рет қайталап есептеуге тура келетін сәттер де жиі кездеседі. Осындай қайталап орындалатын есептеу процесінің белгілі бір бөліктерін цикл деп атайды. Осы бірнеше рет қайталанатын бөлігі бар алгоритмдер тобы циклдік алгоритмдерге жатады. Циклдік алгоритмдерді пайдалану оларды кейіннен программаларда цикл операторы түрінде қысқартып жазу мүмкіндігін береді. Циклдер қайталану санының алдын ала белгілі және белгісіз болуына байланысты екі топқа бөлінеді. Қайталану сандары алдын ала белгілі болып келетін циклдер тобы *арифметикалық цикл* болып есептеледі, ал орындалу саны белгісіз циклдер – *қадамдық (итерациялық) цикл* болып аталады.



1.5-сурет. Қарапайым циклдік алгоритм



1.6-сурет. Модификаторлы циклдік алгоритм

Практикада белгілі бір айнымалының сандық мәніне байланысты орындалатын арифметикалық циклдер жиі кездеседі. Мұнда арифметикалық прогрессияға ұқсас болып келетін циклдер ең қарапайым арифме-

тикалық цикл болып табылады. Оны басқару қайталану кезеңінде прогрессияның заңына сәйкес тұрақты шамаға өзгеріп отыратын цикл параметрінің сандық мәнімен байланысты болуы тиіс.

Цикл орындалуы алдында оның айнымалы аргументі – параметрі алғашқы мәнге ие болуы керек, сонан кейін қайталану кезеңінде цикл параметрі белгілі бір шамаға (қадамға) өзгере отырып, ол алдын ала берілген ең соңғы мәнге дейін жетуі қажет.

Алгоритмнің орындалу барысында цикл параметрі, мысалы, *x* өзінің ең алғашқы *x*0 мәнінен ең соңғы *xk* мәніне дейін тұрақты шамаға (*dx*) өзгеріп отырады. Осының нәтижесінде *x* мынадай мәндерді қабылдайды: *x*0, *x*0+*dx*, *x*0+2*dx*, ..., *x*0+*(n*-1*)dx*, *xk*, мұндағы *n* – циклдің қайталану саны, ол былай анықталады:

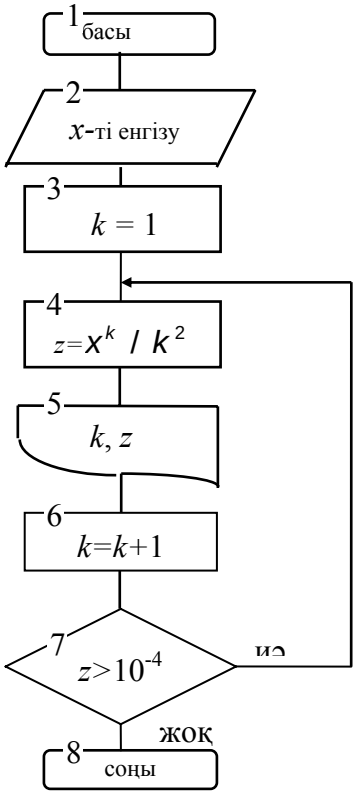
$$n = \left[\frac{xk - x0}{dx} \right] + 1 \ ,$$

мұнда [...] – өрнектің бүтін бөлігі алынатынын көрсетеді. *n* әрқашанда бүтін сан болуы тиіс, егер ол аралас сан болса, онда оның бөлшегі алынып тасталады, өйткені циклдің қайталану саны бүтін натуралдық сан болуы тиіс. Арифметикалық цикл үшін *y=f(x)* функциясының есептелу жолы алгоритм ретінде 1.5-суретте көрсетілген. Мұндағы 3-ші, 4-ші, 7-блоктар циклді ұйымдастыру үшін қажет. Олар цикл параметрінің алғашқы мәнін, өзгеру қадамын белгілеп және оның ең соңғы мәніне жеткен-жетпегенін тексереді. Ал 5- және 6-блоктар бірнеше рет қайталанып циклдің өзін құрайды. 4-блок шартты тексеріп қайталану процесін ұйымдастырады.

Алгоритм схемасын салуды және программаны жазуды жеңілдету үшін цикл алгоритмдері ықшамдалған түрде "модификатор" немесе "цикл басы" блогын пайдалану арқылы жазылады. Онда 1.5-суретте көрсетілген 3-ші, 4-ші, 7-блоктардың орнына "цикл басы" блогы орналасады. Ол алтыбұрыш тәрізді геометриялық фигурадан тұрады және оның міндетті түрде екі кіру және екі шығу сызығы болуға тиіс. Осы блокты пайдалану арқылы жоғарыда келтірілген алгоритм 1.6-суретте көрсетілген түрде кескінделеді. Параметрдің алғашқы *x* мәні оның соңғы *x* мәнінен кем болса, онда оның қадамы *dx* оң сан болады. Керісінше, параметрдің алғашқы мәні оның соңғы мәнінен артық болса, онда қадам теріс сан болады.

4. Қадамдық циклдер. Циклді орындаудың алдында, оның қайталану саны белгісіз болған жағдайда қадамдық циклдер пайдаланылады. Мұнда циклді жазу үшін тек қана "шартты тексеру" блогын қолдану қажет, ол циклді аяқтау үшін белгілі бір шартты тексереді. Қадамдық циклдердің схемасын сызғанда модификаторды (алтыбұрышты) қолдана алмаймыз, себебі алдын ала циклдің неше рет қайталанатыны бізге белгісіз. Енді осындай циклдер жұмысына мысал келтірейік.

3-мысал. $Z = \frac{x^k}{k^2}$ функциясының мәндерін k = 1, 2, 3, ... және Z 0.0001-ден артық болған жағдайда есептейік, мұндағы $0 \le x \le 1$. Бұл мысалда алдын ала цикл неше рет қайталанатынын айта алмаймыз, өйткені бізде тек k параметрінің алғашқы мәні мен қадамы ғана белгілі. Сонымен қатар Z функциясының 0.0001-ден артық болуы циклді қайталау шарты болып есептеледі (Z > 0.001). 1.7-суретте осы есептің алгоритм схемасы көрсетілген.



1.7-сурет. Қадамдық цикл алгоритмі

1.7. Программалау тілдері

Алгоритмдерді ЭЕМ-ге түсінікті мәтін ретінде жазуға арналған қарапайым жасанды тіл программалау тілдері деп аталады. Әрбір ЭЕМ-нің өзінің машиналық программалау тілі болады, оны командалар тілі немесе кодтар (арнайы таңбалау) тілі дейді. ЭЕМ тек өз ана тілінде, яғни машиналық тілде жазылған программаларды ғана орындай алады. Алайда, машина тілінде программа жазу өте күрделі жұмыс, өйткені ол тек екілік (он алтылық) жүйедегі кодтардан тұрады және әр машинада әр түрлі машиналық тіл қолданылады.

Программа жазуды жеңілдету үшін математикалық формулаларды кеңінен қолданатын, ағылшын тілінің негізінде жасалған алгоритмдік тілдер Бейсик, Паскаль, Фортран, Си, т.б. кеңінен қолданылады. Алгоритмдік немесе программалау тілі – жазу ережелері қарапайым жасанды тіл. Оның машина тілдерінен айырмашылығы – табиғи ағылшын тілі негізге алынып, кең тараған математикалық таңбалармен толықтырылып жасалған. Сондықтан алгоритмдік тілдерде программа жасау адамдарға әрі жеңіл, әрі ыңғайлы болып келеді. Алгоритмдік тілдер автоматты түрде ЭЕМ-нің көмегімен аудармашы программалар арқылы машиналық тілге көшіріледі. Алгоритмдік тілді машина тіліне тікелей аударатын үлкен

программаларды транслятор деп атайды. Алгоритмдік тілдерді пайдалану программалауды жеңілдете отырып, ЭЕМ-де есеп шығару процесін оңайлатады, алайда онда есеп шығару уақыты аздап көбейеді.

Алгоритмдік тілдер машинаға және проблемаға бағытталған болып екіге бөлінеді. Машинаға бағытталған тілдердің машина тілінен айырмашылығы, олар ЭЕМ-нің ерекшеліктерін есепке ала отырып әріптерді де пайдаланады. Қазіргі кезде машинаға бағытталған тілдерде маман программалаушылар жұмыс істейді. Оларға – автокод, макроассемблер, ассемблер тәрізді тілдер жатады.

Проблемаға бағытталған тілдер шығарылатын есептердің ерекшеліктерін еске ала отырып, есептің математикада жазылу тіліне жақындастырылады. Бұларға – Бейсик, Фортран, Паскаль, Си, т.с.с. тілдер жатады.

Негізінде ЭЕМ-де кез келген программалау тілінде жазылған есеп машина тіліне аударылып барып орындалады. Есептің орындалу кезеңдерін 1.8-суретте көрсетілген схема түрінде көрсетуге болады.

Есепті шы№ар

1.8-сурет. Есепті шығару кезеңдері

Қазіргі кезде бес жүзге жуық алгоритмдік тілдер тараған. Олардың әрқайсысы белгілі бір мақсаттарда қолданылады. Мысалы, Фортран – ғылыми-техникалық (инженерлік) есептерді шығару үшін, Паскаль – оқып үйренуде, ал Си тілі үйрену үшін де, өндірістік есептерде де, операциялық жүйелер жазу үшін де қолданыла беретін кең қолданыстағы тіл болып табылады.

Бақылау сұрақтары

1. Алгоритм және программа дегеніміз не, олардың қандай ұқсастықтары мен айырмашылықтары бар?
2. ЭЕМ-де орындалатын алгоритмдердің қандай қасиеттері болады?
3. Алгоритмдерді өрнектеу жолдары.
4. ЭЕМ-де есеп шығару кезеңдері.
5. Алгоритм схемаларының әр түрлі блоктары, олардың бейнеленуі, байланыстары.
6. Сызықтық, тармақталу және циклдік алгоритмдер.
7. Қадамдық циклдер және олардың ерекшеліктері.

2. СИ ТІЛІНДЕ ПРОГРАММАЛАУ

Си тілі өткен ғасырдың 70-жылдары басында АҚШ-та Bell Telephon Laboratories компаниясының қызметкері Дэннис Ритчидің бастауымен дүниеге келді. Бұл тілдің негізі Алголдан басталып, Паскаль және ПЛ/1 тілдерімен қатар пайда болды.

Си тілінің шығуы UNIX операциялық жүйесінде программалаумен тығыз байланысты, өйткені бұл жүйе ассемлерде және осы Си тілінде жазылып шыққан болатын. UNIX жұмыс істеу ортасы Си тілін жүйелік программалау тілі ретінде елге таныстырды, ол компиляторлар мен операциялық жүйелер жазу үшін қолайлы деп саналды, кейіннен Си тілі кез келген салада программалар жазуға да өте қолайлы тіл болып табылатыны анықталды.

Алғаш рет UNIX 1969 жылы Нью-Джерси штатындағы Bell фирмасының лабораториясында PDP-7 мини-ЭЕМ-інде жасалып шықты. UNIX PDP-7 компьютерінің ассемблер тілінде жазылды. Бұдан соң сол лабораторияның жетекшісі Кен Томпсон 1970 жылы B деп аталған жаңа тілге арнап компилятор жасап шықты. Осы тілді Си (ағылшынша C) тілінің негізі деп атауға болады.

Жалпы Си тілінің тез дамуына 1983 ж. оның стандартын жасау мақсатында Америка ұлттық стандарттар институтында (ANSI) Техникалық комиссияның құрылуы себепші болды. Комиссия жұмысына сол кездегі алдыңғы қатарлы мамандардың көпшілігі араласып, олар тек программалау мәселелерін ғана қарастырмай, кең таралып келе жатқан IBM PC компьютерлеріне арналған компиляторлар жасауды да осы стандартқа енгізді.

Си тілінің негізінде 1983-жылы Си++ тілі жасалып шықты, сол кезден бері тілдің бірнеше нұсқалары пайда болып, ол қазіргі ең көп тараған тілдердің біріне айналды. Біздің елде Си тілінің алғашқы нұсқасы орыс тілінде Б.Керниган мен Д.Ритчидің “С программалау тілі” деген кітабы арқылы бірнеше рет жарық көрді (1985-1991 жж.).

Бірсыпыра фирмалар осы тілге арнап компиляторлар жазды, мысалы, Borland International фирмасы 1989 ж. жасаған біріктірілген программалау ортасы TurboC++ жүйесін дүниеге келтірді. Ол DOS ортасында жақсы жұмыс істеді. Ал 1992 ж. жасалған Borland C++ жүйесі Windows ортасында да жұмыс істейтін жақсы компилятор болып табылады.

Сонымен Си/Си++ программалары Паскаль тілінің біріктірілген (интегралданған) ортасы сияқты DOS ортасында да және Windows жүйесінде де жұмыс істей береді.

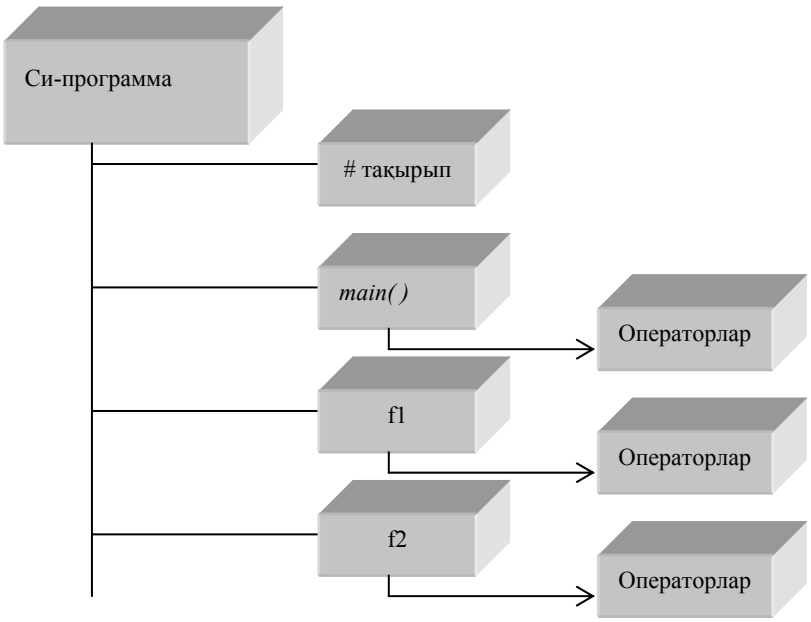
Біз қарастырғалы отырған Си тілінің негізгі ұғымдары мен операторлары кез келген мектептің немесе жоғары оқу орнының компьютерлерінде

жүре беретін біріктірілген ортаның редакторы арқылы теріліп жұмыс істей береді.

2.1. **Си тілінде жазылған программаның құрылымы**

Кез келген программа бір немесе бірнеше функциялардан тұрады. Олар программа құруға керекті негізгі модульдер болып табылады. Кез келген программаның жазылған алғашқы мәтіні бастапқы код деп аталады. Оны біріктірілген программалау ортасында теріп, сонан соң компилятор арқылы машиналық кодқа түрлендіріп орындаймыз.

Келесі суретте Си программасының жалпы құрылымы көрсетілген.



2.1-сурет. Си программасының құрылымы

Сонымен, Си программасы бірнеше функциялардан (main, f1, f2…) құралады және олардың біреуі міндетті түрде main() болуы қажет.

Қарапайым программаның мысалын қарастырайық.Жалпы кез келген функция оның тақырыбы мен тұлғасынан (денесінен) тұрады.

Алдымен программа препроцессор арқылы өңделіп, оның директиваларын (командаларын) орындайды. Мұнда программаға тақырыптық файлдар – программаға қосымша элементтер енгізетін мәтіндік файлдар жазылады. Олар мәліметтерді енгізу/шығару операцияларын немесе экран сипаттамаларын өзгерту үшін қажет.

Программадағы кез келген функция тақырыбы препроцессордың директивасынан және функция атынан тұрады. Функция атына жалғасып, жақша ішіне параметрлер жазылуы мүмкін, кейде параметрлер болмайды, ондайда жақша ішіне ешнәрсе жазылмайды.

Функция тұлғасы операторлардан тұрады, олар жүйелі жақшалармен шектеледі. Әрбір операторлан кейін **;** таңбасы қойылады.

Енді бір программа мысалын келтірейік:

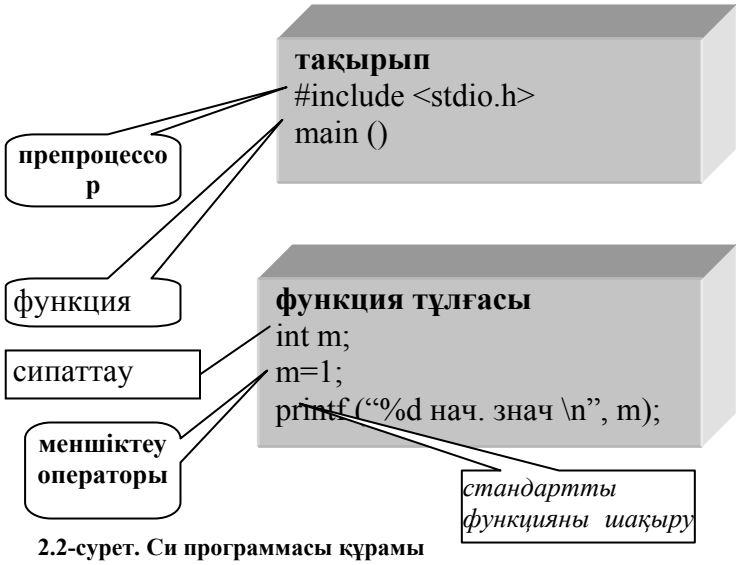
```
/* Герон формуласы арқылы үшбұрыш ауданын табу  */
#include <stdio.h>      /* енгізу/шығару директивасы */
#include <math.h>       /* математикалық функциялар директивасы  */
main()                  /* басты функцияны қолдану */
{
  int a,b,c;           /* бүтін айнымалыларды сипаттау */
  float p,s;           /* нақты айнымалыларды сипаттау */
  printf("\n, үшбұрыш қабырғаларын енгіз : \n");
  scanf("%f%f%f",&a,&b,&c);
  p=(a+b+c)/2;
  s=sqrt(p*(p-a)*(p-b)*(p-c));
  printf("s=%f",s);
}
```

Программада түсініктемелер беру үшін /* және */ таңбалары қолданылады, олардың ішіне қазақша, орысша, ағылшынша сөз тіркестерін жазуға болады.

Препроцессор директивалары **#include** сөзінен кейін жазылады, **stdio.h** тіркесі енгізу/шығару операциялары орындалатынын білдіреді. Ал **math.h** сөз тіркесі программада математикалық функциялар пайдаланылатынын көрсетеді (2.2-сурет).

Басты функция **main()** аргументсіз жазылған, сол себепті жақша ішінде ешнәрсе көрсетілмеген. Ал функция тұлғасы операторлардан (немесе басқа функциялардан) тұруы тиіс. **Int** түйінді сөзі **a,b,c** айнымалыларының бүтін мән қабылдайтынын, **float** түйінді сөзі **p,s** айнымалыларының нақты мән қабылдайтынын сипаттап тұр.

Келесі жол үшбұрыш қабырғаларын енгізуді талап ететін сөз тіркестерін экранға шығарады, мұндағы **\n** таңбалары сөз тіркесі алдында және одан кейін курсор бір жол төмен түсетінін көрсетеді. **Scanf** сөзінен басталатын жол **a,b,c** мәндерін пернелерден қабылдайды, сонан кейін жарты периметр есептеліп, аудан мәні анықталады да, соңғы нәтиже экранға шығарылады.



2.2-сурет. Си программасы құрамы

2.2. СИ тілінің қарапайым элементтері

2.2.1 Пайдаланылатын символдар

СИ тілінің символдарын бес топқа бөлуге болады.

1. Тілдегі түйінді сөздер (ключевое слово – keyword) мен идентификаторларды құрастыру үшін қолданылатын символдар (1 кесте). Бұл топқа ағылшын алфавитінің бас және кіші әріптері мен астын сызу символы кіреді. Басқа Паскаль, Бейсик тілдеріндегі тәрізді бір символды өрнектеу үшін қолданылатын бас әріп пен кіші әріп бірдей болып саналмайды, мысалы, **A** және ***a*** айнымалылары екеуі екі түрлі болып есептеледі.

2.1 кесте

Латын алфавитінің бас әріптері	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Латын алфавитінің кіші әріптері	a b c d e f g h i j k l m n o p q r s t u v w x y z
Астын сызу символы	_

2. Қазақ (орыс) алфавитінің бас және кіші әріптері мен араб цифрлары (2 кесте) сөз тіркестері мен түсініктеме мәтін жазуда пайдаланылады.

2.2 кесте

Қазақ алфавитінің бас әріптері	А Ә Б В Г Ғ Д Е Ж З И К Қ Л М Н Ң О Ө П Р С Т У Ұ Ү Ф Х Ц Ч Ш Щ Ъ Ы І Ь Э Ю Я
Қазақ алфавитінің кіші әріптері	а ә б в г ғ д е ж з и к қ л м н ң о ө п р с т у ұ ү ф х ц ч ш щ ъ ы і ь э ю я
Араб цифрлары	0 1 2 3 4 5 6 7 8 9

3. Операциялар таңбалары, айыру белгілері, қатынас таңбалары және арнайы символдар (3 кесте).

2.3 кесте

Символ	**Атауы**	Символ	**Атауы**
,	үтір	}	оң жақтағы жүйелі жақша
.	нүкте	{	сол жақтағы жүйелі жақша
;	нүктелі үтір	<	кіші
:	қос нүкте	>	үлкен
?	сұрақ белгісі	**<=**	кіші немесе тең
'	апостроф	**>=**	үлкен немесе тең
!	леп белгісі	[	сол жақтағы тік жақша
|	тік сызықша	]	оң жақтағы тік жақша
/	қиғаш сызық	#	нөмір белгісі
\	кері қиғаш сызық	%	қалдық табу (пайыз) белгісі
~	тильда	&	амперсанд
*	жұлдызша	“	қостырнақша
+	плюс	=	теңдік белгісі
-	минус	= =	тең (қатынас таңбасы)
^	логикалық “емес” таңбасы	!=	тең емес (қатынас таңбасы)
)	оң жақтағы жай жақша	&&	конъюнкция (ЖӘНЕ)
(	сол жақтағы жай жақша	| |	дизъюнкция (НЕМЕСЕ)

4. **Басқару және айыру символдары**. Бұл топқа: босорын, табуляция символы, жаңа жолға көшу, жаңа бетке көшу таңбалары жатады. Бұлар тұтынушы анықтаған объектілерді – константтар мен идентификаторларды бір-бірінен айыру үшін қолданылады.

5. Көрсетілгендерден басқа Си тілінде **басқару тізбектері** деп аталатын мәліметтер енгізу мен шығаруда қолданылатын арнайы символдар тіркесі бар. Басқару тізбектері кері бөлу сызықшасы белгісінен (\) басталатын латын әріптері мен цифрлар тізбегінен тұрады (4 кесте).

2.4 кесте

Басқару тізбектері	**Атаулары**	**Он алтылық кодтары**
\a	Қоңырау	007
\b	Бір орынға кері қайтару	008
\t	Горизонталь табуляция	009
\n	Жаңа жолға көшу	00A
\v	Вертикаль табуляция	00B
\r	Сырғыманы (каретканы) қайтару	00C
\f	Форматты жылжыту	00D
\"	Қостырнақша	022
\'	Апостроф	027
\0	Нөл-символ	000
\\	Кері қиғаш сызық	05C
\ddd	Сегіздік жүйедегі кодтар жиыны	
\xddd	Он алтылық жүйедегі кодтар жиыны	

\ddd и \xddd (мұндағы d цифр тұрғанын көрсетеді) түріндегі тізбектер компьютердегі кодтар жиыны арқылы өрнектелетін символды сәйкесінше сегіздік және он алтылық цифрлар тізбегімен бейнелей алады. Мысалы, каретканы қайтару символы бірнеше тәсілмен өрнектеле алады:

\r – жалпы басқару тізбегі,

\015 – сегіздік сандардан тұратын басқару тізбегі,

\x00D – он алтылық сандардан тұратын басқару тізбегі.

Мысалы, жеке мынадай \n (жаңа жолға көшу) басқару тізбегін \010 немесе \xA түрінде де жазуға болады.

2.2.2 Тілдің қарапайым объектілері

Тілдің қарапайым объектілеріне сан, идентификатор, константа, айнымалы және функция, өрнек ұғымдары кіреді.

Программадағы негізгі амалдардың орындалуына керекті мәліметтердің сандық, логикалық немесе символдық (литерлік) мәндері болады.

Олармен жұмыс істеу қолайлы болуы үшін алгебра курсындағы белгілеулерге ұқсас шартты атаулар пайдаланылады. Бұл атаулар әр түрлі мәндерді (сандық мән, символдық мән, т.с.с.) қабылдауы мүмкін, сондықтан оның типі деген ұғым енгізіледі.

Сандар. Сандар мен айнымалылар бүтін және нақты болып екіге бөлінеді. *Бүтін сандар*: +4, -100, 15743, 0 т.с.с. Қазіргі дербес компьютерлер үшін қолданылатын бүтін сандар (ағылшынша INTEGER) -32768 бен +32767 аралығында ғана жазылады, бұдан үлкен сандар нақты сандарға айналдырылады.

Нақты сандар кәдімгі табиғи аралас сандар тәрізді санның бүтіні мен бөлшегін нүкте арқылы бөлген күйде жазылады. Мысалы: 2.65, 0.5, -0.862, -6.0. Ал өте үлкен немесе өте кіші нақты сандар көрсеткіші бар экспоненциал сандар ретінде $mE\pm p$ түрінде жазылады да, олардың диапазоны әлде қайда кең болады, мұндағы m-санның мантиссасы деп аталады, E-онның дәрежесі дегенді білдіреді, ал p- дәреженің өз мәні. Мысалы:

Кәдімгі жазылуы	*Си тілінде жазылуы*
145	145
147,125	147.125
-6,045	-6.045
$12*10^{14}$	12E+14
$-0,52*10^{4}$	-0.52E4
$5,2*10^{-12}$	5.2E-12
$-45*10^{6}$	- 45E6

Тұрақты немесе **константа** деп программаның орындалу барысында мәндері өзгеріссіз қалатын шамаларды айтады. Тіл ережесі бойынша бірнеше константтар типтері болады, мысалы, символдық, бүтін, нақты константтар, т.б.

Айнымалылар деп программаның орындалу барысында әр түрлі мәндерді қабылдай алатын шамаларды айтады. Әрбір айнымалы мен константтар программа алдында сипатталуы тиіс. Олардың компьютер жадында алатын орны типтеріне байланысты болады. Константтар мен айнымалылар идентификатормен белгіленеді.

Атау – идентификатор (identification – объектінің белгілі бір символдар тіркесіне сәйкестігін бекіту) программаны және программадағы тұрақтыларды, типтерді, айнымалыларды, функцияларды, файлдарды және тағы басқаларды белгілеп жазу үшін қажет. Идентификаторлар тұрақтыларды, айнымалыларды, олардың түрлерін, функцияларды, программаларды, файлдарды, т.б. программа объектілерін белгілеу үшін

қолданылады. Оның ұзындығын өте үлкен етудің қажеті жоқ, өйткені атауларды теру және кейіннен сақтау біраз уақыт керек етеді.

Идентификатор – латын әрпінен басталып, әріптер мен цифрлардан тұратын тізбек. Мысалы, a, beta, b5, baga, т.с.с. Айнымалыны сипаттау мынадай нұсқада орындалады:

```
char  f;
long  z, t;
int   a, beta, baga;
float b5, k, n;
int   y = 10;
```

Қолданылатын негізгі типтерге мыналар жатады:

char short int long float double

Алғашқы төрт тип бүтін сандарды, соңғы екеуі – нақты сандарды өрнектейді, char типі компьютерде бір символды (байт) бейнелейді.

Идентификаторлар латын алфавитінің бас және кіші әріптерінен және цифрлардан құралады. Әріп ретінде астын сызу символын (_) қолдануға рұқсат етілген. Бас әріп пен кіші әріп бірдей болғанымен әр түрлі идентификаторлар болып саналады, мысалы, abc, ABC, A128B, a128b төрт түрлі идентификатор болып есептеледі.

Идентификатор ұзындығына шек қойылмайды, бірақ оның алғашқы 31 символы ғана мағыналы болып саналады. Идентификатор сипаттау кезінде анықталады да, кейінгі операторларда қолданыла береді. Сипатталатын идентификатор Си тілінің түйінді сөздерімен сәйкес келмеуі тиіс.

Константалардан, айнымалылардан, функциялардан және операциялар таңбаларынан ***өрнектер*** құралады. Әрбір өрнек арифметикалық операциялар таңбаларымен қажетті жақшалар көмегімен біріктірілген бірнеше операндтардан (сан, айнымалы, константа) тұрады. Математикадағы формулалар, алгебрадағы көпмүшеліктер программалау тілінде тек осы өрнек ұғымы арқылы беріледі.

Егер өрнек мәні бүтін немесе нақты сан болатын болса, ол арифметикалық өрнек болып саналады. Арифметикалық өрнектерде мынадай операциялар: + - * / % болады. Жалпы өрнектер бір жол бойына жазылады және олардағы операция реттілігі жақшалармен анықталады. Өрнектерді жазу мысалдары:

```
i = i+1;   k = 5.35;   x1= (-b+sqrt(b*b-4*a*c))/(2*a) ;
y = sqrt(sin(x)+1);   c = 2*pi*r;   R = 19.36;
```

Қатынас таңбасы арқылы біріктірілген екі арифметикалық өрнек мәні басқа тілдердегідей ақиқат (0-ге тең емес) немесе жалған (0-ге тең) деп айтылады. Бірақ Си тілінде логикалық тип түсінігі айтылмайды, ол Си++ тілінде бар.

Түйінді сөздер – мағынасы алдын ала анықталған идентификаторлар, олардың саны шектеулі. Тұтынушы айнымалы, константа, өз функциялары

аттарында тілдің түйінді сөздерін пайдаланбауы тиіс, олар тек өз мағынасында ғана қолданылады.

Бірсыпыра түйінді сөздер тізімін келтірейік.

```
auto double int struct break else long switch
register tupedef char extern return void case float
unsigned default for signed union do if sizeof
volatile continue enum short while
```

Бұған қоса операторлар мен стандартты функциялар аттары да түйінді сөздер тізімі секілді басқа мағынада қолданылмайды.

Стандартты функциялар. Си тілінде алдын ала программалары жасалып стандартты модульге жинақталып қойылған, қажет кезінде пайдалануға болатын объектілердің бірі стандартты функциялар болып табылады. Олар жиі кездесетін математикалық және басқа да функцияларды есептеу үшін қолданылады. Стандартты функцияны жазу үшін міндетті түрде функцияның аты және жақшаның ішінде аргументі көрсетілуі қажет. Стандартты функциялар: fabs(x), sin(x), cos(x), asin(x), acos(x), tan(x), exp(x), log(x), sqrt(x), atan(x), т.с.с. Функцияны есептеу барысында аргумент пен функция типтерінің әр уақытта сәйкес келе бермейтінін есте сақтаған жөн. Си тіліндегі стандартты функцияларды пайдалану үшін <math.h> тақырыптық файлы (прототипі) қолданылады.

Комментарий – түсініктеме ретінде қолдануға болатын символдар тізбегі. Олар ұлттық алфавиттерді де пайдалана береді. Комментарий басы мен аяғы /* ...*/ осындай таңбалармен шектелуі тиіс. Олар бір немесе бірнеше жолдардан да тұра алады. Си++ тілінде жол соңында тұратын комментарийлер // сиволдарынан кейін орналасады. Си және Си++ тілдерінде құрастырылған есептерді бір компилятор арқылы шығаруға болатындықтан, көбінесе түсініктемелердің жоғарыдағы екі түрін де пайдалана беруге болады.

```
/* программа ішіндегі комментарийлер осылай орналасады */
```

Программалау тілінің белгілі бір іс-әрекетті орындай алатын тиянақты мағынасы бар ең қарапайым сөйлемі **оператор** болып табылады. Тіл объектілерін, яғни программада пайдаланылатын мәліметтердің атаулары мен типтерін, олардың алғашқы мәндерін алдын ала тағайындау программаның *сипатталуы* болып саналады.

Енді Фаренгейт градустарын Цельсий градустарына ауыстыратын программа мәтінін келтірейік.

```
/* Градустарды Фаренгейт бірлігінен Цельсий
   бірлігіне алмастыру, f = 0, 20, ...,300 */
#include <stdio.h>
#include <conio.h>
main()
{
```

```
   int t0, tn, step;
   float f,c;
   t0 = 0;     /* төменгі температура */
   tn =300;    /* жоғарғы температура */
   step  = 20; /* өзгеру қадамы */
   clrscr();
   f = t0;
   printf("ГрадФ  ГрадЦ\n");
   while (f <= tn)
    {
     c = (5.0/9.0) * (f -32.0);
     printf("%4.0f %6.1f\n", f, c);
     f = f + step;
    }
    printf("\nАяқтау үшін ENTER басыңыз");
    getch();
   }
}
```

2.3. Си тіліндегі мәліметтер типтері

Мұнда мәліметтердің бірнеше негізгі типтері қолданылады. Олар:

- **char** (8 бит) – символдық, яғни таңбалық тип,
- **short** – қысқа бүтін сан,
- **int** – бүтін сан типі,
- **long** – екі еселенген бүтін сан,
- **unsigned** – таңбасыз бүтін сан,
- **float** – нақты сан типі, яғни жылжымалы нүктелі сандар,
- **double** – екі еселенген нақты сан типі,
- **long double** – ұзартылған, әрі екі еселенген нақты сан типі.

Алғашқы төрт тип бүтін сандарды сипаттау үшін қолданылады. Төмендегі кестеде әр түрлі типтердің ұзындықтары көрсетілген.

Си тілінің ішкі құрамындағы мәліметтер типтері мен олардың ені (ұзындығы) **2.5 кесте**

Мәлімет типі	Ұзындығы (бит – байт)	Сандар диапазоны
char	8 бит – 1 байт	-128 ... +127
unsigned char	8 бит – 1 байт	0 ... 255
short int	16 бит – 2 байт	-32768 ... 32767

unsigned short	16 бит – 2 байт	0 ... 65 535
int	16 бит – 4 байт	-32768 ... 32767
unsigned [int]	32 бит – 4 байт	0 ... 4294967295
long	32 бит – 4 байт	-2 147 483 648 ... 2 147 483 647
unsigned long	32 бит – 4 байт	0 ... 4 294 967 295
float	32 бит – 4 байт	$3.4 \cdot 10^{-38}$ ... $3.4 \cdot 10^{38}$
double	64 бит – 8 байт	$1.7 \cdot 10^{-308}$ ... $1.7 \cdot 10^{308}$
long double	80 бит – 10 байт	$3.4 \cdot 10^{-4932}$ ... $3.4 \cdot 10^{4932}$

2.4. Бүтін сан түріндегі мәліметтерді сипаттау

2.4.1. Int бүтін сандар типі

Int типін стандарт бекітпеген, ол компьютерге немесе компиляторға байланысты өзгеріп отырады. 16-разрядты процессорде ол 2 байт, ал 32-разрядтысында – 4 байт.

Егер int алдында short спецификатор сөзі тұрса, онда ол әрқашан 2 байт, ал егер спецификаторы long болса, 4 байт болады. Санға компьютер жадында берілген орынға қарай олардың мәндері өзгереді.

short int – 2 байт, оның диапазоны –32768 ..+32767;

long int – 4 байт, оның диапазоны –2 147 483 648..+2 147 483 647.

Int типі 16-разрядты компьютер үшін short int типімен бірдей, ал 32-разрядты компьютер үшін long int типімен бірдей.

Signed және unsigned модификаторлары да сандар шамасына әсер етеді, олар:

unsigned short int – 2 байт, оның диапазоны 0 ..65536;

unsigned long int – 4 байт, диапазоны 0..+4 294 967 295.

Айнымалыларды сипаттау кезінде бүтін тұрақтылар – костанталар мәндерін де көрсетуге болады. Мысалы:

```
int k=0;
```
(бір ғана сан сипатталған және оған мән берілген)

```
int k1,k3=0;
```
(біреуі тек сипатталған, екіншісіне мән берілген)

Unsigned типі **int**, **long, short** түйінді сөздерімен сипатталатын типтердің модификаторы ретінде қолданылады. Мысалы:

```
unsigned int sum=0;
```

char типін 0–255 аралығындағы таңбасыз бүтін сандарды сипаттауда қолдануға болады, ДК жадында бұларға бір байт орын бөлінген. Мысалы:

```
char c1;
char ck='k';
```

2.4.2. Char типі

Бұл тип мәндері реттелген символдар жиыны болып табылады. Әрбір символға бір бүтін сан сәйкес келеді, ол символ коды деп аталады. Символдық тип ені – 1 байт. Char типі де signed және unsigned спецификаторларымен қолданылады. Signed char типі диапазоны –128 .. 127. Unsigned char типін қолданғанда, оның мәндер диапазоны 0 .. 255 болады. Символдарды кодтау үшін ASCII (American Standard Code for International Interchange) стандарты негізге алынған. Символдардың 0 .. 31 кодтары басқару кодтарына жатады, олар тек енгізу-шығару кезінде ғана қолданылады.

Char типі символдарды олардың бүтін сан түріндегі кодтары арқылы сақтап, басқа шамалардың көрсетілген диапазоны сандарын да көрсету үшін де қолданыла алады.

2.4.3. Float және double жылжымалы нүктелі нақты сандар типтері

Нақты сандар компьютерде 2 бөліктен – дәреже мен мантиссадан тұрады. IBM-PC компьютерлерінде float типінің ені – 4 байт, оның бір разряды – сан таңбасы, 7 разряды – дәреже, 24 бит3 – мантисса.

Егер double типі аты алдында long сөзі тұрса, онда оған 10 байт орын беріледі.

Программалау практикасында көбінесе жылжымалы нүктелі нақты (аралас) сандар пайдаланылады.

Double типті сандар екі еселенген дәлдікпен 64 бит арқылы өрнектеледі. Double типінің ені – 8 байт, 1 бит – таңба, 11 бит – дәреже және 52 разряд – мантисса. Мантисса ені – санның дәлдігін, ал дәреже ені – оның диапазонын анықтайды.

Кейбір ЭЕМ-дерде мантисса үшін қосымша 32 бит беріледі де, санның дәлдігі артады. Ал кейде санның таңбасы мен дәрежесі үшін 32 бит беріліп, ол санның кескінделу диапазонын арттырады.

Мысалдар:

```
float s1, s2;
float sum=1.5;
```

Си тілінде объектілердің мәндерін байт арқылы анықтау үшін **sizeof** стандартты операторы қолданылады. Мысалы:

```
printf("double типіндегі мәліметтер ені %d байт\n",
        sizeof(double));
```

2.5 Символдық тіркестер (жолдар, қатарлар)

Си тілінде символдық тіркестерді сипаттау үшін арнайы тип жоқ, олар көбінесе char типтегі элементтерден тұратын жиым (массив) ретінде қарастырылады. Жолдық немесе тіркестік символдар ЭЕМ жадында көршілес ұяшықтарда сақталады да, олардың соңында '\o' символы тұрады. Символдар тіркесінің ұзындығын анықтау үшін strlen() функциясы қолданылады.

#define – символдарды немесе солардан тұратын константаларды анықтау мақсатында қолданылады. Мысалы:

```
#define NULL  '/o'
#define VNAME  "КазГУ"
```

2.6. Printf және scanf функциялары

Си тілінде сыртқы ортамен мәліметтер алмасу **<stdio.h>** енгізу-шығару функциялары кітапханасын пайдалану арқылы орындалады. Ол тақырып файлы ретінде былай жазылады:

```
#include <stdio.h>
```

printf() функциясы мәліметтерді экранға шығару үшін қолданылады. Оның жалпы жазылу түрі:

```
printf(<формат тіркесі>,<аргументтер тізімі>);
```

(<формат тіркесі> – қостырнақшамен (") шектеліп, аргументтердің қалай бейнеленетінін көрсетіп тұрады, экранға (баспаға) шығару алдында барлық аргументтер формат спецификациясына сәйкес түрлендіріледі, спецификация % символымен басталады және мәліметтер типін, оларды түрлендіру тәсілін көрсететін бір әріп жазылады. Объектілер ретінде айнымалылар, константалар, өрнектер қолданылуы мүмкін. Мысалы:

```
printf (" Пи санының мәні = %f\n", pi);
```

Формат тіркесінде мыналар болады:

1) мәтін ретінде шығарылатын символдар тіркесі;
2) түрлендіру спецификациялары;
3) басқару символдары.

Әрбір аргументке өз спецификациясы сәйкес келуі тиіс, олар:

%d – бүтін ондық сан шығарылуы тиіс,

%i – бүтін ондық сан шығарылуы тиіс,

%f – жылжымалы нүктелі нақты ондық сан ([-]dddd.dddd) жазылып шығады,

%e – жылжымалы нүктелі экспоненциалды сан ([-]d.dddde±dd) шығарылады,
%E – жоғарыдағы сияқты, тек е орнына E ([-]d.ddddE±dd) шығарылады,
%c – бір символ шығарылуы тиіс,
%s – символдар тіркесі (қатары) шығарылуы тиіс,
%g – нақты сан, сан ұзындығына қарай %e немесе %f қолданыла алады,
%u – таңбасыз ондық бүтін сан жазылып шығады,
%o – таңбасыз бүтін сегіздік сан шығады,
%x – таңбасыз бүтін он алтылық сан шығады.
\n – келесі жаңа жолға көшуді атқаратын басқару символы.
Мысалы:
%9i – бүтін сан ені 9 цифрдан тұрады, сан ені аз болса, оның сол жағында бос орындар орналасады.
%9.3f – нақты сан ені 9 цифрдан тұрады, оның 3 таңбасы бөлшекке беріледі, сан ені аз болса, оның сол жағында бос орындар орналасады.

Әрбір спецификация % символынан басталып, түрлендіру символымен аяқталады. Ол екеуінің ортасында мыналар тұруы мүмкін:

- минус таңбасы, аргумент мәні сол жақ шетке ығыстырылып жазылады.
- цифрлар, бүтін санның жалпы орналасу енін анықтайды. Сан осы енге немесе одан артық болып шығарылады. Егер аргумент ені көрсетілген еннен аз болса, онда ол бос орындармен толтырылып жазылады.
- нүктеге дейін санның жалпы ені, нүктеден соң бөлшек сандар ені көрсетіледі.
- L модификаторы, сәйкес аргумент мәні INT емес LONG екенін білдіреді.

scanf() енгізу функциясы жоғарыда қарастырылған түрлендіру спецификациясының көбін пайдаланады.

scanf (<формат тіркесі>,<аргументтер тізімі>);

Аргументтер ретінде адрес нұсқауыштары пайдаланылады. Мысалы:

```
scanf("%d%f", &x,&y);
```

Кейбір айырмашылықтарын атап өтейік.

1) %e және %f спецификациялары енгізу кезінде бірдей болып табылады;
2) short типті бүтін санды енгізу кезінде %h спецификациясы қолданылады.

ЕСКЕРТУ. Айнымалы адресін беру үшін адрестерді жазғанда, айнымалы адресін анықтау үшін & символы қолданылады. Ал тіркестік (жолдық) айнымалыны енгізгенде, & символы жазылмайды.

Енді санның дәрежесін есептейтін программа құрайық.

```
/* Санды дәрежелеу */
#include <conio.h>
#include <stdio.h>
#include <math.h>
main()
{
float x,y,s;
clrscr();
printf("\n x-ті және оның дәрежесін-у енгізіңіз :\n");
scanf("%f%f",&x,&y);
s=pow(x,y);
printf("\nНәтижесі s=%9.2f",s);
}
```

Бақылау сұрақтары

1. *Си тілінің шығу тарихы.*
2. *Си программасының жалпы құрамы мен құрылымы.*
3. *Препроцессор директивалары түсінігі.*
4. *Тілдің алфавиті құрамы. Операциялар таңбалары, айыру белгілері, қатынас таңбалары және арнайы символдар.*
5. *Басқару тізбектері не үшін қажет және олар қалай жазылады?*
6. *Тілдің қарапайым объектілері.*
7. *Си тіліндегі сандар мен айнымалылар, олардың жазылу жолдары.*
8. *Идентификатор дегеніміз не?*
9. *Өрнек ұғымы.*
10. *Тілдің түйінді сөздері.*
11. *Си тіліндегі мәліметтер типтері мен олардың ені.*
12. *Оператор дегеніміз не, олар қандай топтарға жіктеледі?*
13. *Бүтін сан типтері мен оларды қолдану ерекшеліктері.*
14. *Символдық тип және оны қолдану жолдары.*
15. *Нақты сандар типтері және оларды пайдалану.*
16. *Символдық тіркестерді сипаттау.*
17. *Мәліметтерді пернелерден енгізу функциясы және оның жазылу форматтары.*
18. *Нәтижені экранға шығару жолдары.*
19. *Формат арқылы жазылатын енгізу-шығару функциясы қандай қызмет атқарады?*
20. *Сандардың және олардың арасында қалдырылатын бос орындардың енін қайтіп көрсетеді?*
21. *Енгізу функциясының жазылуы және оның түрлендіру спецификациялары. printf() және scanf() функцияларының негізгі айырмашылықтар*

3. СИ ТІЛІНДЕ ҚОЛДАНЫЛАТЫН НЕГІЗГІ ОПЕРАТОРЛАР

Арифметикалық операциялар символдар арқылы жазылады. Си тілінде мынадай операциялар бар:

* – көбейту, / – бөлу, % – модуль бойынша бөлу (қалдықты анықтау),

+ – қосу, – – азайту.

Модуль бойынша бөлу бүтін санды бүтін санға бөлген кездегі қалдықты анықтайды. Мысалы: 20 % 3 = 2.

3.1. Меншіктеу операторы

Меншіктеу операторы символдар арқылы жазылады. Кез келген ; таңбасымен аяқталатын өрнек меншіктеу операторы болып табылады. Өрнектің бір түріне бос оператор жатады, ол жай **;** операторы.

Си тілінде меншіктеу операторының бірнеше түрі бар. Жалпы меншіктеу операторының жазылу форматы мынадай болады:

<айнымалы> = <айнымалы> <операция> <өрнек>;

Мұны Си тілінде қысқаша былай жазуға болады:

<айнымалы> <операция> = <өрнек>;

Төменде бірнеше мысал келтірілген.

a=a+b; → a+=b; **a=a*b; → a*=b;**

a=a-b; → a-=b; **a=a/b; → a/=b;**

Си тілінде тізбектеле жазылған меншіктеу операцияларын да қолдануға болады. Мысалы:

sum = a = b;

Мұнда меншіктеу операторы оңнан солға қарай орындалады, яғни b-ның мәні a-ға меншіктеледі, ал a-ның мәні sum-ға меншіктеледі.

Меншіктеу операторын былай да жазуға болады:

1) **a = (b = 1) +2;**

мұнда **a=3, b=1**.

2) **a = b = 1 + 2;**

ал мұнда **a = 3, b = 3**.

Дөңгелек жақшаға алынған кез келген меншіктеу операторы анықталған мәні бар өрнек болып табылады, мысалы: **((s=13+12)<=30)** деген өрнек ақиқат мәнді болып табылады.

Арттыру немесе кеміту (инкремент және декремент) операциялары ++ және – – түрінде жазылады. Бұлар кез келген операндтың мәндерін бірге арттыруға немесе кемітуге мүмкіндік береді.

Инкремент, яғни арттыру операциясы (++) және декремент (--), кеміту операциясы айнымалы (тек айнымалы мәнін) мәнін бірге арттырады немесе кемітеді. *Олар айнымалы мәнін өзгертеді,* яғни жасырын түрдегі

меншіктеу амалы болып табылады. Кейде олар жеке оператор түрінде жазылады:

i++; немесе **++i;**

Бұл екеуі де мынадай амалмен бірдей болып саналады

i = i + 1;

Бұл екеуін өрнектерде жиі қолданады. Мысалы:

sum = sum + x * --i;

Инкремент пен декремент екі формада жазылады: *префикстік* және *постфикстік*. Постфикстік арттыру былай жазылады: **x++** , ал префикстік арттыру былай жазылады: **++x .** Префикстік амалдар негізгі операция *алдында*, ал постфикстік амалдар негізгі амалдан *кейін* орындалады.

Постфикстік формада **x** айнымалысының мәні оны қолданғаннан кейін өзгереді, ал префикстік формада – айнымалы мәні оны қолданғанға дейін өзгереді, яғни бір деген санға артады. Мысалы:

```
k=10;
x=k++;  /* x=10  k=11  */
x=++k;  /* x=12  k=12  */
```

Арттыру/кеміту операцияларын өрнек ішінде де орындау мүмкіндігі бар. Мысалы:

sum=a+b++; /*алдымен a,b қосылады, сонан соң b 1-ге артады */

sum=a+ ++b; /*алдымен b 1-ге артады, сосын барып a,b қосылады */

Арттыру/кеміту операцияларының приоритеттері өте жоғары, тек жақша ішіндегі операциялардың приоритеті олардан жоғары болады.

Құрама оператор бірнеше операторды жүйелік жақшаға алып біріктіру үшін қолданылады. Ол шартты және циклдік операторларда жиі қолданылады. Мысалы:

```
#include <stdio.h>
#include <conio.h>
main()
{
  int a = 5, b = 6, c = 7, d;
  clrscr();
  d = (a++ - (--b)) + (c - a--); //d=2  a=5
  printf("d=%i a=%d",d,a);
  getch();
}
```

3.2. Типтерді түрлендіру

Егер өрнекте әр түрлі типтегі сандар мен айнымалылар қолданылса, онда олар жалпы бір типке түрлендіріледі. Біз қарастырған барлық негізгі типтер ішінде төменнен жоғары қарай бағытталған түрлендірілу реттілігі

бар. Егер оларды оңған солға қарай реттеп орналастырсақ, мынадай болып шығады:

char → short → int → long → float → double

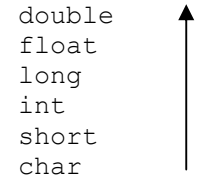
Оң жақтағылары сол жақтағылардан гөрі жоғары дәрежелі болып табылады.

Егер **char** мен **short** типтері араласса, нәтижесі – **chort** болады,
ал **short** пен **int** типтері араласса, нәтижесі – **int** болады,
ал **int** пен **long** типтері араласса, нәтижесі **long**,
ал **long** пен **float** типтері араласса, нәтижесі **float**,
ал **float** пен **double** типтері араласса, нәтижесі **double** болады.

Егер екі-үш тип араласып, ең үлкен дәрежелісі – **float** болса, әрқайсысы да және нәтиже де осыған келтіріледі.

Компилятор типтерді автоматты түрде түрлендіру үшін төмендегі негізгі ережелер жиынын пайдаланады:

1. Егер операция екі түрлі типтегі мәліметтер үшін орындалатын болса, онда олар осы мәліметтер типтерінің арасындағы “жоғарғы” типке келтіріледі.
2. “Жоғары” типтен бастап, “төмен” типке дейін реттелген типтер аттарының тізбегі келесідей түрде көрсетіледі:



Меншіктеу операторында оң жақта орналасқан өрнектің есептелген нәтижесі осы оператордың сол жағына жазылған айнымалының типіне келтіріледі. Осындай процесс типтің “жоғарысына” немесе “төменіне” келтірілуі мүмкін.

Мысалы:

```
#include <stdio.h>
#include <conio.h>
main ()
{
char ch;
int i; float fl;
fl=i=ch='A';
printf("ch=%c i=%d fl=%6.2f\n",ch,i,fl);
               // ch=A i=65 fl= 65.00
ch=ch+1;       // ch=66
i=fl+2*ch;     // i=65.00+2*66=197
fl=2.0*ch+1;   // fl=2*66+1=133
```

```
printf("ch=%c i=%d fl=%6.2f\n",ch,i,fl);}
              // ch=B i=197 fl=133.00
```

Келтіру операциясы. Жоғарыда көрсетілген типтердің түрлендірілуі автоматты түрде орындалады. Мәліметтердің көрсетілген қажетті типіне келтіру үшін Си тілінде арнайы бір тәсіл бар. Бұл тәсілде типтерді бірыңғайлау үшін, айнымалының алдында дөңгелек жақшада қажетті типтің аты жазылады. Жалпы түрге келтіру операциясы мынадай болып жазылады: **(тип)өрнек**. Мысалы:

```
int m;
float x,y;
y=pow(x,2)+sqrt((double)m);
```

3.3. Программа жұмысын басқару операторлары

Программа жұмысын басқару операторлары программаның басқарушы конструкциясы деп аталады. Олар:

- құрама операторлар;
- таңдау операторлары;
- цикл операторлары;
- көшу операторы.

Құрама операторларға жай құрама операторлар және блоктар жатады. Екеуі де жүйелі жақшаға алынып жазылады. Блокта жай құрама операторларға қарағанда, айнымалыларды сипаттау жолдары болады. Мысалы:

```
{  n++;                          жай құрама оператор
   summa+=n;
}
{
   int n=0;
   n++;                          бұл блок
   summa+=n;
}
```

Қатынас операциялары. Қатынас операциялары екі мәнді салыстыру үшін қолданылады. Си тіліндегі қатынас операциясының тізбегі мынадай: ==, !=, <, <=, >, >= .

Егер қарастырылатын қатынас нәтижесі ақиқат болса, шарттық өрнектің мәні 1-ге тең болып саналады. Егер ол жалған болса, шарттық өрнек мәні 0-ге тең болады. Қатынас операциясының приоритеті арифметикалық операцияларға қарағанда, төмен және меншіктеу операторымен салыстырғанда жоғары болады. Ал қатынас операцияларын приоритеттеріне сәйкес етіп екі топқа бөлуге болады. Мұнда соңғы 4

операцияның приоритеті 1-ші және 2-ші операциялар приоритетімен салыстырғанда жоғары болады.

Логикалық оператор. Екі немесе бірнеше шарттық өрнектерді біріктіру үшін логикалық оператор қолданылады. Си тілінде келесі логикалық операторлар бар:

1) және (&&) операциясы; f1 && f2
2) немесе || (or) операциясы; f1 || f2 (| коды - 124)
3) терістеу ! (not) операциясы. !f1

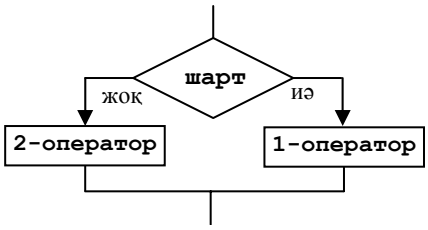
Терістеу (емес, қарсылық) операциясының приоритеті өте жоғары, одан тек жақша ішіндегі мәндердің проритеті жоғарылау болады. && операциясының приоритеті || (or) операциясымен салыстырғанда жоғары, ал осы екеуінің приоритеттері қатынас операцияларымен салыстырғанда төмен, меншіктеу операциясынан жоғары болады.

Таңдау операторлары – бұлар шартты оператор және ауыстырғыш.

3.4. Шартты оператор

Шартты операторлың орындалу схемасы мен жалпы жазылу түрі мынадай:

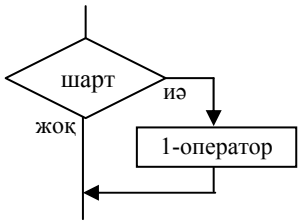
```
if (шарт)
     1-оператор;
  else
     2-оператор;
```



3.1-сурет. Шартты оператор схемасы

Мұнда жақшадағы шарт түріндегі өрнек ақиқат болса, 1-оператор орындалады, әйтпесе 2-оператор атқарылады. Операторлар қарапайым немесе құрама болып жазыла береді. Операторлың қысқаша жазылу түрі:

```
if (шарт)
     1-оператор;
```



3.2-сурет. Шартты операторлың қысқаша түрі

Кейде қабаттасқан шартты операторлар кездеседі, мысалы:

```
if (1-шарт)
    1-оператор;
 else if (2-шарт)
         2-оператор;
       else
         3-оператор;
```

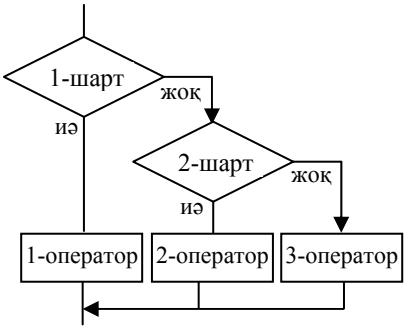
Мұнда егер 1-шарт ақиқат болса, 1-оператор орындалады, егер 1-шарт жалған

болып, 2-шарт ақиқат болса, 2-оператор орындалады, ал 1-шарт және 2-шарт жалған болса, 3-оператор атқарылады.

Ондағы кез келген **else** түйінді сөзі (keyword) оның алдында ең жақын тұрған **if** операторына қатысты болып саналады.

Мысалы, берілген *x, y* – екі санның үлкенін анықтау үшін жазылған шартты операторды былай жазуға болады:

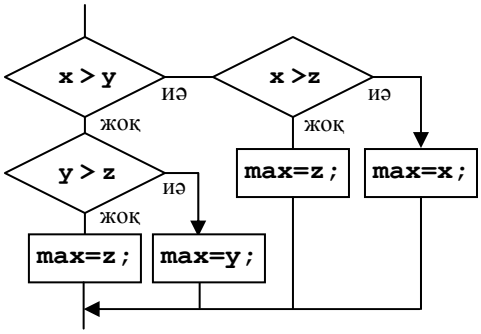
```
if  (x>y)  max=x;
     else  max=y;
```



3.3-сурет. Қабаттасқан шартты операторлар схемасы

Ал *x, y, z* сияқты үш санның үлкенін табу үшін, қабаттасқан шартты операторлар жазылады.

```
if (x>y)
 {if (x>z) max=x;
    else   max=z;}
  else if (y>z) max=y;
          else  max=z;
```



3.4-сурет. Үш санның үлкенін табу схемасы

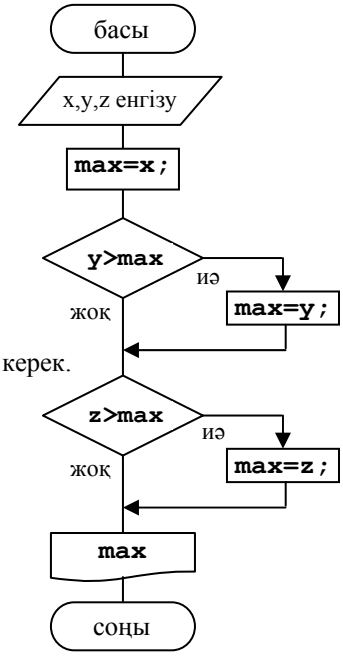
Логикалық ЖӘНЕ операциясын пайдалана отырып, бұл есепті мынадай түрде де жаза аламыз.

```
if ((x>y)&&(x>z)) max=x;
  else if ((y>z)&&(y>x))
          max=y;
   else   max=z;
```

1-мысал. Осы алгоритмді толығынан қысқаша шартты оператор арқылы орындаудың блок-схемасы (3.5-сурет) мен программасын келесі түрде жазып шығайық.

```
#include <stdio.h>
#include <conio.h>
main ()
```

```
{
 clrscr();
 int max,x,y,z;
 printf("3 бүтін сан енгізіңіз:");
 scanf("%d%d%d",&x,&y,&z);
 max=x;
 if (y>max) max=x;
 if (z>max) max=z;
 printf("max=%d",max);
 getch();
}
```



3.5-сурет. Үш санның үлкенін табу схемасының екінші түрі

2-мысал. Формула арқылы берілген төмендегі ***y*** функциясын есептеу программасын құрастыру керек.

$$y = \begin{cases} x+2, & \text{егер} \quad x<0 \\ 2x^3, & \text{егер} \quad x\geq 0 \end{cases}$$

```
#include <stdio.h>
#include <conio.h>
main ()
{
 clrscr();
 float x,y;
 printf("x нақты санын енгізіңіз:");
 scanf("%f",&x);
 if (x<0) y=x+2;
    else  y=2*x*x*x;
 printf("\ny=%f",y);
 getch();
}
```

3- мысал. Программаға бір жыл нөмірін енгізіп, сол жылдың кәбисә (366 күн) немесе қарапайым жыл (365 күн) екендігін анықтау керек. Ол үшін жылды төртке бөлеміз, егер қалдық 0-ге тең болса, ол кәбисә жыл, әйтпесе қарапайым жыл болады.

```
#include <stdio.h>
#include <conio.h>
main ()
{ int gil;
   int r; /* gil-ды 4-ке бөлгендегі қалдық */
   clrscr();
   printf("Жылды, мысалы, 2007 енгізіп, Enter басыңыз: ");
```

```
   scanf("%i",&gil);
   r=gil % 4;
   if (r)
         printf("%i жыл - қарапайым \n", gil);
   else
         printf("%i жыл – кәбисә \n", gil);
   printf("\nАяқтау үшін Enter басыңыз");
   getch();
}
```

4- мысал. Квадрат теңдеуді шешу программасын құру керек.

```
/* Квадрат теңдеуді шешу */
#include <stdio.h>
#include <conio.h>
#include <math.h>
main ()
{
float a,b,c;
float x1,x2,d;
clrscr();
printf("\n * Квадрат теңдеуді шешу * \n");
printf(" a,b,c мәндерін енгізіп, Enter басыңыз: ");
scanf("%f%f%f",&a,&b,&c);
d=b*b-4*a*c;
if (d < 0)
      printf("Теңдеудің шешуі жоқ \n");
else {
      x1=(-b+sqrt(d))/(2*a);
      x2=(-b-sqrt(d))/(2*a);
      printf("Теңдеу түбірлері: x1=%3.2f
              x2=%3.2f\n", x1,x2);
     }
printf("\nАяқтау үшін Enter басыңыз");
getch();
}
```

5- мысал. Төмендегі функция мәнін кез келген **x** үшін есептеу керек.

$$y = \begin{cases} 0, & егер \quad x \le 0 \\ x^2 - x, & егер \quad 0 < x \le 1 \\ \sin \pi x^2, & егер \quad x > 1 \end{cases}$$

```
#include <stdio.h>
#include <math.h>
#define pi 3.14159
main ()
{
float x,y;
clrscr();
printf("Нақты сан түріндегі x мәнін енгізіңіз: ");
scanf("%f",&x);
if (x <= 0)
      y = 0;
else if (x <= 1)
      y = x*x - x;
     else
      y = sin(pi * x*x);
printf("x = %f болғанда, y = %10.6f \n",x,y);
return(0);
}
```

6- мысал. Төменде алынған баллға сәйкес бағаны анықтау программасы келтірілген.

балл	баға
90...100	A
75...89	B
60...74	C
50...59	D
0...49	F

$$baga = \begin{cases} A, \text{ егер } 90 \le ball \le 100 \\ B, \text{ егер } 75 \le ball < 90 \\ C, \text{ егер } 60 \le ball < 75 \\ D, \text{ егер } 50 \le ball < 60 \\ F, \text{ егер } 0 \le ball < 50 \end{cases}$$

```
#include <stdio.h>
#include <conio.h>
main ()
{
int  ball;
```

```
char baga;
clrscr();
printf("Балл мөлшері: ");
scanf("%i",&ball);
if (ball >= 90)
      baga = 'A';
else if (ball >= 75)
      baga = 'B';
else if (ball >= 60)
      baga = 'C';
else if (ball >= 50)
      baga = 'D';
else
      baga = 'F';
printf("Бағасы - %c, балл мөлшері - %i
\n",baga,ball);
printf("\nАяқтау үшін Enter басыңыз");
getch();
}
```

3.4.1 Шартты операция

Шартты операция (? :) шартты өрнек жазуға мүмкіндік береді, яғни берілген шартқа байланысты әр түрлі мән қабылдайтын *шартты өрнектер* құрады. Бұл операция үшорынды болып табылады. Егер оның шарты (бірінші операнд) ақиқат болса, өрнек мәні екінші операндқа тең; егер жалған болса, онда – үшіншіге тең. Мысалы:

```
max_ab = a > b? a : b;
```

3.5. Switch көп нұсқалы таңдау операторы

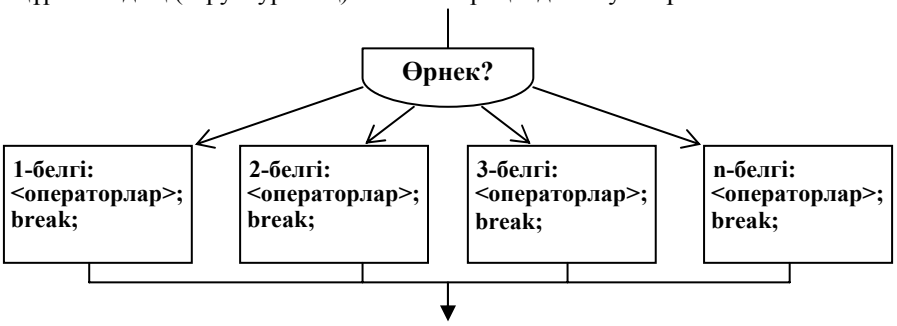
Программада кездесетін бірнеше нұсқаның бірін таңдап алу керек болған жағдайда, *switch* ауыстырғыш операторы қолданылады. Оператордың орындалу схемасы төмендегі суретте келтірілген. Оның жалпы жазылуы:

```
switch <бүтін типті өрнек>;
{
 case белгі1:  операторлар;
 case белгі2:  операторлар;
 ..............
 [default: операторлар;]
}
```

Мұнда switch сөзінен кейінгі өрнек мәні есептеледі, ол бүтін санды (char типі де) типте болуы тиіс. Сол мән **case** сөздерінен кейін

жазылған константалар мәндерімен салыстырылады. Егер олардың біріне тең болса, сол жол орындалады, жол соңында көшу операторы болмаса, келесі жолдар толық орындалады. Ал бір жолды орындап болған соң, *switch* операторынан шығу үшін **break** операторы қолданылады. Егер **switch** сөзінен кейінгі өрнек мәні ешбір константамен сәйкес келмесе, онда **default** сөзінен кейінгі операторлар атқарылады. Кейде **default** сөзі болмауы да мүмкін.

Default сөзі болмаса, онда *switch* операторынан кейінгі келесі операторлар орындала береді. *Switch* операторындағы өрнек түрінде нақты типтегі мәліметтерді, сөз тіркестерін (жолдарды) пайдалануға болмайды. Кейде бүтін мәндермен үйлестірілген мәліметтердің құрылымдық (структуралық) элементтері қолданылуы мүмкін.



3.6-сурет. Switch операторының орындалу схемасы

7-мысал: Екі бүтін сан енгізіп, олармен арифметикалық 4 амалдың бірін орындау қажет.

```
#include <stdio.h>
#include <conio.h>
char symbol;
int x,y,z;
clrscr();
printf("Екі бүтін сан енгізіңіз: ");
scanf("%i%i",&y,&z);
printf("Қандай амал орындау керек: ");
scanf("%s",symbol);
switch (symbol)
{
 case '-' : x=y-z; break;
 case '+' : x=y+z; break;
 case '*' : x=y*z; break;
 case '/' : x=y/z; break;
```

```
 default:  printf (”белгісіз операция\n”);
 printf("\nНәтижесі - %d\n",x);
 getch();
}
```

8-мысал: Шығыс календары бойынша жылға сәйкес жануар атын анықтау.

```
#include <stdio.h>
#include <conio.h>
main()
{ int  gil;
  printf("Жылды енгізіңіз: \n");
  scanf("%i",&gil);
  switch (gil % 12)
  {
    case 0 : printf("Мешін жылы");break;
    case 1 : printf("Тауық жылы"); break;
    case 2 : printf("Ит жылы"); break;
    case 3 : printf("Доңыз жылы"); break;
    case 4 : printf("Тышқан жылы"); break;
    case 5 : printf("Сиыр жылы"); break;
    case 6 : printf("Барыс жылы"); break;
    case 7 : printf("Қоян жылы"); break;
    case 8 : printf("Ұлу жылы"); break;
    case 9 : printf("Жылан жылы"); break;
    case 10: printf("Жылқы жылы"); break;
    case 11: printf("Қой жылы"); break;
   default : printf("Таңбасыз бүтін сан
                     енгізіңіз");
  }
  printf("\nENTER басыңыз");
  getch();
}
```

Switch орындалуы кезінде цикл аяқталмай-ақ одан шығып, қалған операторларды аттап өтіп, осы цикл параметрінің келесі мәніне көшу үшін *continue* операторы қолданылады, яғни циклдің келесі итерациясына – қадамына басынан бастап ауысу жүзеге асырылады.

Мысал:

```
#include <stdio.h>
main()
{
      int i;
      printf("\nБүтін сан енгіз: )";
```

```
        scanf("%i",&i);
        switch(i)
        {case 1: printf("\nСан бірге тең!");
        case 2: printf("\n2*2=%d",i*i);
        case 3: printf("\n3*3=%d",i*i;break;
        case 4: printf("\n Сан төртке тең!";
        default: printf("\nАяқталды";
        }
}
```

Бұл программаның жұмыс нәтижесі:

1 енгізілгенде мыналар шығарылады:

Сан бірге тең!

2*2=1

3*3=1

2 енгізілгенде мыналар шығарылады:

2*2=4

3*3=4

3 енгізілгенде мыналар шығарылады:

3*3=9

4 енгізілгенде мыналар шығарылады:

Сан төртке тең!

Қалған сандар енгізілсе:

Аяқталды!

сөзі шығарылады.

Бақылау сұрақтары

1. Си тілінде қандай операциялар бар?
2. Меншіктеу операторының түрлері.
3. Меншіктеу операторының жазылу форматтары.
4. Арттыру немесе кеміту (инкремент және декремент) операциялары.
5. Префикстік және постфикстік операциялар.
8. Құрама операторлар қалай ұйымдастырылады?
9. Бос оператор деген не?
10. Типтер ішінде төменнен жоғары қарай бағытталған түрлендірілу реттілігі.
11. Келтіру операцияларының жазылуы.
12. Программа жұмысын басқару операторлары
13. Бірнеше шарттық өрнектерді біріктіретін логикалық операторлардың қолданылуы.
14. Қандай жағдайларда шартты оператор пайдаланылады?
6. Шартты операторлардың жазылуының қандай түрлері бар? Олардың мағына жағынан ерекшелігі неде?
10. Шартты операторға мысалдар келтіріңдер.
11. Шартты операторды пайдаланып y=1/(x-1)+1/(x-2) мәнін есептейтін программа құрыңдар.

12. Қабаттасқан шартты операторлардың жазылуы.
13. Шартты операция дегеніміз не?
14. Көп нұсқалы таңдау операторы не үшін қажет? Оның жазылу форматы қандай?
15. Көп нұсқалы таңдау операторына мысал келтіріңдер.

ТАПСЫРМАЛАР

Меншіктеу операторларына берілген есептер

1. А-ның берілген мәндері а) a = 1.0; ә) a = 4; б) a = 5 болған кездердегі x мәндерін анықтау керек:
   *b = 2.4*a;*
   *x = (a+b)/a*b-a;*
   x = a/b % b;
   *b = a*a-2*a;*
   a = (b + 2)(b - 1);*
2. Төмендегі операторларда жіберілген қателерді табыңдар:
   а) 2-x = k + 4;
   ә) x =x < 4;
   б) x = 3,74*a;
   в) 3*k = m;
   г) -w = a + b;
   д) x = a/- b;
   е) y = y>2
   ж) 5 = a - b;
   з) p = 5.5 % 2;
   и) x = sin$x$ + cos$x$;
3. Төмендегі өрнектерді алгоритмдік тілдерде жазыңдар:

   а) $y = \dfrac{\sqrt{x^2+1} - \sqrt{5x-1}}{6x}$;

   ә) $y = \dfrac{\sin 3{,}5 + tg\,0{,}6}{e^{2{,}7} + \ln 4{,}5}$;

   б) $y = \dfrac{\sin(\cos x) + 5{,}4 \sin \sqrt{|x|}}{\cos x - \sin x}$
4. Кубтың қабырғаларының ұзындығы берілген. Оның көлемі мен қабырғасының бетінің ауданын табыңдар.
5. Тікбұрышты үшбұрыштың катеттері берілген. Оның гипотенузасы мен ауданын табыңыздар.
6. Теңбүйірлі үшбұрыштың жақтары берілген. Үшбұрыштың ауданын табыңыздар.
7. x_1, y_1 және x_2, y_2 координаталарымен берілген нүктелердің ара қашықтығын табыңыздар.

8. Нақты x саны берілген. Тек көбейту, қосу және азайту амалдарын қолданып $2x^4 - 3x^3 + 4x^2 - 5x + 6$ есептеңіз. 4 көбейту, 4 қосу және 4 азайту амалын қолдануға болады.

6. Нақты x саны берілген. Тек көбейту, қосу және азайту амалдарын қолданып $1 - 2x + 3x^2 - 4x^3$ и $1 + 2x + 3x^2 + 4x^3$ есептеңіз. Тек 8 операция қолдануға болады.

11. x, y, z мәндері берілген, a және b мәндерін есептеңіздер.

$$a = \frac{2 * \cos(x - \pi / 6)}{1 / 2 + \sin^2 y}, \quad b = 1 + \frac{z^2}{3 + z^2 / 5}$$

12. x, y, z мәндері берілген, a және b мәндерін есептеңіздер.

$$a = \frac{1 + \sin^2(x + y)}{2 + \left| x - 2 * x / (1 + x^2 * y^2) \right|}, \quad b = \cos^2\left(\operatorname{arctg}\frac{1}{z}\right)$$

13. x, y, z мәндері берілген, a және b мәндерін есептеңіздер.

$$a = \ln\left|(y - \sqrt{|x|}) * \left(x - \frac{y}{z + x^2 / 4}\right)\right|, \quad b = x - \frac{x^3}{3!} - \frac{x^5}{5!}$$

14. a және b нақты сандары берілген. Осы сандардың қосындысын, айырмасын және көбейтіндісін табыңыздар.

16. Екі нақты оң сан берілген. Осы сандардың арифметикалық және геометриялық ортасын табыңыздар.

17. Екі нақты сан берілген. Осы сандардың абсолют шамасының арифметикалық және геометриялық ортасын табыңыздар.

18. Үшбұрыштың төбелерінің координатасы берілген. Үшбұрыштың периметірі мен ауданын табыңыздар.

19. Берілген *a, d, n* мәндері бойынша арифметикалық прогрессияның мүшелерінің

$$a, a + d, \ldots, a + (n - 1)d$$

қосындысын табыңыздар

20. Тікбұрышты үшбұрыштың гипотенузасы мен катеті берілген. Оның екінші катеті мен оған іштей сызылған шеңбердің радиусын табыңыздар.

21. Бір бірінен *r* арақашықтықта орналасқан, массалары m_1 және m_2 екі дене арасындағы *F* тартылыс күшін анықтаңыздар.

22. x, y нақты сандары берілген. Тек 8 көбейту, 8 қосу және 8 азайту амалдарын қолданып, $3x^2y^2 - 2xy^2 - 7x^2y - 4y^2 + 15xy + 2x^2 - 3x + 10y + 6$ есептеңіздер.

23. Тастың h биіктіктен жер бетіне құлау уақытын анықтаңыздар.

24. Берілген x, y, z мәндері бойынша a, b мәндерін есептеңіздер.

$$a = \frac{e^{x+y} + \sin(x) + \cos(y)}{3x^2 + 2x + 6}, \qquad b = \ln\left|\frac{z+1}{x}\right| + z$$

25. *a, b, c* нақты оң сандары берілген. Ұзындықтары *a, b, c*-ға тең үш қабырғалары бойынша үшбұрыш тұрғызып, оның бұрыштарын табу керек.

26. Үшбұрыштың ішкі бұрыштары және сырттай сызылған шеңбердің радиусы берілген. Оның қабырғаларының ұзындықтарын табыңыздар.

27. Шеңбердің ұзындығы белгілі. Осы шеңбермен шектелген дөңгелектің ауданын табыңыздар.

28. Ішкі радиусы 20-ға, ал сыртқы радиусы *r* ($r > 20$) болатын сақинаның ауданын табыңыздар.

29. R_1, R_2, R_3 кедергілері параллель жалғанған. Осы тізбектің жалпы кедергісін табыңыздар.

30. Берілген c, d бойынша төмендегі өрнекті есептеңіз.

$$\left|\frac{\sin^3\left|cx_1^3 + dx_2^2 - cd\right|}{\sqrt{(cx_1^3 + dx_2^2 - x_1)^2 + 3.14}}\right| + tg(cx_1^3 + dx_2^2 - x_1),$$

Мұнда x_1, x_2 $x^2 - 3x - |cd| = 0$ теңдеуінің түбірлері.

Тармақталу операторына берілген есептер

1. Тармақты алгоритмдерді программалау тәсілдерін пайдаланып, төмендегі функциялардың мәндерін есептейтін программа жазыңдар:

а) $$y = \begin{cases} \dfrac{e^{\frac{\sin x}{2}} + \sin e^{\frac{x}{4}}}{x^2 + 2}, & \text{егер } x < 2 \\ 2x^2 + \dfrac{1}{\sqrt{3x}}, & \text{егер } x \geq 2 \end{cases}$$

ә) $$y = \begin{cases} \sin^2 2x + 5x^2, & \text{егер } x > 1{,}66 \\ \dfrac{61\sqrt{x} - 17}{\sqrt{4 + x^2 + \cos^2 4x}}, & \text{егер } x \leq 1{,}66 \end{cases}$$

б) $y = \begin{cases} \dfrac{e^x + 1}{e^{x^2}}, & \textit{егер} \quad x < -2{,}4 \\ (x + \sin 4x) + \lg x^2, & \textit{егер} \quad x \geq -2{,}4 \end{cases}$

в) $y = \begin{cases} \sqrt{(\sin x + 1)} + \lg x^2, & \text{егер} \quad x < -1 \\ \dfrac{\sin x + \cos x}{\cos x}, & \text{егер} \quad -1 \leq x < 3 \\ e^{-\sin x} + \sin x, & \text{егер} \quad x > 3 \end{cases}$

г) $z = \begin{cases} 5\sin y + \cos y, & \text{егер} \quad y < 1 \\ \dfrac{y^2 - 2y - 5}{e^y}, & \text{егер} \quad 1 \leq y < 4 \\ \sqrt{y^2 + 5} + \lg y, & \text{егер} \quad y \geq 4 \end{cases}$

мұндағы $y = \begin{cases} \lg x + \sqrt{x}, & x \leq 2 \\ \lg 2x + \sqrt{3x}, & x > 2 \end{cases}$

д) $y = \begin{cases} \sqrt{x} + \sin x, & \text{егер} \quad x > 0{,}4 \\ \sqrt{2 - x} + \cos x, & \text{егер} \quad -1 < x \leq 0{,}4 \\ e^{5x-1} + \lg x, & \text{егер} \quad x \leq -1 \end{cases}$

2. Жазықтағы нүктенің координаталары бойынша оның қай ширекте жататындығын анықтайтын программа құрыңдар.

3. Жазықтықта үш нүкте берілген: $A(x_1,y_1)$,$B(x_2,y_2)$ және $C(x_3,y_3)$. Осы үш нүкте арқылы үшбұрыш құруға бола ма? Егер болса, Герон формуласын пайдаланып оның ауданын есептейтін программа құрыңдар.

4. Квадрат теңдеуді шешу программасын құрыңдар ($ax^2+bx+c=0$; $a \neq 0$)

5.Үшбұрыштың төбелері координаталар арқылы берілген: $A(x_1,y_1)$, $B(x_2,y_2)$, және $C(x_3,y_3)$. Бұл үшбұрыш тең қабырғалы, тең бүйірлі болатынын анықтау программасын құрыңдар.

6. Жазықтықта екі нүкте N (x_1,y_1) және M(x_2,y_2) берілген. Бұлардың қайсысы координаталар басына жақын болатынын анықтайтын программа құрыңдар.

7. Бүтін *n* санын берілген, ол бүтін *m* (*m*<*n*) санына қалдықсыз бөлінетінін анықтайтын программа құру керек.

8. Егер берілген *a* саны жұп болса, онда *p* атауына *true*, ал тақ болса *false* мәнін меншіктейтін программа құрыңдар.

9. Кез келген айдың бірінші жұлдызы аптаның қай күні екені белгілі болғанда, сол айдың енгізілген кез келген күнінің аптаның қандай күні болатынын анықтау программасын құру қажет.

10. Апта күндерінің реттік нөмірі бойынша олардың аттарын анықтайтын программа құру керек.

11. Нақты x, y (x ≠ y) берілген. Кішісін олардың жарты қосындысымен, ал үлкенін - екі еселенген көбейтіндісімен алмастырыңыз.

12. Үш нақты сан берілген. Теріс емес сандарды квадраттаңыздар.

13. Егер берілген нақты x, y, z сандарының қосындысы 1-ден кем болса, онда бұл үш санның ең кішісін қалған екі санның жарты қосындысымен алмастырыңыз, кері жағдайда x және y-тің кішісін қалған екеуінің жарты қосындысымен алмастырыңыз.

14. Нақты *a*, b, c, d сандары берілген. Егер *a*<=b<=c<=d болса, онда әр санды ең үлкен санмен алмастырыңыз, егер *a*>b>c>d болса, садарды өзгеріссіз қалдырыңыз, кері жағдайда барлық сандарды олардың квадратымен алмастырыңыз.

15. Нақты x, y, z сандары берілген. Егер x y-ке қалдықсыз бөлінсе және y z-ке қалдықсыз бөлінсе, онда барлық сандарға 1-ді қосыңыз, кері жағдайда барлық сандарды нөлге теңестіріңіз.

16. Нақты *a*, b, c, d сандары берілген. Осы сандардың терістерін квадраттап, ал оң сандарын түбірден шығарыңыз.

17. Нақты *a*, b, c, d сандары берілген. Егер кем дегенде бір сан нөлге тең болса, ол жайлы экранға мәлімет шығарыңыз, кері жағдайда a-ның b-ға және c-ның d-ға қалдықсыз бөлінетіндігін тексеріңіз.

18. Бүтін *a*, b, c сандары берілген. Егер *a*<=b<=c болса, онда барлық сандарды олардың квадратымен алмастырыңыз; егер *a*>b>c болса, онда әр санды ең үлкен санмен алмастырыңыз, кері жағдайда барлық сандардың таңбасын кері таңбаға ауыстырыңыз.

19. Нақты x, y, z сандары берілген. max (x + y + z, x*y*z) + 10 өрнегін есептейтін программа жазыңыз.

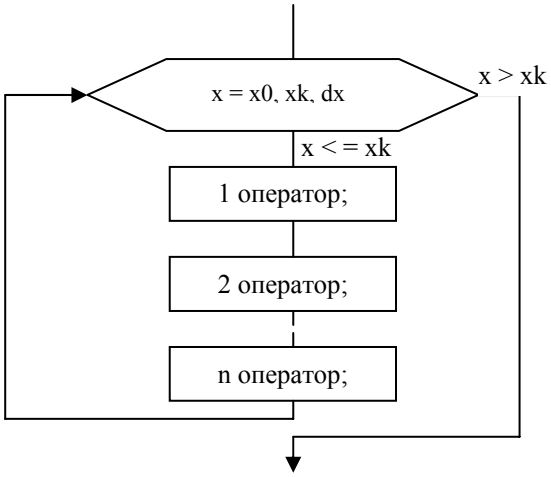
20. Нақты x, y, z сандары берілген. max($x^2 + y^2$, $y^2 + z^2$) – 1 өрнегін есептейтін программа жазыңыз.
21. Бүтін *k, l, m* сандары берілген. Нөлге тең сандардың санын анықтаңыз.
22. Бүтін *k, l, m* сандары берілген. Оң сандардың квадратының қосындысын есептеңіз. Егер бір де бір оң сан жоқ болса, ол жайлы экранға мәлімет шығарыңыз.
23. Бүтін x және y сандары берілген. Егер екі санда жұп болса оларға 1-ді қосыңыз; егер тек біреуі жұп болса, онда олардың көбейтіндісін табыңыз; қалған жағдайда сандарды өзгеріссіз қалдырыңыз.
24. Үш сан берілген. Солардың ішінен [0;1] аралығына кіретіндерін анықтау керек.
25. Нақты x, y, z оң сандары берілген. Қабырғаларының ұзындығы x, y, z-ке тең үшбұрыш тұрғызуға болатынын тексеріңіз. Сандарды енгізгенде олардың теріс емес және нөлге тең емес екендігін де тексеру қажет.
26. Нақты x, y, z сандары берілген. $\min^2$(x +y+z/2, x*y – z) + 1 өрнегін есептеңіз.
27. Бүтін *a*, b, c, d сандары берілген. Нөлге тең емес сандардың көбейтіндісін табыңыз. Егер барлық сандар нөлге тең болса, экранға мәлімет шығарыңыз.
28. Бүтін *a*, b, c сандары берілген. Тақ сандардың қосындысын табыңыз. Егер барлық сандар жұп болса экранға мәлімет шығарыңыз.
29. Бүтін *a*, b, c, d сандары берілген. Егер a>=b>=c>=d болса, онда барлық сандарды нөлге теңестіріңіз; егер a<b<c<d болса, онда әр санды 1-ге өсіріңіз; қалған жағдайда әр санды 1-ге кемітіңіз.
30. Бүтін x, y, z (x≠ y, x≠ z, y≠ z) сандары берілген. Осы сандардың ең кішісін тауып, оның жұп екендігін тексеріңіз.

4. ЦИКЛ ОПЕРАТОРЛАРЫ

4.1. FOR цикл операторы

For операторы айнымалы ретінде берілген цикл параметрінің алғашқы, соңғы мәні мен өзгеру қадамы белгілі болғанда, соған сәйкес бір немесе бірнеше операторларды қайталап орындау кезінде қолданылады. Бұл оператор параметрлі цикл операторы немесе арифметикалық цикл деп аталады.

Параметрлі циклдің орындалу схемасы:



4.1-сурет. For циклінің орындалу алгоритмі

For цикл операторының жалпы жазылу түрі:

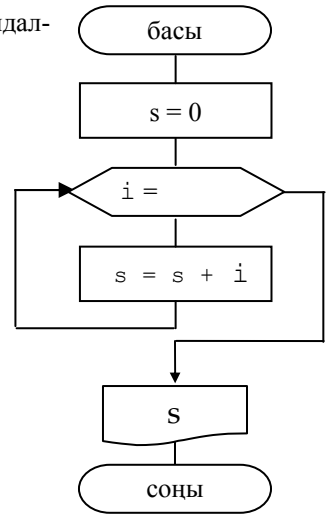
```
for (x=x0; x<=xk; x=x+dx)
     {
      <1-оператор>;
      <2-оператор>;
          . . .
      <n-оператор>;
     }
```

Мұнда **x=x0** – цикл айнымалысының бастапқы мәні, **x<=xk** – циклдің орындалу шарты, **x=x+dx** – цикл айнымалысының қадамы. **x=x0** цикл операторы орындаларда бір рет есептеледі, **x<=xk** ақиқат болса немесе 0-ге тең болмаса, цикл тұлғасы ретіндегі **операторлар** атқарылады. Содан соң **x=x+dx** есептеледі және **x<=xk** мәні қайта анықталады. **x<=xk** мәні жалған болса немесе жалпы жағдайда ол 0-ге тең болса, **for** операторының жұмысы аяқталады. Сонымен цикл

тұлғасының келесі орындалуы немесе орындалмауы оның атқарылуы алдында анықталады.

1-мысал.

```
/* 1-ден 100-ге дейінгі сандар
қосындысын анықтау */
#include <stdio.h>
#include <conio.h>
main ()
{ int s=0,i;
  clrscr();
  printf("1-ден 100-ге
  дейінгі сандар қосындысы:");
  for (i=1;i<=100;i++)
   s+=i;
  printf("s=%d",s);
  printf("\nАяқтау үшін Enter
          басыңыз\n");
  getch();
}
```

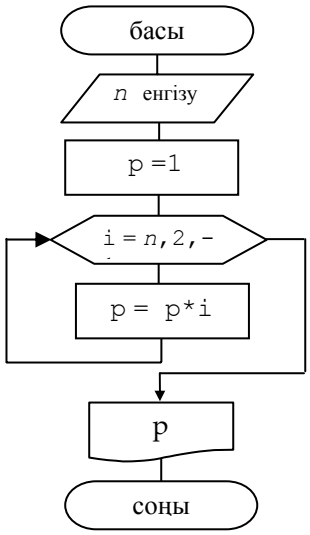


4.2-сурет. Қосынды табу алгоритмі

2-мысал. Бүтін сандардың көбейтіндісін өрнектейтін *n*! мәнін, яғни *n*!=1*2*... *n* табу қажет.

Бұл алгоритмді құру барысында for операторының кері қарай есептейтін мүмкіндігін пайдаланайық (4.3-сурет).

```
#include <stdio.h>
#include <conio.h>
main ()
{ int p=1,i;
  int n;
  clrscr();
  printf("n санын енгізіңіз
        де, Enter басыңыз:");
  scanf("%d",&n);
  printf("1-ден n-ге дейінгі
         сандар көбейтіндісі:");
  for (i=n;i>1;i--)
   p*=i;
  printf(" %d",p);
  getch();
}
```



4.3-сурет. Факториал табу алгоритмі

3-мысал.

```
/* x айнымалысы берілген алғашқы мәннен (x0) соңғы
мәнге (xk) дейін тұрақты қадаммен (dx) өзгеріп
отырғанда, y функциясының мәндерін анықтау */
#include <stdio.h>
#include <conio.h>
main ()
{ float x,y,x0,xk,dx;
clrscr();
printf("x-тің алғашқы,соңғы мәндері : ");
scanf("%f%f",&x0,&xk);
printf("x-тің өзгеру қадамы dx: ");
scanf("%f",&dx);
x=x0;
printf("----------------------\n");
printf("     x    |     y\n");
printf("----------------------\n");

for (x=x0;x<=xk;x+=dx)
{ y=-2.4*x*x+5*x-3;   /* функция */
    printf("%6.2f   |  %6.2f\n",x,y);
}
 printf("----------------------\n");
 printf("\nАяқтау үшін Enter басыңыз");
 getch();
}
```

4-мысал. $y = \sum_{i=1}^{10} \frac{i^2}{2}$ қосындысын анықтау керек.

```
#include <stdio.h>
#define n 10
main ()
{
int i;
float s=0;
for(i=1,i<=n;i++)
    s+=i*i/2;
printf("нәтиже= %f\n",s);
}
```

For цикл операторындағы жақша ішіндегі соңғы өрнек ретінде жалпы дұрыс жазылған кез келген өрнекті пайдалануға болады. Мысалы:

```
for (d=0.1; d<50; d*=5)
   printf("%f",d);
```

For цикл операторындағы жақша ішіндегі бір немесе бірнеше өрнектерді жазбауға да болады, бірақ мұндайда **;** символын міндетті түрде өз орындарына жазып отыру керек, мысалы:

```
x=2; for(n=4; x<=100;)
        x=x*n;
```

For цикл операторында құрама өрнектерді « , » операциясы арқылы жазуға да болады.

« , » операциясы – құрама өрнекті ұйымдастыру үшін қолданылады. Осы операцияны қолданғанда, үтір арқылы бөлектенген өрнектер сол жақтан оң жаққа қарай есептеледі. « , » операциясы цикл операторының тиімді болуы үшін жиі пайдаланылады. Мысалы:

```
main ()
{
  int x,y;
  for (x=1,y=9;x<=10; x++,y--)
  printf("%d%d\n", x,y);
}
```

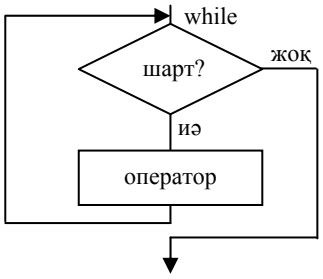
Мұнда алғашқы ; белгісіне дейін және соңғы өрнек арқылы осы цикл операторында екі параметр мәні беріліп (**x=1,y=9;**), олар **x,y** айнымалыларын өзгерту үшін қолданылып отыр.

4.2. While операторы

Орындалу саны алдын ала белгісіз болатын циклдер құру кезінде шарттары алдын ала немесе соңынан тексерілетін екі цикл түрі бар. Шарты алдын ала тексерілетін цикл операторының орындалу схемасы

Оның жазылуы:

```
while (шарт-өрнек)
      оператор;
```



4.4-сурет. Шарты алдын ала тексерілетін цикл

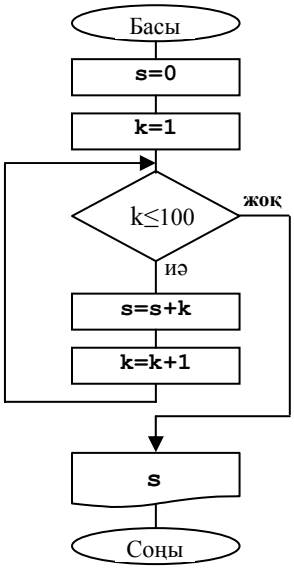
Мұнда шарт ретінде шартты өрнек немесе кез келген типтегі өрнек пайдаланылуы мүмкін. Опера тор қарапайым немесе құрама болуы мүмкін. Ол құрама оператор болса, онда операторлар жиыны жүйелі жақшаға алынып жазылады. **While** операторы орындалғанда, алдымен жақша ішіндегі өрнек есептеліп тексеріледі. Егер өрнек мәні ақиқат болса немесе жалпы жағдайда 0-ге тең болмаса, онда **оператор** атқарылады. Содан соң жақшадағы өрнек тағы

да есептеледі. Егер өрнек мәні жалған болса (немесе жалпы жағдайда 0-ге тең болса), онда **while** цикл операторы өз жұмысын аяқтайды.

Мұнда шарт-өрнек құрамына кіретін айнымалы цикл ішінде өзгеріп отырады.

5-мысал.

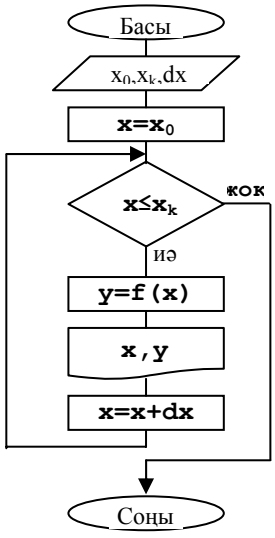
```
/* 1-ден 100-ге дейінгі бүтін
сандар қосындысы */
#include <stdio.h>
#include <conio.h>
main ()
{
int s,k;
clrscr();
s=0; k=1;
while (k<=100)
 { s+=k;
   k++;
 }
printf("s= %d",s);
printf("\nАяқтау үшін
        Enter басыңыз\n");
getch();
}
```



4.5-сурет. Бүтін сандарды қосу алгоритмі

6-мысал. $y=-2.4x^2+5x-3\sqrt{|x|}$

функциясы мәндерін оның аргументі x_0-ден x_k-ға дейін қадамы dx болып өзгерген кездерде анықтау керек. Мұнда цикл алдында параметрге алғашқы мән меншіктеледі де, параметр цикл ішінде берілген қадамға өзгеріп отырады (4.6-сурет). Жалпы функция кез келген түрде беріле алады. Ол параметр мәніне байланысты тармақталып кететін функция да болуы мүмкін.



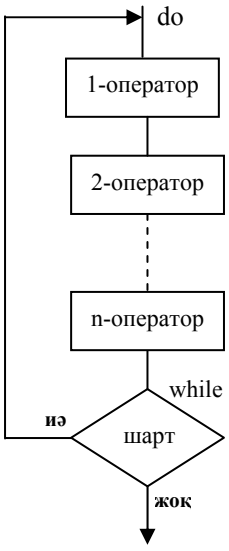
4.6-сурет. Функция мәндерін есептеу алгоритмі

```
/* x тұрақты қадаммен x0-ден xk-ға дейін өзгергенде, функция мәндері
кестесін алу, x0, xk, dx (қадам) пернелерден енгізіледі */
#include <stdio.h>
#include <math.h>
#include <conio.h>
main ()
{
float x,y,x0,xk,dx;
clrscr();/* экранды тазалау */
printf("x-тің бастапқы,соңғы мәндері: ");
scanf("%f%f",&x0,&xk);
printf("x-тің өзгеру қадамы dx-ті енгізіңіз: ");
scanf("%f",&dx);
printf("----------------------\n");
printf("     x    |     y\n");
printf("----------------------\n");
x=x0;
while (x<=xk)
 {
  y=-2.4*x*x+5*x-3*sqrt(fabs(x));
  printf("%6.2f   |
%6.2f\n",x,y);
  x=x+dx;
 }
printf("----------------------
\n");
getch();
}
```

4.3. Do ... while цикл операторы

Шарты соңынан тексерілетін **do** ... **while** циклінің орындалу схемасы суретте көрсетілген. Осыған сәйкес операторлардың жалпы жазылу түрі:

```
do
 {
  1-оператор;
  2-оператор;
    ...  ...  ...
  n-оператор;
 }
while (ернек);
```



4.7-сурет. Шарты соңынан тексерілетін цикл

Цикл тұлғасы ретінде қарапайым немесе құрама оператор қолданылуы мүмкін. Жақшадағы өрнек цикл тұлғасынан кейін тексеріледі. Сондықтан **do while** цикл тұлғасы ең болмағанда бір рет орындалады. Цикл тұлғасынан кейін жазылған өрнек ақиқат болса (немесе жалған жағдайда ол 0-ге тең болмаса), цикл тұлғасы қайтадан орындалады. Ал өрнек жалған болса (немесе 0-ге тең болса), цикл аяқталады. Енді мысалдар келтірейік.

7-мысал.

```
// Енгізілген сандардың үлкенін (максимумын) табу
#include <stdio.h>
#include <conio.h>
main ()
{
int a, max;
clrscr();
printf("\n Сандар максимумын табу \n");
printf("Аяқтау үшін 0 енгізіңіз \n");
max = -32000;
// алдын ала максимумды ең кіші бүтінге теңейміз
do
 {
    printf("Сан енгізіңіз : ");
    scanf("%i",&a);
    if (a > max) max = a;
  }
 while (a!=0);
 printf("Сандардың максимумы: %i",m);
getch();
}
```

8-мысал. Келесі программада шексіз сандар қосындысын

$$s = \sum_{i=1}^{\infty} \frac{1}{i^2} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \ldots + \frac{1}{n^2} + \ldots$$

алдын ала берілгін дәлдікпен $\varepsilon=10^{-5}$ анықтау керек, яғни келесі қосылатын қатар мүшесі осы ε санынан кіші болғанда, қосынды табу аяқталады.

```
#include <stdio.h>
#include <conio.h>
#define epsilon 1e-5
main ()
{
```

```
int i;
float a,s;
clrscr();
s=0; i=1;
do
  {a=1.0/i/i;
   s+=a;
   i++;}
 while (a>epsilon);
 printf("s=%7.4f",s);
 getch();
}
```

9-мысал. Төмендегі программада енгізілген бүтін санның тақ немесе жұп екендігі анықталады.

```
/* Санның жұп екендігін анықтау */
#include <stdio.h>
#include <conio.h>
main ()
{
int k;      /* енгізілетін сан */
char symbol;
textcolor(RED);
textbackground(WHITE);
clrscr();
printf("\n* Санның жұп/тақ екендігін анықтау *\n");
do
 {printf("\nБір бүтін сан енгізіңіз : ");
 scanf("%i",&k);
 printf("Бұл %i саны -",k);
 if (k % 2 == 0)
      printf("жұп сан.");
 else
      printf("тақ сан.");
 printf("\nТағы да енгісесіз бе? Иә-'Y',Жоқ-'N':");
 scanf("%s",&symbol);
  }
 while ((symbol=='Y')||(symbol=='y'));
}
```

Бақылау сұрақтары

1. *Параметрлі циклдің орындалу схемасы мен жазылуы.*
2. *Параметрлі цикл бір де бір рет орындалмауы мүмкін бе?*
3. *for операторының параметрі қандай типтерде бола алады?*
4. *for операторы параметрінің алғашқы мәні оның соңғы мәнінен кіші бола ма?*
5. *Параметрлі цикл операторының неше рет қайталанатынын алдын ала білуге бола ма, болса – ол қалай анықталады?*
6. *for цикл операторындағы жақша ішіндегі бір немесе бірнеше өрнектерді жазбауға бола ма?*
7. *for цикл операторы нүктелі үтірмен аяқтала ала ма?*
8. *for цикл операторында қай кезде құрама операторлар қолданылады?*
9. *for цикл операторы қай кезде шексіз циклге айналады?*
10. *for цикл операторында бірнеше құрама өрнектерді үтір арқылы қалай жазуға болады?*
11. *Шарты алдын ала тексерілетін цикл операторының орындалу схемасы мен оның жазылуы.*
12. *while цикл операторының ішкі тұлғасы бір де бір рет орындалмауы мүмкін бе?*
13. *while цикл операторының шарты қатынас таңбаларысыз жазыла ма?*
14. *while цикл операторы қай кезде шексіз циклге айналады?*
15. *while цикл операторының тұлғасындаоның шартына әсер ететін өрнектер жазыла ма?*
16. *Шарты соңынан тексерілетін do ... while циклінің орындалу схемасы мен жазылуы.*
17. *do ... while цикл операторының ішкі тұлғасы бір де бір рет орындалмауы мүмкін бе?*
18. *do ... while цикл операторының шартында қатынас таңбасы болмаса, оның ақиқат немесе жалған екенін қалай анықтауға болады?*
19. *do ... while цикл операторының ішкі тұлғасында шартсыз көшу операторын қолдануға бола ма?*
20. *do ... while цикл операторы қай кезде шексіз орындалады?*

ТАПСЫРМАЛАР

1. 1-ден *N*-ге дейінгі сандардың қосындысын есептейтін программа құрыңдар. *N*-нің мәні пернетақтадан енгізіледі.
2. 1-ден *N*-ге дейінгі сандардың көбейтіндісін есептейтін программа құрыңдар. *N*-нің мәні пернетақтадан енгізіледі.
3. Пернетақтадан *N* сан енгізіледі. Енгізілген сандардың ішіндегі теріс, оң сандардың және нөлдердің санын анықтайтын программа құрыңдар.
4. Ұзындықтың 1-ден 20 дюймге дейінгі мәндерін сантиметрге (1 дюйм = 2,54 см) айналдыратын және оны экранға шығаратын программа құрыңдар.
5. Банктегі жылдық өсімі 9 пайыздық (проценттік) салымға *S* теңге салынды. *N* жылдан кейін салынған ақша неше теңгеге жетеді?
6. Пернетақтадан 10 бүтін сан енгізіп, солардың квадраты мен кубын анықтаңыз.
7. 20-дан 50-ге дейінгі натурал сандар берілген. Олардың ішіндегі 3-ке бөлінетін, бірақ 5-ке бөлінбейтін сандарды анықтаңдар.
8. 35-тен 87-ге дейінгі натурал сандар берілген. Олардың ішінде 7-ге бөлгенде, 1-ге, 2-ге немесе 5-ке тең қалдық қалатын сандарды табыңдар.
9. 1-ден 50-ге дейінгі натурал сандар берілген. Олардың ішіндегі 5-ке немесе 7-ге бөлінетін сандардың қосындысын табыңдар.
10. Пернетақтадан 10 сан енгізіңдер. Егер олардың ішінде 15-тен асқаны бар болса, онда оларды 15-пен алмастырыңдар. Сандарды экранға шығарыңдар.
11. Пернетақтадан он теріс және оң сан енгізіңдер. Барлық теріс сандарды олардың модульдерімен алмастырып сандарды экранға басып шығарыңдар.
12. Екі орынды сандардың ішіндегі 4-ке бөлінетінін, бірақ 6-ға бөлінбейтінін табыңдар.
13. 13-ке қалдықсыз бөлінетін екі орынды тақ сандардың көбейтіндісін табыңдар.
14. 100-ден 200-ге дейінгі сандардың ішіндегі 17-ге қалдықсыз бөлінетін сандардың қосындысын табыңдар.
15. Пернетақтадан 10 сан енгізіңдер. Егер сан 100-ден кем болса, онда осы санды және оның квадратынтабыңдар.
16. 1-ден бастап өздерің енгізген бүтін n санына дейінгі сандардың квадраттарының қосындысын есептейтін программа құрыңдар.

17. 200-ге дейінгі 5-ке бөлгенде 4 қалдық қалатын сандар нешеу екенін табу керек.

18. 200-ге дейінгі бүтін сандардың 25-ке қалдықсыз бөлінетін сандары нешеу екенін анықтаңдар.

19. 20-дан үлкен және 100-ден кіші 3-ке қалдықсыз бөлінетін оң сандардың қосындысын табу керек.

20. Берілген a нақты және b бүтін сандарының мәндері бойынша a^b өрнегінің мәнін pow() функциясын пайдаланбай табатын программа жазыңдар.

21. Төмендегі есептерде бір өлшем бірлігін екінші өлшем бірлігіне түрлендіру қажет (цикл санауышы мәні 1-ден 20-ға дейін өзгереді):

а) футпен берілген ұзындық өлшемін метрге (1 фут = 0,3048 м);

ә) драхмды граммға (1 драхм = 3,7325 г;

б) унцияны граммға (1 унция = 29,86 г);

в) фунтты килограммға (1 фунт = 0,40951 кг);

г) аршынды метрге (1 аршын = 0,7112 м);

д) қадақты граммға (1 қадақ = 400 г);

е) қарысты сантиметрге (1 қарыс = 18 см);

ж) дюймді миллиметрге (1 дюйм = 25,3995 мм).

22. Төмендегі функциялар мәндерін x айнымалысы x0-ден xk-ға дейін dx қадамымен өзгерен кезде анықтаңдар.

Вар	Функция	Берілгендері
1	$Y=\begin{cases}\sqrt{\lvert a-x\rvert}\cdot\sin^2 x, & \text{егер } a<x;\\ \left(\dfrac{x}{\lvert a+x\rvert}\right)\cdot\sqrt[3]{\lvert\sin x\rvert}, & \text{егер } a=x;\\ e^{\sqrt{\lvert x\rvert}}, & \text{егер } a>x;\end{cases}$	$a=0.265\cdot10^2$ x_0=10 x_k=30 dx=1.5
2	$Y=\begin{cases}ax+0.23x^2\log_2 a, & \text{егер } a<x;\\ \left(\dfrac{xe^a}{\lvert a+x\rvert}\right)\cdot\sqrt{\lvert\cos x\rvert}, & \text{егер } a=x;\\ x\cdot tga, & \text{егер } a>x;\end{cases}$	a=1.5 x_0=-2 x_k=5 dx=0.5
Вар	Функция	Берілгендері

3	$Y=\begin{cases}\left(a^2+x^2\right)\cdot e^x, & \text{егер } a<x;\\ \sqrt{\lvert a\rvert}\cdot\sin^4 x, & \text{егер } a=x;\\ e^{\frac{\lvert a-x\rvert\cdot\cos x^4}{a}}, & \text{егер } a>x;\end{cases}$	a=2.5 x_0=0 x_k=3 dx=0.25
4	$Y=\begin{cases}\ln\lvert a+x\rvert\cdot\cos\lvert x^3\rvert, & \text{егер } a<x;\\ e^{1.2}-\sqrt{\lvert a+x\rvert}, & \text{егер } a=x;\\ \frac{\sqrt[3]{\lvert a+x\rvert}}{(a-x)}, & \text{егер } a>x;\end{cases}$	a=2.5 x_0=0 x_k=3 dx=0.25
5	$Y=\begin{cases}a^2+\sqrt{\lvert a^2+x\cdot\sin x\rvert}, & \text{егер } a<x;\\ 2x^2+a^3\cdot tgx, & \text{егер } a=x;\\ \frac{x^2}{\sqrt{\lvert a\rvert}}, & \text{егер } a>x;\end{cases}$	$a=0.637$ x_0=-3 x_k=3 dx=0.5
6	$Y=\begin{cases}e^{0.2}+\sqrt{a+x}, & \text{егер } a<x;\\ (a+x)\cdot\sqrt[3]{\lvert\sin x\rvert}, & \text{егер } a=x;\\ \sqrt{\lvert a+x\rvert}, & \text{егер } a>x;\end{cases}$	$a=0.234\cdot10^2$ x_0=3 x_k=6 dx=0.5
7	$Y=\begin{cases}2.75e^{\lvert x+a\rvert}+\cos^4 x, & \text{егер } a<x;\\ \frac{(x+a)\cdot tgx}{\lg\lvert x\rvert}, & \text{егер } a=x;\\ e^{ax}+\frac{a\cdot\sin^2 x}{\sqrt[3]{\cos^2 x}}, & \text{егер } a>x;\end{cases}$	$a=0.567\cdot10^1$ x_0= -3 x_k= 3 dx= 0.5

23. Төмендегі жалпы мүшесі берілген қатардың 10 мүшесінің қосындысын табыңдар:

а)	$a_n = e^{-\sqrt{n}}$	ә)	$a_n = n^3 e^{-n}$	б)	$a_n = \frac{2^n n!}{n^n}$
в)	$a_n = \frac{3^n n!}{(3n)!}$	г)	$a_n = \frac{n!}{3n^n}$	д)	$a_n = \frac{(n!)^2}{2^{n^2}}$
е)	$a_n = \lg(n!)e^{-n\sqrt{n}}$	ж)	$a_n = 10^{-n}(n-1)!$	з)	$a_n = \frac{n^3}{(3n-3)!}$

24. Жалпы мүшесі төменгі өрнекке сәйкес қатар қосындысын $\varepsilon=10^{-4}$ дәлдігімен анықтау керек.

а)	$a_n = \frac{(-1)^{n-1}}{n^n}$	ә)	$a_n = \frac{1}{2^n} + \frac{1}{3^n}$	б)	$a_n = \frac{1}{((3n-2)(3n+1))}$
в)	$a_n = \frac{(2n-1)}{2^n}$	г)	$a_n = \frac{10^n}{n!}$	д)	$a_n = \frac{n!}{(2n)!}$
е)	$a_n = \frac{n!}{n^n}$	ж)	$a_n = \frac{n}{(n-1)^2}$	з)	$a_n = e^n \cdot 100^{-n^2}$
и)	$a_n = \frac{n^{\ln n}}{(\ln n)^n}$	к)	$a_n = \frac{n!}{n^{\sqrt{n}}}$	л)	$a_n = n^2 e^{-\sqrt{n}}$

5. СИ ТІЛІНДЕ ЖИЫМДАРДЫ ПАЙДАЛАНУ

Жиым немесе массив – бір типтегі элементтердің реттелген жиыны. Олар бір атаумен – идентификатормен аталады да, индексті айнымалы ұғымына сәйкес келеді. Мысалы, мынадай тізбек

0 1 1 2 3 5 8 13 21

Фибоначчи тізбегінің 9 элементін құрайды (алғашқы екі санды таңдап алып, келесі санды алдыңғы екеуін қосу жолымен алады). Ал мынау өзіне және бірге ғана бөлінетін жай сандар тізбегінің алғашқы 7 элементі:

1 3 5 7 11 13 17

Осындай бір текті тізбектерді жиым түрінде Си тілінде сипаттап, оған бастапқы мән беріп инициалдау үшін былай жазамыз:

```
int fib[8]={0, 1, 1, 2, 3, 5, 8, 13, 21};
```
немесе
```
int fib[]={0, 1, 1, 2, 3, 5, 8, 13, 21};
```
деп көрсетеміз. мұндағы **fib** – жиым аты, оның элементтерінің типі **int**, ал ені, яғни ұзындығы – 9, жиым элементтерінің индекстері 0-ден бастап нөмірленеді, сол себепті 9 элемент 8 индекспен көрсетіледі. Мәндері толық көрсетілсе, индексті жазбаса да болады. Ал былай болса,

```
int fib[8]={0, 1, 2, 3};
```
қалған элементтері 0 болып саналады.

```
n=10; k=2; fib[n-k]={0, 1, 2, 3};
```
десе де болады.

Жоғарыдағы тізбектің 7-ші элементін бір бүтін айнымалыға меншіктеу үшін былай жазамыз.

```
int a = fib[6];  // a = 8
```

Жиымды сипаттау кезінде оның ені нақты санмен көрсетіледі, мыс., **a[20], a[n]** деп жазу үшін алдын ала

#define n 20 жолы көрсетіледі немесе

const n=20; болып жазылады.

Жиым элементтерін енгізу немесе оларды түрлендіру үшін цикл операторлары қолданылады. Төменде 10 элементі бар жиымды 0-ден 9-ға дейінгі сандармен толтырып, сонан кейін оларды кері бағытта экранға шығару мысалы көрсетілген:

```
main ()
{
 int ia[10];
 int index;
 for (index = 0; index <10; index ++)
    ia[index] = index ;
 for (index = 9; index >=0; index --)
    printf(" %i", ia[index]);
```

Си тілінде жиымды жиымға бірден теңестіруге болмайды, мыс., a_0, a_1, a_2, ... , a_9 және c_0, c_1, c_2, ... , c_9 жиымдары үшін $a = c$ деп жазуға рұқсат

етілмейді. Олардың элементтерін цикл ішінде бір-біріне біртіндеп теңестіру керек.

Мысалы, мынадай цикл жазылуы тиіс:

```
int a[9], c[9];
for (int i=0; i<9; ++i)
    a[i]=c[i];
```

Си тілінде кездейсоқ сандарды пайдалану

Си тілінде кездейсоқ сандар беретін функциялар бар.

int rand() – 0..RAND_MAX=32767 аралығынан кез келген кездейсоқ бүтін сан береді.

Ал **int random(n)** 0..n аралығынан кез келген кездейсоқ бүтін сан береді.

Бұл функцияларды пайдалану **<stdlib.h>** файлы арқылы орындалады.

Мысалы:

```
//a[n] жиымына кездейсоқ сандар енгізу
#include<conio.h>
#include<stdio.h>
#include<stdlib.h>
void main()
{
int a[100];
int n;
printf("\nEnter the size of array:", n);
scanf("%i",&n);
for(int I=0;I<n;I++)
{a[I]=rand()%100-50;
 printf("  %i  ", a[I]);}
 getch();
}
```

Жиымды өңдеу есептерінің түрлері (кластары)

Жиымды өңдеу есептері көбінесе бірыңғайланған төрт түрге бөлінеді.

1) Есептердің 1-түріне жиым элементтерінің барлығын немесе көрсетілгендерін бірдей бір тәсілмен өңдеу есептері жатады.
2) Есептердің 2-түріне (класына) жиым элементтерінің орналасу реттілігін өзгерту тәсілдері жатады.
3) Есептердің 3-класына бірнеше жиымдарды қатар өңдеу немесе бір жиымның ішкі элементтерін бірнеше топқа бөліп жеке-жеке өңдеу тәсілдері жатады. Жиымдар бір тәсілмен – синхронды өңделеді немесе әр түрлі тәсілмен – асинхронды түрде өңделеді.

4) Есептердің 4-класына жиымның берілген санға тең бірінші элементін табу, яғни іздеу есептері жатады.

1-түрдегі есептер

1 есеп. *Жиымның ең үлкен элементін анықтау керек.*

```
// максимум табу
#include<conio.h>
#include<stdio.h>
#include<stdlib.h>
void main()
{
   int a[100];
   int n;
   printf("\nEnter the size of array:", n);
   scanf("%i",&n);
   for(int I=0;I<n;I++)
   {  a[I]=rand()%100-50;
      printf("  %i  ", a[I]);
   }
   int max=a[0];
   for(I=1;I<n;I++)
   if (a[I]>max) max=a[I];
   printf("\nMax= %i", max);
   getch();
}
```

2 есеп. *Жиымның жұп индексті элементтері қосындысын анықтау.*

```
/* 0, 2, 4... индексті элементтер қосындысын табу  */
#include<conio.h>
#include<stdio.h>
#include<stdlib.h>
void main()
{
int a[100];
int n;
printf("\nEnter the size of array:", n);
scanf("%i",&n);
for(int I=0;I<n;I++)
{a[I]=rand()%100-50;
 printf(" %i ", a[I]);
}
```

```
  int Sum=0;
  for(I=0;I<n;I+=2)
  Sum+=a[I];
  printf("\nSum= %i ", Sum);
  getch();
}
```

Соңғы циклді басқаша да құрастыруға болады:

```
  //Екінші тәсіл
  for(I=0;I<n;I++)
  if(I%2==0) Sum+=a[I];
  printf("\nSum= %i ", Sum);
```

2-түрдегі есептер

Жиым ішіндегі екі элементтің бір-бірімен орнын ауыстыру үшін қосымша тағы бір айнымалы керек болады. Мысалы, a[I] және a[J] элементтерінің орнын ауыстыру үшін қосымша R айнымалысы керек:

```
R=a[I]; a[I]=a[J]; a[J]:=R;
```

3-есеп. Жиым элементтерін кері бағытта орналастыру.

```
for(int i=0, j=n-1; i<j; i++, j--)
{int r=a[i];
 a[i]=a[j];
 a[j]=r;}
```

4-есеп. Жиымның қатар тұрған екі элементін: 1 және 2, 3 және 4, 5 және 6, т.с.с. элементтерін бір-бірімен орын ауыстыру

```
for(int i=0;i<n-1;i+=2)
{int r=a[i];
 a[i]=a[i+1];
 a[i+1]=r;}
```

5 есеп. Жиым элементтерін **k** орынға солға (оңға) ығыстыру, яғни жылжыту.

```
int k,i,t,r;
printf("k = ");
skanf("%d",&k);
for(t=0;t<k;t++)
{
      r=a[0];
      for(int i=0; i<n-1; i++)
            a[i]=a[i+1];
      a[n-1]=r;
}
```

3-класс есептері

Жиымдарды синхронды түрде өңдеуде жиымдар элементін қарастыру кезінде индекстер бірдей қадамға өзгереді. Мысалы, бүтін сандардан құралған n элементтерден тұратын 2 жиым берілген делік. Жаңа **c** жиымы мынадай формула арқылы алынады: c[I]=a[I]+b[I].

```
for (int I=0; I<n; I++)   c[I]=a[I]+b[I];
```

Жиымдарды асинхрондық өңдеу кезінде әр жиым индексі өз реттілігімен өзгеріп отырады.

6-есеп. Бүтін сандардан құралған жиымдағы теріс элементтердің барлығын оның бас жағына орналастыру керек.

```
int b[10];    //қосымша массив
int i,j=0;
for(i=0;i<n;i++)
   if(a[i]<0){b[j]=a[i];j++;}
   // a-дан b-ға теріс элементтерді көшіріп жазу
for(i=0;i<n;i++)
   if(a[i]>=0){b[j]=a[i];j++;}
   // a-дан b-ға оң элементтерді көшіріп жазу
for(i=0;i<n;i++) printf (" %d ", b[I]);
```

7 есеп. Жиымның барлық жұп элементтерін жою керек.

```
int b[10];
int i,j=0;
for(i=0;i<n;i++)
  if(a[i]%2!=0){b[j]=a[i];j++;}
for(i=0;i<j;i++)  printf (" %d ", b[I]);
printf ("\n");
```

4-класс есептері

Іздеу есептерінде берілген шартқа сәйкес келетін элементті іздеп табу керек. Ол үшін жиым элементтерін біртіндеп тізбектей қарастырып отырып шартты тексеріп шығу қажет. Осылай ету барысында циклден шығудың екі жолы бар:

- керекті элемент табылғаннан кейін;
- жиым элементтері тегіс қаралып шықты, керекті элемент табылмады.

8- есеп. Берілген k санына тең жиымның алғашқы элементін табу.

```
int k;
printf("\nK=");
scanf("%i",&k);
int ok=0;//элемент табылғаны/табылмағаны белгісі
```

```
int i,nom;
for(i=0;i<n;i++)
   if(a[i]==k){ok=1;nom=i;break;}
if(ok==1) printf("\nnom=",nom);
  else  printf("\nk-ға тең элемент жоқ!");
```

Жиымды сұрыптау (сорттау, реттеу)

Сұрыптау – берілген объектілер жиынын (сандарды) ұсынылған реттілікпен қайта теріп орналастыру процесі.

Жиымдарды сұрыптау жылдамдығы әр түрлі болады. Қарапайым сұрыптау тәсілдері n*n рет салыстыруды керек етеді, мұндағы n – жиым элементтері саны; ал жылдам сұрыптау тәсілі n*ln(n) рет салыстыруды қажет етеді. Қарапайым тәсілдер түсінуге жеңіл, өйткені алгоритмі түсінікті. Күрделі тәсілдер аз әрекеттер санын керек еткенмен, операциялары күрделірек болады, сондықтан элементтер саны аз жиымдарға қарапайым тәсілдерді қолданған дұрыс.

Қарапайым тәсілдер 3 топқа бөлінеді:

- жай таңдау жолымен сұрыптау;
- жай енгізу тәсілімен сұрыптау;
- жай алмастыру тәсілімен сұрыптау.

Жай таңдау жолымен сұрыптау

Жиымның ең кіші элементі анықталады да, ол бірінші элементпен орын ауыстырады. Қалған элементтермен де осы тәсіл қайталанады.

44	55	12	42	94	18
		минимум			

```
int i,min,n_min,j;
for(i=0;i<n-1;i++)
{
  min=a[i];n_min=i;  // минимумды іздеу
  for(j=i+1;j<n;j++)
  if(a[j]<min)
    { min=a[j];n_min=j; }
  a[n_min]=a[i]; //алмастыру
  a[i]=min;}
```

Жай енгізу (кірістіру) тәсілімен сұрыптау

Жиым элементтері екіге – бастапқы тізбекке және дайын тізбекке бөлінеді. Әрбір адымда I=2 нөмірінен бастап, бастапқы берілген тізбектен I-ші элемент алынады да, ол дайын тізбектің керекті жеріне орналастырылады. Мұнан кейін I-ге 1 қосылады да, сол әрекеттер қайталанады.

44	55	12	42	94	18

дайын тізбек бастапқы тізбек

Керекті орынды іздеу кезінде оң жақтағы келесі элементпен орын ауыстыру қарастырылады, яғни таңдалып алынған элемент сұрыпталғандардың J:=I-1 нөмірінен басталатын кезекті элементімен салыстырылады. Егер таңдалып алынған элемент a[I]-ден артық болса, онда ол сұрыпталғандар ішіне қосылады, әйтпесе a[J] бір орынға ығысады да, таңдалған элемент сұрыпталғандар ішіндегі келесі элементпен салыстырылады. Керекті орынды іздеу әрекеті екі жағдайда:

- егер a[J]>a[I] болатын элемент табылса;
- дайын тізбектің сол жақ шетіне жеткен кезде аяқталады.

Мысалы:

```
int i,j,x;
for(i=1;i<n;i++)
 { x=a[i];//ауысатын элементті есте сақтау
   j=i-1;
   while(x<a[j]&&j>=0) //керекті орынды іздеу
    {
      a[j+1]=a[j]$      //оңға жылжыту
      j--;
    }
   a[j+1]=x;//элементті кірістіріп қою
 }
```

Жай алмастыру арқылы сұрыптау

Мұнда ең соңғыдан бастап, екі элемент салыстырылады да, қажет болса, орындары алмастырылады. Осындай әрекет нәтижесінде ең кіші элемент жиымның ең сол жақ шетіне ығысады. Қалған жиым элементтері үшін де осы процесс қайталанады.

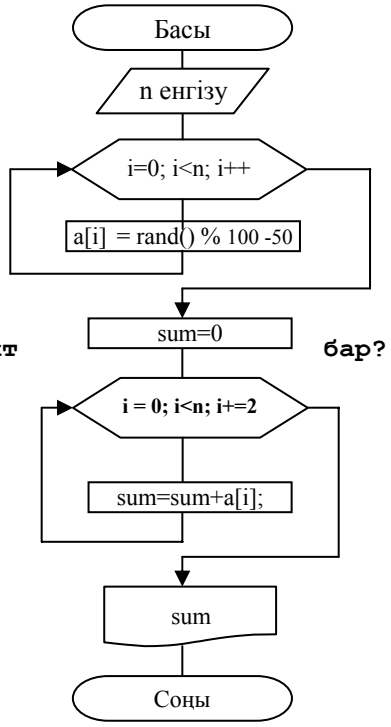
44	55	12	42	94	18

```
for(int i=1;i<n;i++)
for(int j=n-1;j>=i;j--)
   if(a[j]<a[j-1])
```

```
{int r=a[j];a[j]=a[j-1];a[j-1]=r;}
```

9-есеп. Бүтін оң және теріс сандардан тұратын a[n] жиымының жұп нөмірлі элементтерінің қосындысын табу керек, мұнда n саны енгізіледі, ал жиым элементтерінің мәндері кездейсоқ бүтін сандардан тұрады.

rand() функциясы 0...32767 аралығындағы бүтін сан береді. Оны пайдалану үшін **stdlib.h** директивасын, яғни тақырып файлын қолдану қажет. Жиым элементтері екі разрядты оң және теріс сандардан тұруы үшін алынған кездейсоқ сан 100-ге бөлініп, қалдығынан 50 алып тасталынған.



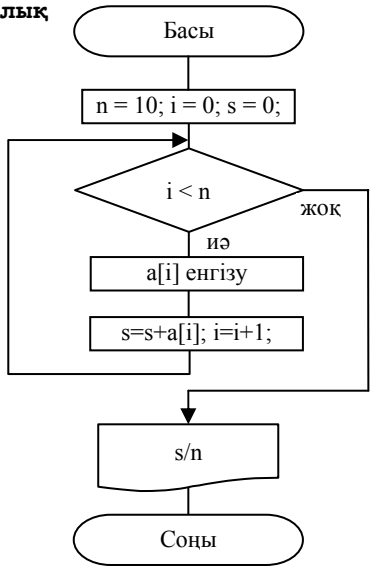
5.1-сурет. Жиымның жұп элементтерін қосу

```
/*жиымның  жұп элементтері
қосындысы*/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

main()
{
int a[50];
int n;
printf("\nЖиымда неше элемент бар?
");
scanf("%d",&n);
printf("Жиым
элементтері:\n");
for(int i=0;i<n;i++)
  {
   a[i]=rand()%100-50;
   /* жиымға 0 - 50 аралығындағы
кездейсоқ сандарды меншіктеу */
   printf("%d, ",a[i]);
  // сандарды экранда бейнелеу
  }
int sum=0;
for(i=0;i<n;i+=2)
sum+=a[i];
// 0, 2, 4... индексті элементтерді қосу
printf("\nЖиымның жұп элементтері қосындысы:
%d",sum);
getch(); // нәтижелік экранды жапқызбай, көруге мүмкіндік беру
}
```

10-есеп. Бүтін сандардан құралған A(10) бір өлшемді жиымы берілген. Сол жиым элементтерінің арифметикалық ортасын табу керек.

```
/* A[10] жиымының арифметикалық
  ортасын табу */
#include <conio.h>
#include <stdio.h>
#define n 10
main ()
{int i=0,s=0;
 int a[n];
 textcolor(RED);
// экран символдары қызыл түсті
 textbackground(GREEN);
// экран фоны жасыл түсті
 clrscr();
 printf ("Жиым элементтерін
    - 10 сан енгізіңіз:\n");
 while (i<n)
{
 printf("a[%i]=",i);
 scanf("%i",&a[i]);
 s=s+a[i];
 i=i+1;
}
printf("Жиым арифметикалық ортасы : %5.2f",(float)s/n);
printf("\nАяқтау үшін Enter басыңыз");
getch();
}
```



5.2-сурет. Жиымның арифметикалық ортасын анықтау

11-есеп. Бүтін сандардан тұратын A_{10} жиымы берілген. Сол жиымның ең үлкен элементін – максимумын және оның индексін анықтау керек.

```
/* Жиым максимумын табу */
#include <conio.h>
#include <stdio.h>
#define n 10
main()
{ int i,t,a[n]={6,5,9,8,7,4,1,2,3,0},max;
textcolor(BLUE);
textbackground(YELLOW);
clrscr();
printf("a[10] элементтері : ");
for (i=0; i<n; i++)
```

```
 printf(" %d ",a[i]);
max=a[0]; t=0;   // max - максимум, t - оның индексі
for (i=1; i<n; i++)
 if (a[i] > max)
  {max = a[i]; t=i;}
printf("\nmax = %d, индексі = %d\n", max, t);
getch();
}
```

12-есеп. Нақты сандардан тұратын A[15] жиымы берілген. Жиымның оң элементтерінің геометриялық ортасын анықтау керек.

Геометриялық орта мынадай өрнекпен анықталады:

$$g = \sqrt[n]{a_1 \cdot a_2 \cdot \ldots \cdot a_n}$$

```
/* Нақты сандардан тұратын A[15] жиымының оң
элементтерінің геометриялық ортасын табу керек */
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#define n 15

main ()
{
 int i;
 float a[n];
 randomize(); // кездейсоқ сандарды өзгертіп отыру функциясы
 textcolor(MAGENTA);
 textbackground(WHITE);
 clrscr();
 printf("\nЖиым элементтері:\n");
 for(i=0;i<n;i++)
  {a[i] = (float)(rand() % 100 - 50)/10;
   printf(" %5.1f",a[i]);}
   // көбейтінді мен оң элементтердің санын табу
 i=0;          // бастапқы индекс = 0
 float p=1;    // оң элементтер көбейтіндісі
 int k=0;      // оң элементтер саны
 do
 { if (a[i]>0)  { k++; p*=a[i];}
   i++;
 }
 while (i<n);
```

```
 printf("\nоң элементтер саны : %d\n",k);
 p=pow(p,1.0/k);
 printf("геометриялық орта = %f",p);
 getch();
}
```

Адрестік операциялар

Адрестік операциялар үшін Си тілінде екі арнайы оператор қолданылады.

& – адресті анықтау үшін қолданылатын операция;

* – адрес арқылы қатынас жасау үшін қолданылатын операция.

& операциясы берілген айнымалының адресін қайтарады. Мысалы, программа мәтінінде sum айнымалысы былай сипатталған болсын:

```
int sum;
```

онда **&sum** жазуы осы айнымалының компьютер жадындағы орналасқан ұясының адресі болып табылады.

Нұсқауыштар. Нұсқауыш мәліметтердің адресін сақтайтын айнымалы болып табылады. Жалпы алғанда, нұсқауыш адрестің символдық кескінделуі болып саналады. Қарастырылатын мысалда **&sum** – **sum** атты айнымалыға сілтейтін нұсқауыш болып табылады. Нақты адрес ретінде белгілі бір сан тұрады, ал **&sum** нұсқауыш типті константа болып табылады.

Си тілінде нұсқауыш типті айнымалылар да бар. Нұсқауыш типті айнымалылар мәні болып белгілі бір шаманың адресі саналады.

Мысалы, нұсқауыш типті айнымалы **ptr** идентификаторы арқылы белгіленген болсын, онда төмендегі оператор **sum** айнымалысының адресін **ptr** атты нұсқауыш типті айнымалыға меншіктейді. **ptr** атты нұсқауыш типті айнымалы басқа да объектіге сілтеуі мүмкін. Мысалы:

```
int *ptr;
ptr=&sum; ptr=&max;
```

* операциясы – адрес арқылы қатынас жасау үшін пайдаланылатын операция. Мысалы, **ptr** нұсқауыш типті айнымалысында **max** айнымалысына нұсқайтын сілтеме сақталған болсын. Осы айнымалының мәнін білу үшін * адресі бойынша қатынас жасау операциясын қолдануға болады. **ptr** нұсқауышы мәнін анықтау үшін келесі операцияны орындау қажет: **res=*ptr;**

Нұсқауышты сипаттау. Нұсқауыш типті айнымалыны сипаттағанда берілген нұсқауыш қандай типті айнымалыға сілтейтінін көрсету қажет. Өйткені әр түрлі типті айнымалыға ЭЕМ жадында ұялардың әр түрлі саны бөлініп беріледі.

```
int   *iptr;
```

```
char  *cptr;
float *fptr;
```

Функциялар арасында байланыс жасау үшін нұсқауыштарды пайдалану

13 мысал. Бұл программада айнымалының мәндерін ауыстыру үшін нұсқауыштар пайдаланылған:

```
#include <stdio.h>
#include <conio.h>
void change (int *u, int *v)
  {
   int temp;
   temp = *u; *u = *v; *v = temp; }
main ()
{ int x=5, y=10;
  clrscr();
  printf ("x=%d y=%d\n",x,y);
  change (&x,&y);
  printf ("x=%d y=%d\n",x,y);
  getch();
}
```

Нәтижесі:

```
x=5 y=10
x=10 y=5
```

Жиымдар және жиымдарға қолданылатын нұсқауыштар

Жиымдарды сипаттағанда, элементтер типі және жалпы жағдайда компьютер жадының қажетті класы көрсетіледі. Қарапайым айнымалыларда қарастырылатын қасиеттер жиымдарды де сипаттау кезінде қолданылуы мүмкін. Мысалы:

```
int b[30];
main()
{ float a[30];
  static char c[20];
  extern b[];
  ......
}
```

Жиымдарды инициалдауды қарастырайық. Жиым сипатталуында тек сыртқы немесе статистикалық жиымдар ғана инициалдануы мүмкін. Мысалы, 8 топтағы студенттер саны **stud**[8] жиымы ретінде көрсетілген:

```
#include <conio.h>
#include <stdio.h>
int stud[8]={15,16,14,18,15,20,17,19};
```

```
main()
{ int i;
   clrscr();
   extern int stud[];
   for (i=0; i<8; i++)
   printf("N %i тобында %i студент\n",
          i+1,stud[i]);
}
```

Жиымның нұсқауыштарын қарастырайық. Мысалы:

```
int *a;
```

Жиым аты нұсқауышты қолданған жағдайда, жиымның 0-ші элементін анықтайды, яғни **a** жиымы сипатталған болса, программа мәтініндегі **a** идентификаторы 0-ші элементті көрсетеді деп саналады:

```
a == &a[0];
```

бұл теңдеудің екі бөлігі де – **a** да және **&a[0]** де жиымның 0-ші элементінің адресін анықтайды. Осы екі белгілеу де нұсқауыш типті константа болып табылады. Сондықтан оларды мән ретінде нұсқауыш типті айнымалыға меншіктеуге болады немесе қажет болса, нұсқауыш типті айнымалының мәнін өзгертуге болады.

14 мысал. Нұсқауыштың мәніне санды қосуға болатынын көрсететін программа қарастырайық.

```
main()
{ int a[4], *pti, i;
  float b[4], *ptf;
  pti=a;
  ptf=b;
  for (i=0; i<4; i++)
  printf("нұсқауыштар +% d: %8u  %10u\n",i,pti+i,
         ptf+i);
}
```

Мұның нәтижесі:

```
нұсқауыштар + 0:   65518       65498
нұсқауыштар + 1:   65520       65502
нұсқауыштар + 2:   65522       65506
нұсқауыштар + 3:   65524       65510
```

Келесі мысалды қарастырайық.

```
(a+2)==&a[2];
*(a+2)==a[2];
```

Бұлар нұсқауыштар мен жиымның арасындағы байланысты анықтайды, яғни жиымның жеке элементін анықтау үшін немесе оның мәнін пайдалану үшін нұсқауышты қолдануға болады.

Нұсқауыштарды пайдаланып жиымдармен жұмыс істеу

Жиымдарды функция арқылы қарастырып, содан кейін осы функцияны нұсқауыштарды пайдаланып жазып шығу керек болсын.

15 мысал. Функция арқылы жиым қосындысын табу.

```
// функция арқылы жиым қосындысын табу
#include <conio.h>
#include <stdio.h>
int f1(int a[], int t)
{
 int i,sum=0;
 for (i=0; i<t; i++)
    sum+=a[i];
 return (sum);
}
main()
{ int i,s,b[10]={5,6,14,12,30,5,9,7,15,5};
  clrscr();
  s=f1(b,10);
  printf("s=%d",s);
  getch();
}
```

Негізгі программада **f1** функциясын шақыру үшін нақты параметрлерді жазып, функция келесі түрде шақырылып отыр:

```
f1(b,10);
```

16 мысал. Нұсқауышты функцияда (**f1**) пайдалану программасын жазайық.

```
// функцияда нұсқауыш арқылы жиым қосындысын табу
#include <conio.h>
#include <stdio.h>
int f1(int *pa, int t)
{
 int i,sum=0;
 for (i=0; i<t; i++)
    sum+=*(pa+i);
 return (sum);
}
main()
{ int i,s,b[10]={5,6,14,12,30,5,9,7,15,5};
clrscr();
s=f1(b,10);
printf("s=%d",s);
getch();}
```

программада осы **f1** функциясын шақыру үшін, нақты параметрлер бұрынғыдай жазыла береді: **f1(b,10);**

Нұсқауыштарға қолданылатын операциялар

Си тілінде нұсқауыш типті айнымалыларға бес негізгі оператор қолдануға болады:

1. меншіктеу операциясы. Нұсқауышқа адресті меншіктеуге болады. Жиымның атын қолданып немесе адресті анықтайтын & операторын пайдаланып, әдетте адресті меншіктеуге болады;
2. мәнді анықтау. Берілген адрес бойынша кейбір ұяшықта сақталатын мәнді анықтау үшін * операциясы қоланылады;
3. нұсқауыштың адресін анықтау. Кез келген айнымалылар сияқты нұсқауыш типті айнымалылар мәні немесе адресі болуы мүмкін. & операциясы арқылы нұсқауыштың адресін анықтауға болады;
4. нұсқауыштарды арттыру. Бұл амал әдеттегі + операциясы көмегімен немесе арттыру операциясы арқылы орындалуы мүмкін. Нұсқауышты арттырып, жиымның келесі элементіне өтуге болады (қажет болса, нұсқауыш мәнін кемітуге де болады);
5. нұсқауыштардың айырмасы. Бір жиымның элементіне сілтейтін нұсқауыштың айырмасын табуға болады. Жиым элементінің арасындағы ара қашықтығын анықтау үшін нұсқауыштың айырмасын есептеуге болады.

17 мысал.

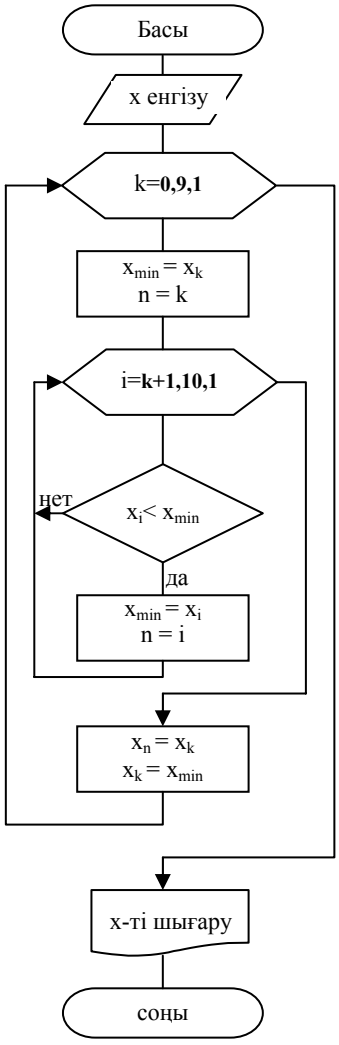
```
/* Жиымның максимумын тауып, одан кейінгі
элементтерін кемуі бойынша реттеп орналастыру */
#include <conio.h>
#include <stdio.h>
#include <math.h>
#define n 10
main()
{ int i,j,t,c;
  int a[n]={6,5,9,8,7,4,1,2,3,0};
  int *pa,max;
clrscr();

printf("a[10] элементтері : ");
for (i=0; i<n; i++)
 printf(" %d ",a[i]);

pa=a; max=*pa; t=0;   // максимумды және оның
                      // индексін табу
for (i=1; i<n; i++)
 if((*(pa+i)) > (max))
```

```
  {max = *(pa+i); t=i;
  }
 printf("\nmax = %d оның индексі = %d\n", max, t);

for (i=t; i<n-1; i++) // элементтерді кемуі бойынша
for (j=i+1; j<n; j++) // реттеу
if(*(pa+i) < *(pa+j))
 { c=*(pa+i); // жиымның 2
элементін алмастыру
   *(pa+i)=*(pa+j);
   *(pa+j)=c;
 }
 printf("\nнәтиже : ");
for (i=0; i<n; i++)
    printf("%d  ", *(pa+i));
getch();
}
```



5.3-сурет. Жиым элементтерін өсуі бойынша реттеу

18-есеп. Берілген жиым элементтерін – x_{10} сол жиымда өсу реті бойынша орналастыру керек.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main()
{
  int xmin,x[10];
  int n,k,i;
  clrscr();
  printf("\nБерілген жиым
элементтері:");
  for (k=0; k<10; k++)
  {x[k] = rand() % 100;
  /* 32767-ге дейінгі
кездейсоқ сандар */
   printf(" %i",x[k]);}
  for (k=0; k<9; k++)
  { xmin=x[k];n=k;
    for (i=k+1; k<10; k++)
    if   (x[i] < xmin)
         {
           xmin=x[i];
```

```
            n = i;
          }
      x[n]=x[k]; x[k]=xmin;
    }
    printf("\nРеттелген жиым
            элементтері:");
    for (k=0; k<10; k++)
      printf(" %i",x[k]);
  getch();
}
```

Бақылау сұрақтары

1. Жиым дегенімз не?
2. Жиымдарға бастапқы мәндер қалай тағайындалады?
3. Жиымды сипаттау тәсілдері.
4. Жиым элементтерін енгізу және экранға шығару жолдары.
5. Кездейсоқ сандарды қалай шығаруға болады?
6. Жиымға кездейсоқ сандарды меншіктеу қалай орындалады?
7. Жиымды өңдеу есептерінің түрлері (кластары).
8. Жиымның ең үлкен (ең кіші) элементін анықтау.
9. Жиым элементтері қосындысын табу.
10. Жиым ішіндегі екі элементтің бір-бірімен орнын алмастыру.
11. Жиым элементтерін кері бағытта орналастыру.
12. Жиымның көрсетілген элементтерін өңдеу тәсілдері.
13. Жиымды сұрыптау (сорттау, реттеу) жолдары.

ТАПСЫРМАЛАР

Жиымдарды өңдеу кезінде төмендегі ережелерді есте сақтаған жөн.

1. Жиым өлшемі (ені, көлемі) тек константа немесе константалық өрнек бола алады. Жиым өлшемін идентификатор түріндегі (ат қойылған) константа арқылы беру дұрыс деп саналады.
2. Жиым элементтері нөлден бастап нөмірленеді, сол себепті элементтің ең үлкен нөмірі элементтер санынан бірге кем болады.
3. Индекс нөмірлерінің жиым шекарасынан асып кетпеуін программа қадағаламайды, оны программалаушы өзі бақылауы тиіс.
4. Нұсқауыш — жедел жады аймағының адресін сақтайтын айнымалы.
5. Жиым аты оның 0-элементіне нұсқауыш болып табылады.
6. Егер программаға енгізілетін элементтер мәні алдын ала белгілі болса, онда жиымды сипаттау кезінде оның мәндерін көрсетіп (инициалдау) кету керек. Жиымды глобальді айнымалы емес, локальді айнымалы етіп сипаттаған дұрыс.

7. Жиым элементтерін пернеден енгізгеннен гөрі rand() және random() функцияларын пайдаланып, кездейсоқ сандар тізбегін беру арқылы енгізген ыңғайлы болады.

8. Жиымдарды сұрыптау алгоритмдерінің жұмыс істеу жылдамдығы, алатын көлемі және қолданылу аймағы әр түрлі болып келеді.

Тапсырмаларды орындауға арналған нұсқаулар

A. Әрбір студент журналдағы өз нөміріне сәйкес (мұғалімнің көрсетуі бойынша) төрт есеп шығаруы тиіс.

B. Әр есептің блок-схемасы және соған сәйкес программасы құрылып, компьютерде сол программаның нәтижесін алу керек.

C. Екі блок-схема Word программасындағы *Сурет салу-Автофигуралар-Блок-схемалар* мүмкіндігімен сызылып көрсетілуі тиіс.

D. Берілген жиым элементтері мен нәтижелік элементтерді экранға шығарып, нәтижелерді жазып алу қажет немесе нәтижені бірден файлға жазуға тырысу керек.

Есептер

1. Кез келген сандардың бірөлшемді *A*(10) жиымындағы оң элементтерді екі есе кемітіңдер, ал теріс элементтері болса, онда оларды индекстерінің мәнімен ауыстырыңдар.

2. Бірөлшемді *A*(10) жиымындағы теріс элементтердің ең үлкенін табыңдар.

3. Бірөлшемді *A*(20) жиымындағы –5-тен кіші элементтердің қосындысын және олардың жалпы санын, сонымен бірге 5 санына еселік болатын элементтердің нөмірлерін анықтаңдар.

4. Бірөлшемді *A*(10) жиымындағы оң элементтердің квадраттарының арифметикалық ортасын есептеңдер.

5. Бірөлшемді *A*(100) жиымындағы теріс элементтердің бірінші тобындағы элементтердің санын анықтаңдар.

6. Бірөлшемді жиым мәндері берілген диапазонда жататын элементтерінің нөмірін басып шығарыңдар.

7. Бірөлшемді *A*(10) жиымындағы оң элементтердің ішінде мәні ең кішісін және оның индексін (нөмірін) табыңдар.

8. Бірөлшемді жиымда 2-элементті бірінші орынға, 3-ні 2-шімен және т.б. алмастырулар орындай отырып, 1-элементі соңғы орынға қойыңдар.

9. Берілген оң сандар тізбегіндегі қосындысы берілген саннан асып кетпейтін элементтердің санын табыңдар.

10. Ұтыс билетінің нөмірі алты орынды сан. Билет “бақытты” (алдыңғы және соңғы үш цифрының қосындысы өзара тең) немесе “табысты” (жұп орындарда тұрған цифрлардың қосындысы тақ орында тұрғандардың қосындысына тең) болатындығын анықтаңдар.

11. Оң және теріс сандар тізбегіндегі бірінші теріс санға дейінгі орналасқан сандар тізбегінің өсу ретімен орналасатындығын анықтаңдар.

12. *N* кәсіпорынның бір жылғы электр энергиясын тұтынуы туралы дерек бар. Осы бойынша энергия тұтынудың арифметикалық ортасын және энергияны ең көп үнемдеген кәсіпорынды анықтаңдар.

13. 14 аудан бойынша жанармай қорының мөлшері белгілі. Жанармаймен еңжақсы қамтылған үш ауданды анықтаңдар.

14. Берілген *A*(14) жиымы бойынша мына шарттарды қанағаттандыратын *B* жиымын құрыңдар:

B(1) = *A*(2); *B*(2) = *A*(2) * *A*(4); ... *B*(4) = *A*(2) * *A*(4) * ... * *A*(14)

15. Берілген *A*(13) жиымы бойынша мына шарттарды қанағаттандыратын *B* жиымын құрыңдар :

B(1) = *A*(1); *B*(2) = *A*(1) * *A*(3); ... *B*(7) = *A*(1) * *A*(3) * ... * *A*(13)

16. Берілген *A*(14) жиымы бойынша мына шарттарды қанағаттандыратын B жиымын құрыңдар:

B(1) = *A*(1) – *A*(14); *B*(2) = *A*(2) – *A*(13); ... *B*(7) = *A*(7) – *A*(8)

17. Берілген *A*(14) жиымы бойынша мына шарттарды қанағаттандыратын B жиымын құрыңдар:

B(1) = *A*(1)/*A*(14); *B*(2) = *A*(2)/*A*(13), ... *B*(7) = *A*(7)/*A*(8)

18. *A*(14) жиымы берілген. Жиым элементтерінің арифметикалық ортасын және ол мәнге еңжақын элементтің нөмірін анықтаңдар.

19. *A*(14) жиымы берілген. Екі жиым құрыңдар: оның біріншісі индекстері жұп сандар, ал екінші жиым индекстері тақ сан болатын элементтер болсын.

20. *N* элементтен тұратын бірөлшемді жиым берілген. Әрбір жолында *L* элемент болатын екіөлшемді жиым құрастырыңдар.

21. *A*(10,10) жиымы берілген. Бас диагональдағы ең кіші элемент пен қосымша диагональдағы ең үлкен элементтің орындарын ауыстырыңдар.

22. *A*(10,10) жиымы берілген. Бас диагональдағы ең үлкен элементі бар жолды нөмірі берілген жолмен алмастырыңдар.

23. *A*(17) жиымы берілген. 7-ден үлкен барлық элементтерді берілген жиымның ең үлкен элементімен ауыстырыңдар.

24. *A*(16) жиымы берілген. Алғашқы оң элементке дейінгі теріс элементтердің санын анықтаңдар.

25. *A*(18) жиымы берілген. 2-ден кіші барлық элементтерді 3 санымен ауыстырыңдар.

26. *A*(17) жиымы берілген. 0-ден үлкен барлық элементтерді 5-ке, ал қалғандарын 0-ге ауыстырыңдар.

27. *A*(14) жиымы берілген. *A* жиымының барлық тақ элементтері индекс нөмірлерімен ауыстырылатын *B* жиымын құрыңдар.

28. *A*(34) жиымы берілген. Тақ нөмірлі индекстерде тұрған элементтердің ең үлкенін анықтаңдар.

29. *A*(*N*) жиымы берілген. Алғашқы еңкіші (минимал) элементтің орнын (индексін) анықтаңдар.

30. *A*(*N*) жиымы берілген. Соңғы еңкіші элементтің орнын анықтаңдар.

31. *A*(*N*) жиымы берілген. Соңғы ең үлкен (максимал) элементтің орнын анықтаңдар.

32. *A*(*N*) жиымы берілген. Алғашқы ең үлкен элементтің орнын анықтаңдар.

33. *A*(*N*) жиымы берілген. Мәндері 3 пен 11 саны интервалында жататын элементтердің арифметикалық ортасын анықтаңдар.

34. *A*(12) = {2,5; 4,3; –0,57; 10,45; 1,5; –7,1; 11,4; 5,12; 4,9; 7,7; –12,3; 0,031} жиымы берілген. *B*(*I*) = SIN(*A*(*I*)) формуласы бойынша *B*(12) жиымын құрастырыңдар және *P* = *A*(1) * *B*(12) + *A*(2) * *B*(11) + ... + *A*(12) * *B*(1) өрнекті есептеп шығарыңдар.

35. Пернетақтадан жиымға бес бүтін мән енгізіңдер. Олар бір жолға үтір арқылы жазылады. Жиымның арифметикалық ортасын табыңдар.

36. Пернетақтадан *X* жиымының бес бүтін мәнін енгізіңдер. Жиым элементтерінің әрқайсысының түбірінің мәнін және квадратын экранға шығарыңдар.

37. Бес адамның аты-жөнінен тұратын жиым құрыңдар және экранға оларды соңынан бастап баған түрінде шығарыңдар.

38. Бес фамилиядан тұратын жиым құрастырып, олардың ішінен пернетақтадан енгізілген нақты әріптен басталатындарын экранға шығарыңдар.

39. Кездейсоқ сандар генераторының көмегімен *A*[1..7] жиымын құрыңдар және оны экранға шығарыңдар. Оның барлық элементтерін 2 есеге арттырыңдар.

40. Кездейсоқ сандар генераторының көмегімен элементтері –10 мен 10 сан аралығында болатын *A*[1..8] жиымын құрыңдар және оны экранға шығарыңдар. Жиымның теріс элементтерінің санын есептеңдер.

41. Кездейсоқ сандар генераторының көмегімен элементтері –20 мен 10 саны аралығында болатын A[1..12] жиымын құрыңдар және оны экранға шығарыңдар. Жиымның барлық теріс элементтерін 0-ге ауыстырыңдар.

42. Кездейсоқ сандар генераторы арқылы элементтері –15 пен 30 саны аралығында болатын бүтін саннан A[1..15] жиымын құрастырыңдар және оны экранға шығарыңдар. Жиымның ең үлкен элементін және оның индексін анықтаңдар.

43. 8 сағ-тан 20 сағ-қа дейінгі уақыт аралығында ауа температурасы сағат сайын өлшенеді. Осы аралықта температураның төмендегені мәлім. Температураның алғашқы теріс мәні қай сағатта пайда болғандығын анықтаңдар.

44. Қараша айында он күн бойына ауа температурасы туралы деректер жиымда сақталған. Температураның –10-нан қанша рет төмен болғанын анықтаңдар.

45. Балқаш көлі жағалауының температурасы туралы мәлімет қыркүйек айында он күн бойы жиымда сақталған. Осы уақыт ішінде неше күн шомылуға қолайлы болатынын анықтаңдар.

46. Сәуір айының онкүндік ауа температурасы мен жауын-шашын мөлшері туралы мәлімет жиымда сақталады. Осы он күнде жауған жаңбыр және қардың мөлшерін анықтаңдар.

47. Желтоқсан айының онкүндік ауа температурасының мәліметі жиымда сақталған. Онкүндік температура мәліметі бойынша қанша рет орта температурадан жоғары ауытқу болатындығын анықтаңдар.

48. Қараша айының он күн ішіндегі желдің (солтүстік, оңтүстік, шығыс, батыс) бағыты және соғу күшінің мәліметтері жиым түрінде сақталған. Неше күн желдің жылдамдығы 8 м/с-тан артық жылдамдықпен соққандығын анықтаңдар?

49. Бүтін саннан тұратын 15 элементті жиым құрыңдар және олардың арасындағы ең кіші элементті анықтаңдар.

50. Жерге еркін түсу кезінде дененің 1, 2, ... , 10 с ішінде жүріп өткен қашықтықтарынан құралатын нақты сандардың сызықтық жиымын құрыңдар.

51. Бүтін сандардың сызықтық жиымы берілген. Оның кему ретімен орналасқандығын тексеріңдер.

52. Бүтін сандардың сызықтық жиымындағы оң элементтердің қосындысын табыңдар. Жиымның өлшемі — 10. Жиымды пернетақтадан толтырыңдар.

53. Бүтін сандар жиымындағы жұп нөмірлі элементтердің қосындысын табыңдар. Жиымның өлшемі 20. Жиымды 100 бен 200 арасындағы кездейсоқ сандармен толтырыңдар.

54. Бүтін сандар жиымындағы 7-ге еселік болатын элементтердің көбейтіндісін табыңдар. Жиым өлшемі 15. Жиымды 10 мен 50 арасындағы кездейсоқ сандармен толтырыңдар.

55. Нақты сандар жиымындағы тақ нөмірлі элементтердің қосындысын табыңдар. Жиым өлшемі 20. Жиымды 100 бен 200 арасындағы кездейсоқ сандармен толтырыңдар.

56. Бүтін сандар жиымындағы барлық 0-ден кіші элементтердің көбейтіндісін табыңдар.

57. Бүтін сандар жиымында: “2-ге бөлгенде қалдық 3-ке тең” шартын қанағаттандыратын барлық элементтердің қосындысын табыңдар. Жиымның өлшемі 20. Жиымды 200 бен 300 аралығындағы кездейсоқ сандармен толтырыңдар.

58. Нақты сандар жиымындағы берілген саннан кіші барлық элементтердің қосындысын табыңдар. Жиымның көлемі 20. Жиымды 50 мен 100 аралығындағы кездейсоқ сандармен толтырыңдар.

59. Нақты сандар жиымындағы берілген саннан кіші барлық элементтердің көбейтіндісін табыңдар. Жиымның көлемі 10. Жиымды 50 мен 100 аралығындағы кездейсоқ сандармен толтырыңдар.

60. Жиымдағы 3 пен 9-ға қалдықсыз бөлінетін элементтердің көбейтіндісін табыңдар. Жиымның көлемі 10. Жиымды 5 пен 500 аралығындағы кездейсоқ сандармен толтырыңдар.

61. Бүтін сандар жиымында элементтердің арифметикалық ортасынан кіші болатын барлық элементтердің қосындысын табыңдар. Жиымның көлемі 20. Жиымды 150 мен 300 аралығындағы кездейсоқ сандармен толтырыңдар.

62. Бүтін сандар жиымында 5-ке де, 8-ге де бөлінетін элементтердің қосындысын табыңдар. Жиымның көлемі 30. Жиымды 500 бен 100 аралығындағы кездейсоқ сандармен толтырыңдар.

63. Науқастың тәулік бойы температурасын талдайтын программа жазыңдар: температураның минимал, максимал және арифметикалық орта мәнін анықтаңдар. Температура тәулік бойына 6 рет өлшенеді және нәтиже пернетақтадан *T* жиымына енгізіледі.

64. Сызықтық жиымға бір жылдың әрбір 12 айының жауын-шашын мөлшері туралы мәлімет тіркелген. Жыл бойы жауған жауын-шашынның мөлшерін, оның орта айлық мөлшерін, құрғақшылық (жауын-шашын мөлшері 30 мм-ден аз) болған айлардың саны, жылдың ең құрғақ айын анықтайтын программа жазыңдар.

65. Бірөлшемді жиымның жұп элементтерінің санын табыңдар.

66. Бірөлшемді жиымда берілген *a* санының алдындағы санға тең алғашқы жұп элементтердің санын табыңдар.

67. Бірөлшемді жиымда реті бойынша алғашқы ең кіші элементтен кейін орналасқан элементтердің арифметикалық ортасын есептеп шығарыңдар.

68. Бірөлшемді жиым элементтері –2 мен 10 санының аралығында болса, онда сол элементтердің арифметикалық ортасын есептеп шығарыңдар.

69. Егер бірөлшемді жиымда еңболмағанда, бір теріс және бір жұп сан болса, онда *t* айнымалысына “АҚИҚАТ” деген мән беріңдер.

70. Бірөлшемді жиымда алғашқы нөлге тең элементке дейін орналасқан жұп элементтерінің қосындысын есептеп шығарыңдар.

71. Егер бірөлшемді жиымның жалғыз бір максимал элементі болса және ол берілген *a* санынан артық болмаса, онда *t* айнымалысына “АҚИҚАТ” деген мән беріңдер.

72. Бірөлшемді жиымның жұп нөмірлі элементтерінің ең үлкенін табыңдар. Оның біреу ғана екенін анықтаңдар.

73. Ең кіші элементтің алдында орналасқан элементтерді жою арқылы сызықтық жиымды қысқартыңдар.

74. *A*(20) жиымы берілген. Оның барлық нөлге тең емес элементтерінің көбейтіндісін табыңдар.

75. *X*(20) жиымында әр элемент 0, 1 не 5 сандарына тең. Жиым элементтерін алдымен барлық нөлдері, одан кейін бірлері, содан кейін бестіктері орналасатындай етіп орындарын ауыстырыңдар. Қосымша жиым құрылмайды.

76. Сызықтық жиым да қосындысы максимал болатын екі элемент табыңдар.

77. 20 элементтен тұратын жиым енгізіңдер және онда бірдей мәнді элементтердің бар-жоғын анықтаңдар.

78. Өлшемдері бірдей екі сызықтық жиым берілген. Осы жиымдардың индекстері бірдей орындарында орналасқан элементтердің көбейтіндісінен тұратын үшінші жиым құрыңдар.

79. Бүтін сандардың бірөлшемді жиымы берілген. Ондағы ең көп кездесетін санның неше рет қайталанатынын табыңдар.

80. 10 бүтін саннан тұратын бірөлшемді жиым берілген. Ондағы әр түрлі сандардың санын есептеңдер.

81. Егер бірөлшемді жиымның барлық элементтері оң болса, онда “шарт орындалады”, басқа жағдайда “шарт орындалмайды” деген хабарды басып шығарыңдар.

6. ЕКІ ӨЛШЕМДІ ЖИЫМДАР

Екі өлшемді жиымды – матрицаны пайдалану үшін тік жақшалар ішінде олардың екі өлшемінің де енін көрсету керек. Мысалы:

```
int a[4][3];
```

алғашқы сан жолдар санын, ал екінші сан бағаналар санын көрсетеді, *a* жиымы 12 элементтен тұрады. Оларға бастапқы мәнді былай береміз:

```
int a[4][3]={ {0,1,2},
              {3,4,5},
              {6,7,8},
              {9,10,11}
            };
```

ішкі жүйелі жақшаларды қоймаса да болады:

```
int a[4][3]={0,1,2,3,4,5,6,7,8,9,10,11};
```

Келесі түрде сипаттау жолдардың тек бірінші элементтерін ғана анықтайды, қалған элементтер 0-ге тең болып саналады:

```
int a[4][3]={ {0},{3},{6},{9} };
```

Егер ішкі жүйелі жақшалар алынып тасталса, онда мағынасы өзгереді.

```
int a[4][3]={ 0,3,6,9 };
```

мұнда бірінші жолдың 3 элементі мен екінші жолдың бірінші элементі анықталады да, қалғандары 0 болып саналады.

Екі өлшемді жиымды инициалдау қабаттасқан циклдер арқылы орындалады.

1-мысал.

```
/* a[3][4] жиымы  элементтерін rand() арқылы енгізу
және экранға шығару */
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main()
{
 const int row=3, col=4;
 int a[row][col];
 clrscr();
 for (int i=0; i<row; i++)
    for (int j=0; j<col; j++)
        a[i][j]=rand()%100-50;
 printf("\na[3][4] жиым элементтері мәндері:");
 for (i=0; i<row; i++)
    for (j=0; j<col; j++)
        printf(" %i",a[i][j]);
 getch();
}
```

Матрицаларды өңдейтін негізгі алгоритмдер ретінде бір өлшемді жиымдарды өңдеу кезінде қолданылған алгоритмдер саналады. Жалпы матрицаларды өңдейтін барлық алгоритмдерді екі топқа бөліп қарастыруға болады, олар:

1. матрицаның барлық элементтерін өңдейтін алгоритмдер.
2. матрицаның әр жолы немесе әр бағанасы элементтерін жеке-жеке өңдейтін алгоритмдер.

Матрицаның барлық элементтерін өңдейтін алгоритмдер

2-мысал. Нақты сандардан тұратын $a_{4,6}$ матрицасы берілген. Мынадай өрнекті $Z = \frac{P1}{|P2|}$ есептеу керек, мұндағы P1 и P2 – сәйкесінше алынған матрицаның оң және теріс элементтерінің көбейтіндісі.

```
/* a[4][6] матрицасы берілген. z=p1/|p2| есептеу
керек, p1 и p2 – матрицаның
   оң және теріс элементтерінің көбейтіндісі  */
#include <math.h>
#include <conio.h>
#include <stdio.h>
void line()
 {printf("------------------------------\n");
 return;}
main ()
{
  static int a[4][6]={
                        {5,-11,4,-2,5,6},
                        {3,3,-12,-5,7,8},
                        {2,3,-3,14,-9,-3},
                        {-9,3,-6,14,9,-3}
                      };
  int i,j;
  float p1, p2, z;
  clrscr();

  printf("Берілген матрица :\n");
  line();
  for (i=0; i<4; i++)
   {for (j=0; j<6; j ++)
      printf(" %3i ", a[i][j]);
    printf("\n");}
  line();
```

```
    /* Матрицаны өңдеу */
   p1 = 1;
   p2 = 1;
      for (i=0; i<4; i++)
       for (j=0; j<6; j ++)
        {if  (a[i][j]>0)  p1 = p1*a[i][j];
         if  (a[i][j]<0)  p2 = p2*a[i][j];
        }
         z = p1/abs(p2);
      printf("      z = %f\n",z);
      line();
      getch();
}
```

3-мысал. Бүтін сандардан тұратын квадрат $b_{5,5}$ матрицасы берілген. Оның бас диагоналының сол жағында және оң жағында орналасқан нөлге тең элементтері санын анықтап, солардың айырмасының модулін табу керек.

Мынадай белгілеулер енгізейік:

L1 – бас диагональдың сол жағында (төменінде) орналасқан элементтер саны;

L2 – бас диагональдың оң жағында (жоғарысында) орналасқан элементтер саны;

L= |L1-L2| – солардың айырмасы модулі.

```
#include <math.h>
#include <conio.h>
#include <stdio.h>
void line()
 {printf("------------------------\n");
 return;}
main ()
{
  static int b[5][5]={
                   {5,0,0,0,0},
                   {0,3,12,0,0},
                   {0,33,13,14,0},
                   {0,23,0,14,0},
                   {35,0,13,14,9},
                  };
  int i,j;
  int L1,L2,L;
  clrscr();
  printf("Берілген матрица :\n");
```

```
  line();
  for (i=0; i<5; i++)
   {for (j=0; j<5; j ++)
      printf(" %3i ", b[i][j]);
    printf("\n");}
  line();

 L1 = L2 = 0;
 for (i=0; i<5; i++)
  for (j=0; j<5; j ++)
   if  (b[i][j]==0)
    {if  (i>j)  L1 = L1+1;
     if  (i<j)  L2 = L2 +1;
    }
  L= abs(L1 - L2);
  printf("       L = %i ",L);
  getch();
}
```

Екінші типтегі есептер алгоритмдері

4-мысал. Бүтін сандардан тұратын $a_{3,6}$ матрицасы жолдарының алғашқы элементін осы жолдың минимальды элементімен алмастыру керек. Нәтижелік $a_{3,6}$ матрицасы элементтерін экранға шығару қажет.

```
/*  a[3][6] матрицасы жолдарының алғашқы элементін
осы жолдың минимальды элементімен алмастыру керек.
Натижелік X матрицасы элементтерін экранға шығару
кажет.*/
#include <math.h>
#include <conio.h>
#include <stdio.h>
void line()
 {printf("------------------------------\n");
 return;}
main ()
{
  static int a[3][6]={
                      {5,-11,4,-2,5,6},
                      {2,3,-3,14,-9,-3},
                      {-9,3,-6,-14,9,-3}
                     };
  int i,j,jmin,min;
  clrscr();
  printf("Берілген матрица :\n");line();
```

```
  for (i=0; i<3; i++)
   {for (j=0; j<6; j ++)
      printf(" %3i ", a[i][j]);
    printf("\n");
   }
  line();

  for (i=0; i<3; i++)
   { min=+1E6;
    for (j=0; j<6; j ++)
     if (a[i][j]<min)
       {min=a[i][j];
      jmin=j;
       }
    a[i][jmin]=a[i][0];

 a[i][0]=min;
   }
   printf("Өңделген матрица :\n");line();
   for (i=0; i<3; i++)
   {for (j=0; j<6; j ++)
     printf(" %3i ", a[i][j]);
    printf("\n");
   } line();
  getch();
}
```

5-мысал. Бүтін сандардан тұратын $a_{3,4}$ матрицасының әрбір бағаналарының арифметикалық орташа мәнін анықтап, оларды бір өлшемді s_4 жиымы ретінде бейнелеу керек.

```
/*  a[3][4] матрицасының әрбір бағаналарының
арифметикалық орташа мәнін анықтап,оларды бір өлшемді
s[4] жиымы ретінде бейнелеу керек. */

//әрбір бағана қосындысы және солардың орташа мәні
#include <conio.h>
#include <stdio.h>
main ()
{
  static int a[3][4]={
                      {5,11,4,2},
                      {3,3,12,5},
                      {2,3,3,14}
```

```
                    };
  int i,j;
  float s[4];
  clrscr();
  printf("Берілген матрица :\n");
  printf("----------------------\n");
  for (i=0; i<3; i++)
   {for (j=0; j<4; j ++)
      printf(" %3i ", a[i][j]);
    printf("\n");}
    /* матрицаны өңдеу */
  printf("----------------------\n");
  printf("");
  for (j=0; j<4; j++)
   { s[j]=0;
     for (i=0; i<3; i ++) s[j]+=a[i][j];
       printf(" %4.2f",s[j]/3);
   }
 printf("\n----------------------");
getch();
}
```

Бұл программада s[j] жиымының әрбір элементін есептеуде қабаттасқан екі цикл қолданылған, онда j индексі сыртқы цикл параметрі, ал і индексі – ішкі цикл параметрі. Осы тәсіл матрица элементтерін бағаналар бойынша өңдеу ісін жүзеге асырады.

6-мысал. Берілген жиымның әрбір жолындағы элементтері қосындыларын және сол қосындылардың орташа мәнін анықтау керек.

```
// әр жол қосындысы және солардың арифметикалық ортасы
#include <conio.h>
#include <stdio.h>
main ()
{ static int a[3][4]={
                       {5,3,4,2},
                       {3,3,4,5},
                       {2,3,3,4}
                      };
int i,j,s=0;
float c=0;
clrscr();
for (i=0; i<3; i++)
```

```
 {for (j=0; j<4; j ++) s+=a[i][j];
  printf("%i-жол косындысы:%i\n",i+1,s);
  c+=s;
 }
printf("--------------------");
printf("\nарифм.ортасы %5/2f", c/4) ;
}
```

7-мысал. Берілген жиымның әрбір бағанадағы элементтері қосындыларын және сол қосындылардың орташа мәнін анықтау керек.

```
// әрбір бағана қосындылары мен солардың орташа мәнін
анықтау
#include <conio.h>
#include <stdio.h>
main ()
{
  static int a[3][4]={
                        {5,11,4,2},
                        {3,3,12,5},
                        {2,3,3,14}
                      };
  int i,j,s;
  float c;
  clrscr();

  printf("Берілген матрица :\n");
  printf("----------------------\n");
  for (i=0; i<3; i++)
   {for (j=0; j<4; j ++)
      printf(" %3i ", a[i][j]);
    printf("\n");}
    /* Матрицаны өңдеу */
  printf("----------------------\n");
  printf("s=");
  for (j=0; j<4; j++)
   { s=0;
     for (i=0; i<3; i ++) s+=a[i][j];
       printf("%3i  ",s); // қосынды s
   c+=s;
   }
 printf("\n----------------------");
 printf("\n арифм. ортасы %5.2f",c/4);
 getch();
```

```
}
```
8-мысал. Берілген $a_{4,4}$ жиымының бас диагоналындағы элементтерді нөлге, ал қосалқы диагоналындағы элементтерді – бірге теңестіру программасы.
```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

main()
{
  int a[4][4]={
            {6,8,9,2},
            {5,3,4,2},
            {3,3,4,5},
            {2,3,3,4}
             };
  int i,j;
  clrscr();
  printf("Берілген матрица :\n");
  for (i=0; i<4; i++)
   {for (j=0; j<4; j ++)
      printf(" %2i ", a[i][j]);
    printf("\n");}
    /* Матрицаны өңдеу */
  for (i=0; i<4; i++)
       a[i][i]=0;
  for (i=0; i<4; i++)
    for (j=0; j<4; j ++)
       if (i+j==3)  a[i][j]=1;
   /* Өңделген матрица элементтерін шығару */
  printf("Нәтижелік матрица :\n");
  for (i=0; i<4; i++)
   {for (j=0; j<4; j ++)
     printf(" %2i ", a[i][j]);
    printf("\n");}
getch();
}
```
9-мысал. Берілген $a_{3,3}$ жиымының бас диагоналынан төмен орналасқан элементтер қосындысын анықтау программасы.
```
#include <conio.h>
#include <stdio.h>
main ()
```

```
{
  static int a[3][3]={
                      {0,1,2},
                      {3,4,5},
                      {6,7,8}
                     };
  int i,j,r,s;
  clrscr();
   printf("Енгізілген a[3][3] жиым элементтері:\n");
  for (i=0; i<3; i++)
   {for (j=0; j<3; j++)
       printf(" %2i",a[i][j]);
    printf("\n");
   }
   /* бас диагоналдан төмен орналасқан элементтер
      қосындысын анықтау */
   s=0;
   for (i=0; i<3; i++)
   for (j=0; j<3; j++)
     if (j<i) s+=a[i][j];
    printf("\ns=%i",s);
   getch();
}
```

10-мысал. Берілген $a_{3,3}$ жиымының әрбір жолындағы элементтерді өсуі бойынша реттеп орналастыру программасы.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main ()

{ static int a[3][3]={
                       {8,7,6},
                       {5,4,3},
                       {2,1,0}
                      };
  int i,j,r,s,n=3,amin,m,k;
  clrscr();
   printf("Берілген матрица:\n");
  for (i=0; i<3; i++)
   {for (j=0; j<3; j++)
       printf(" %2i",a[i][j]);
    printf("\n");
```

```
   }
  for (i=0; i<n; i++)   //жолды таңдау
 {                      //мин тауып алмастыру
  for (k=0; k<n-1; k++)
   { amin=a[i][k];m=k;
    for (j=k+1; j<n; j++)
      if (a[i][j] < amin)
         {amin=a[i][j]; m = j;}
    a[i][m]=a[i][k]; a[i][k]=amin;
   }
 }
printf("\nНәтижелік матрица:\n");
for (i=0; i<3; i++)
   {for (j=0; j<3; j++)
       printf(" %2i",a[i][j]);
    printf("\n");
   }
getch();
```

11-мысал. Берілген $a_{3,3}$ жиымының әрбір бағанасындағы элементтерді өсуі бойынша реттеп орналастыру программасы.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main ()
{
  static int a[3][3]={
                      {8,7,6},
                      {5,4,3},
                      {2,1,0}
                     };
  int i,j,r,s,n=3;
  int amin;
  int m,k;
  clrscr();
   printf("Берілген матрица:\n");
  for (i=0; i<3; i++)
   {for (j=0; j<3; j++)
       printf(" %2i",a[i][j]);
    printf("\n");
   }
  for (j=0; j<n; j++)   //бағананы таңдау
 {for (k=0; k<n-1; k++) //мин тауып алмастыру
```

```
   { amin=a[k][j];m=k;
    for (i=k+1; i<n; i++)
      if (a[i][j] < amin)
         {amin=a[i][j]; m = i;}
    a[m][j]=a[k][j]; a[k][j]=amin;
   }
 }
printf("\nСұрыпталған матрица элементтері:\n");
for (i=0; i<3; i++)
   {for (j=0; j<3; j++)
       printf(" %2i",a[i][j]);
    printf("\n");
   }
getch();
}
```

Екі өлшемді жиымдармен жұмыс істеу кезінде нұсқауыштарды қолдану

$A_{3,2}$ жиым берілген болсын. Олар бүтін сандар, яғни

```
int a[3][2];
int  *pri;
pri = a; // бұл pri=a[0][0] деген сөз
```

a – жиымның аты немесе a[0][0] элементінің адресі

a=&a[0][0];

pri нұсқауышына 1-ді қоссақ, pri+1 деген нұсқауыш a[0][1] элементіне сілтейді. Бұл жиымды қарастырғанда, келесі теңдеулер дұрыс болып табылады:

```
pri  == &a[0][0];
pri+1== &a[0][1];

pri+2== &a[1][0];
pri+3== &a[1][1];

pri+4== &a[2][0];
pri+5== &a[2][1];
```

Екі өлшемді жиым бір өлшемді жиымдардан құрастырылған жиым ретінде қарастырылуы мүмкін. Яғни берілген екі өлшемді жиым үш жолдан тұрады, ал әрбір жол екі элементтен тұратын жиым болып табылады.

Бірінші жол аты – **a[0],**

екінші жол аты – **a[1],**

үшінші жол аты – **a[2].**

Жиымның аты берілген жиымның нұсқауышы болып табылады, яғни ол жиымның 1-элементіне сілтейді.

Екі өлшемді жиымның осы қасиеті бір өлшемді жиымға арналған функцияны екі өлшемді жиыммен жұмыс істеуге мүмкіндік береді.

12-мысал. Төмендегі b[3][4] матрицасы жолдарының арифметикалық ортасын табатын программада функция қолданылған.

```
/* b[3][4]  матрицасы жолдарының қосындысы */
#include <conio.h>
#include <stdio.h>
float f1(int x[], int n)
 { int k; float s;
   for (k=0,s=0;k<n; k++)
      s+=x[k];
   return(s/n);
 }
main ()
{ int i,b[3][4]={ {6,4,3,3},
                  {7,5,3,3},
                  {8,4,2,6}
                };
  clrscr();
  for (i=0;i<3;i++)
     printf("%d жолының орташа мәні: %f\n",
i,f1(b[i],4));
  getch();
}
```

Бақылау сұрақтары

1. *Екі өлшемді жиымды – матрицаны сипаттау, бастапқы мәндерді тағайындау тәсілдері.*
2. *Екі өлшемді жиымды инициалдауды қабаттасқан циклдер арқылы орындау.*
3. *Матрица элементтерін толық өңдейтін алгоритмдер.*
4. *Матрицалардың көрсетілген элементтерін өңдеу тәсілдері.*
5. *Матрицалардың диагоналына байланысты орналасқан элементтерін өңдеу жолдары.*
6. *Матрицалардың жолдарының және бағаналарының ең үлкен (кіші) элементтерін, қосындыларын, көбейтінділерін табу.*
7. *Матрицалардың жолдарында және бағаналарында орналасқан элементтерді өсуі (кемуі) бойынша реттеу алгоритмдері.*

ТАПСЫРМАЛАР

1. Екіөлшемді $A(10,10)$ жиымда әрбір жол үшін оң элементтердің қосындысын есептеп шығарыңдар.

2. Екіөлшемді $A(10,10)$ жиымындағы оң элементтердің санын есептеп шығарыңдар.

3. Екіөлшемді $A(10,10)$ жиымындағы ең үлкен элементті анықтаңдар.

4. Екіөлшемді $A(10,10)$ жиымы берілген. A жиымының әрбір жолындағы элементтердің көбейтіндісіне теңболатын бірөлшемді B жиымын алыңдар.

5. Екіөлшемді $A(10,10)$ жиымы берілген. A жиымының әрбір жолындағы ең үлкен элементтің мәндеріне теңболатын бірөлшемді B жиымын алыңдар.

6. Екіөлшемді $A(10,10)$ жиымы берілген. A жиымының әрбір жолындағы теріс элементтің мәндеріне тең болатын бірөлшемді B жиымын алыңдар.

7. Екіөлшемді $A(10,10)$ жиымында теріс элементтері бар жолдардың санын есептеп шығарыңдар.

8. Екіөлшемді жиым жолдарындағы ең үлкен элементтердің қосындысын есептеп шығарыңдар.

9. Екіөлшемді жиым әрбір бағандағы (тік жолдағы) элементтердің қосындысын есептеп шығарыңдар.

10. Екіөлшемді жиымның әрбір жолындағы теріс сандардың санын, қосындысын және арифметикалық ортасын есептеп шығарыңдар.

11. Екіөлшемді $A(4,4)$ жиымындағы индекстерінің қосындысы 4-ке теңболатын элементтерінің қосындысын есептеп шығарыңдар.

12. Екіөлшемді $A(4,7)$ жиымындағы оң элементтердің арифметикалық ортасын және нөлге тең элементтердің санын есептеп шығарыңдар.

13. Екіөлшемді $A(10,10)$ жиымының бас диагоналі бойындағы элементтердің ең үлкенін табыңдар.

14. Екіөлшемді $A(10,10)$ жиымының қосымша диагоналі бойындағы ең үлкен элементті табыңдар.

15. Екіөлшемді $A(10,10)$ жиымының қосымша диагоналі бойындағы ең үлкен элементті табыңдар.

16. Екіөлшемді $A(4,7)$ жиымының әрбір жолындағы берілген диапазонда жататын элементтердің арифметикалық ортасын есептеп шығарыңдар.

17. Екіөлшемді *A*(10,10) жиымында элементтердің арифметикалық ортасы жиымның барлық элементтерінің арифметикалық ортасынан кем болатын бағанның нөмірлерін анықтаңдар.

18. Екіөлшемді *A*(10,10) жиымында 3- және 1-жолдардың орындарын ауыстырыңдар.

19. Екіөлшемді *A*(7,7) жиымында бас диагональдағы элементтерді әрбір жолдың ең үлкен мәнімен алмастырыңдар.

20. Екіөлшемді *A*(10,10) жиымында бас диагональдан жоғарғы және төмен орналасқан элементтердің қосындысын есептеп шығарыңдар.

21. Екіөлшемді жиымда берілген жолдың ең үлкен элементінің мәнін және нөмірін анықтаңдар.

22. Екіөлшемді жиымда әр баған үшін берілген бағанның мәнінен кіші болатын элементтердің арифметикалық ортасын есептеп шығарыңдар.

23. Бүтін сандардан құралған 10Х10 матрицасының ең үлкен элементінің жолы мен баған нөмірін шығарып беретін программа жазыңдар.

24. *A*(5,5) жиымы және *k* саны берілген. Әрбір жолдың элементтерін осы жолдағы бас диагональда орналасқан диагоналдық элементке бөліңдер.

25. *A*(10,10) жиымы берілген. Осы жиымның бас диагоналінің элементтерінен тұратын бірөлшемді жиым құрыңдар .

26. *A*(10,10) жиымы берілген. *k*- және 1-жолдардың орындарын ауыстырыңдар.

27. *A*(10,10) жиымы берілген. Одан бірөлшемді жиым құрастырыңдар.

28. Бүтін санды *x* [0...5, 0...4] жиымы берілген. Оның 5-тен кіші барлық элементтерін 111 санымен ауыстырыңдар.

29. Бүтін санды *B* [0...4, 0...3] жиымы берілген. Оның элементтерін олардың квадраттарымен ауыстырыңдар.

30. Нақты сандар жиымы *A*[0...5, 0...3] берілген. Оның теріс элементтерінің индекстерін басып шығарыңдар.

31. Екіөлшемді бүтін санды *A*[0..10, 0..7] жиымын құрыңдар және оның екі тақ санды индекстерінің барлық элементтерінің қосындысын табыңдар.

32. Бүтін санды *A* [0..10, 0..7] жиымы берілген. Жиымның 5-ке қалдықсыз бөлінетін элементтерінің қосындысын табыңдар.

33. 3х3 матрицаның бүтін сандық элементтерін пернетақтадан енгізіңдер және әрбір баған элементтерінің қосындысын есептеп шығарыңдар.

34. Бүтін санды B[0..5,0..5] жиымы берілген. Оның диагональдарынан сол және оң жақта орналасқан элементтерін анықтаңдар.

35. Бүтін санды B [0..5, 0..5] жиымы берілген. Диагональдың оңжақтағы элементтерінің қосындысын, сол жақтағы элементтерінің көбейтіндісін табыңдар.

36. Бүтін санды B [0..5, 0..5] жиымы берілген. Жиымның ең үлкен элементін табыңдар және оның диагоналының қай жағында орналасқаны туралы хабарды экранға шығарыңдар.

37. Бүтін санды B [0..5, 0..5] жиымы берілген. Жиымның ең кіші элементін табыңдар және оның диагональдың қай жағында орналасқаны туралы хабарды экранға шығарыңдар.

38. Бүтін санды B [0..5,0..5] жиымы берілген. Жиым диагоналінің сол жағынан жоғары орналасқан элементтердің қосындысын табыңдар.

39. Бүтін санды B [0..5,0..5] жиымы берілген. Жиым диагоналінің сол жағынан төмен орналасқан элементтердің көбейтіндісін есептеп шығарыңдар.

40. Бүтін санды B [0..5,0..5] жиымы берілген. Жиым диагоналінің сол жағынан төмен орналасқан теріс таңбалы элементтерінің санын табыңдар.

41. Бүтін санды B[0..5,0..5] жиымы берілген. Жиым диагоналінің сол жағынан жоғары орналасқан оң таңбалы элементтерінің санын табыңдар.

42. Бес цехтың әрқайсысының 4 бөлімшесіндегі барлық шикізат туралы мәлімет кестесі берілген. Шикізаты ең аз цехтың нөмірін анықтаңдар.

43. A[0..3, 0..15] жиымы берілген. Оның ішінде өзара тең екі элементтің индекстерін басып шығарыңдар.

44. a_1, a_2, a_3 сандары берілген. Элементтері $B[i,j] = a_i - 3a_j$ болатын бүтін санды B [0..3, 0..3] жиымын анықтаңдар.

45. Нақты a_1, a_2, ... , a_{10}, b_1, b_2, ... , b_{20} сандары берілген. Элементтері $a_{i,j} = i + 2j$ болатын бүтін санды A [0..10, 0..12] жиымын алыңдар.

46. Өлшемдері 5*5 матрицаның әрбір элементінің мәні қиылыатын жол мен бағана нөмірінің қосындысына тең болатын элементтерінің қосындысын есептеңдер.

47. Нақты [0..7, 0...7] жиымын алыңдар, оның 1-жолы $a_{1j} = 2j + 3$ формуласымен, 2-жолы $a_{2j} = j + 3/(2+j)$ формуласымен беріліп, содан кейінгі әрбір жол алдыңғы екі жолдың қосындысына тең болатын болсын.

48. Натурал n саны берілген. Егер $a_{i,j}$= sin(i+j/2) болса, A[0..n, 0..n] жиымында қанша оң элемент болатындығын анықтаңдар.

49. Бүтін санды A[0..4, 0..5] жиымы берілген. Әрбір бағанның арифметикалық ортасын табыңдар.

50. Барлық элементтері нөлге теңемес n..m өлшемді нақты жиым берілген. Бұл жиымның модулі бойынша ең үлкен элементіне басқа барлық элементтерін бөлу арқылы жаңа жиым алыңдар.

51. Бүтін санды A[0..4, 0..5] жиымы берілген. Соңғы жолдан басқа жолдарды әрбір элементі бойынша азайту арқылы матрицаны түрлендіріңдер.

52. Екіөлшемді C жиымының әрбір жолын өсу ретімен орналастыратын программа құрыңдар.

53. m..n өлшемді матрицаның әрбір жолын кему ретімен орналастыратын программа құрыңдар.

54. Бүтін санды A[0..4, 0..5] жиымы берілген. Құрылымында кем дегенде бір рет 10-ға тең элементі бар жолдардың нөмірін анықтаңдар.

55. m..n өлшемді матрицаның әрбір бағанасын өсу ретімен орналастыратын программа құрыңдар.

56. A[0..5, 0..5] жиымы берілген. Бұл жиымның әрбір жолының элементтерін диагональдің сол жағында орналасқан элементтерге бөлу жолымен алынған жаңа жиым құрыңдар.

57. A[0..5, 0..6] жиымы берілген. Оның бірінші және соңғы жолдарының орындарын алмастырыңдар.

58. A[0..5, 0..6] жиымы берілген. Оның бірінші және соңғы бағаналарының орнын алмастырыңдар.

59. Тікбұрышты матрица берілген. Элементтерінің қосындысы ең үлкен болатын жолды табыңдар.

60. Тікбұрышты матрица берілген. Элементтерінің қосындысы ең үлкен болатын бағананы табыңдар.

61. Өлшемі 4x8 болатын бүтін сандар жиымындағы барлық жұп нөмірлі элементтерінің қосындысын табыңдар.

62. Бүтін сандардың 5x5 өлшемді жиымындағы бас диагональда орналасқан барлық элементтердің қосындысын табыңдар.

63. Бүтін сандардың 7x4 өлшемді жиымындағы максимал элементтің жолы мен бағана нөмірлерін табыңдар.

64. Бүтін сандардың 6x5 екіөлшемді жиымы бар. Элементтерінің арифметикалық орта мәні максимал болатын жолдың нөмірін табыңдар.

65. Бүтін сандардың 5x9 өлшемді жиымында бірдей нөмірлі жол мен бағананың орнын алмастырыңдар.

66. Бүтін сандардың екі өлшемді жиымындағы жолдың максимал элементтерінің арасындағы минимал элементті табыңдар.

67. Бүтін сандардың екіөлшемді жиымында максимал элементі бар бағанды өшіріп тастаңдар.

68. Бүтін сандардың екіөлшемді жиымындағы қайталанбайтын барлық элементтерді табыңдар.

69. Екіөлшемді жиымды 1-ден 100-ге дейінгі бүтін сандармен орама (спираль) бойымен толтырыңдар.

70. Бүтін сандардың екіөлшемді жиымының барлық элементтерін сол жолдардағы элементтердің қосындысынан солардың ішіндегі ең кіші элементтер айырмасымен алмастырыңдар.

71. Бүтін сандар жиымының жолдарын кему реті бойынша сұрыптаңдар.

72. Жиымның тақ орындардағы бағаналарында тұрған элементтерді өсу ретімен орналастыра отырып сұрыптаңдар.

73. Шеберханада шығарылған түрлі тетік бөлшектер мен олардың бағасы берілген. Осы мәліметтерді а) бағалардың өсуі және ә) тетік бөлшек атауларын алфавиттік реті бойынша сұрыптаңдар.

74. Оқушылардың аты-жөні және олардың телефон нөмірлері көрсетілген екіөлшемді жиым берілген. Оқушының фамилиясы бойынша оның телефон нөмірін табыңдар.

75. Екі матрица берілген. Олардың көбейтіндісін табыңдар.

76. Екіөлшемді жиымның элементтері сиқырлы квадрат (сиқырлы квадратта барлық вертикаль, горизонталь және екі диагональ бойынша сандардың қосындысы бірдей болады) құрайтындығын тексеретін программа құрыңдар.

77. Матрицаның элементтері қосалқы диагональ бойынша симметрия құра отырып орын алмастыратын программа құрыңдар .

78. Екіөлшемді k жиымының бағаналарын циклді түрде, оның i-ші бағанасын $i + 1$ бағанасымен алмастыра отырып, соңғы бағана бірінші болып орналасатындай деңгейге жеткізетін программа құрыңдар.

79. Екіөлшемді A жиымының нөлге тең элементтері жоқ жолдарының оң элементтерінің қосындысын есептеп шығаратын программа құрыңдар.

80. Квадрат пішінді кестенің ең кіші элементін анықтап, соған сәйкес жол мен бағана элементтерінің орындарын алмастырыңдар.

81. Бүтін сандардың екіөлшемді жиымы берілген. Сол жиымның ең кіші элементі орналасқан жолы мен бағанасын жойыңдар.

82. Тікбұрышты кестенің екінші қатарынан бастап, ондағы әрбір жолдың ең кіші элементін алдыңғы жолдың ең үлкен элементімен алмастырыңдар.

83. Бүтін сандардың 10x12 өлшемді матрицасы берілген. Оның барлық ершік нүктелерінің индекстерін басып шығарыңдар. (Ершік нүкте деп өзінің жолында ең кіші, бірақ бағанасында ең үлкен немесе, керісінше, өзінің жолында ең үлкен, бірақ өз бағанасында ең кіші болатын элементті айтады).

7. СӨЗ ТІРКЕСТЕРІН ӨҢДЕУ

PASCAL тілінде сөз тіркестерін өңдеу кезінде қолданылатын арнайы тип – string бар. Ал Си тілінде мұндай арнайы тип жоқ. Сөз тіркестері char типті бір өлшемді жиым ретінде қарастырылады, яғни сөз тіркесі – нөлдік байтпен аяқталатын char типті бір өлшемді жиым. Нөлдік байт – барлық биттері де нөлге тең байт, ол ´ \0 ´ символдық константасымен анықталады (тіркес соңы белгісі немесе нөл-терминатор). Сондықтан егер тіркесте k символ болса, онда жиымды сипаттауда k+1 элемент көрсетілуі тиіс.

Мысалы, **`char a[7]`** деген сипаттау тіркестің 6 символдан тұратынын, ал соңғы байт нөлдік екенін білдіреді. Си тіліндегі тіркестік (жолдық) константа – қос тырнақшаға алынған символдар жиыны. Мысалы, "Берілген тапсырма" тіркесі, оның соңына нөлдік байтты компилятор автоматты түрде өзі жазады.

Айнымалы мәні болатын сөз тіркесін сипаттау кезінде бірден көрсетуге болады, мысалы,

```
char S1[10]="123456789", S2[]="Болат";
```

Соңғы сөз ұзындығы тіркестің символдары санымен анықталады.

Символдар тіркесін пернелерден енгізу үшін екі стандартты функция – **`scanf()`** немесе **`gets()`** қолданылады, ал олардың прототиптері **`stdio.h`** тақырыптық файлында болады.

Символдық таңбаларды енгізу/шығару

Символдарды біртіндеп енгізу/шығару үшін **`printf()`** және **`scanf()`** функцияларының **`%c`** форматы қолданылады.

`getch()` – параметрсіз функция, басылған перненің кодын (int) береді, экранға ешқандай символ шығармайды.

`getchar()` – параметрсіз функция. Пернеден символдарды бір-бірлеп енгізеді. Сөз тіркесі <**Enter**> пернесі басылғанша енгізіле береді, оған дейін оны өзгертуге де болады.

`putch(c)` – бір символды (c – символдық айнымалы немесе константа), яғни бір таңбаны ғана экранға шығарады.

`putchar(c)` – бұл да тек бір таңбаны экранға шығарады.

Бұлар conio.h тақырып файлы бойынша жұмыс істейді.

Мысалы, латын алфавиті әріптерін экранға шығару программасы төмендегідей болады:

```
#include <conio.h>
#include <stdio.h>
void main()
{
 char z;
 clrscr();
```

```
 for(z='A';z<='Z';z++)
    putchar(z);
 getch();
}
```

Нәтижесі:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Ал енді осы символдарды ASCII-кодтарымен бірге шығаратын мына программаны көрейік.

```
   /* латын алфавиті*/
#include <conio.h>
#include <stdio.h>
void main()
{
 char z;
 clrscr();
 for(z='A';z<='Z';z++)
 {
   if (z=='K' || z=='U') printf("\n");
   printf(" %c-%d ",z,z);
 }
 getch();
 }
```

Программа жұмысы нәтижесі:

```
A-65 B-66 C-67 D-68 E-69 F-70 G-71 H-72 I-73 J-74
K-75 L-76 M-77 N-78 O-79 P-80 Q-81 R-82 S-83 T-84
U-85 V-86 W-87 X-88 Y-89 Z-90
```

Келесі программа 0 мен 9 арасындағы цифрлық символдарды және олардың ASCII кодтарын басып шығарады:

```
#include <conio.h>
#include <stdio.h>
void main()
{
 char z;
 clrscr();
 for(z='0';z<='9';z++)
  {
   if (z=='0' || z=='5') printf("\n");
   зкштеа(Э %с-%в Эбябя)ж
  }
 getch();
 }
```

Жұмыс нәтижесі:

```
0 – 48  1 – 49  2 – 50  3 – 51  4 – 52
5 - 53  6 – 54  7 – 55  8 – 56  9 – 57
```

Символдық тіркестер

Символдық жолдарды немесе тіркестерді бірнеше тәсілмен өңдеуге болады, олардың негізгілері:

1. Тіркестік константаларды қолдану;
2. Char типті жиымды қолдану;
3. Char типіне сілтейтін нұсқауыштарды пайдалану;
4. Символдық тіркестерден тұратын жиымдарды қолдану.

Сөз тіркестері немесе тіркестік (жолдық) константа қостырнақшаға алынып жазылады. Тырнақшаға алынатын символдар тізбегінің ең соңына автоматты түрде ‘\0’ символы жазылады. Компилятор жолдық символдарды компьютер жадына жазғанда, олардың көлемін анықтау үшін сол символдар санын есептейді. Символдық константа осы сөз тіркесі жазылған жады аймағына сілтейтін нұсқауыш болып табылады. Символдық тіркестер жиымын (массивін) беру кезінде компилятор компьютер жадының қажетті көлемін анықтау үшін жиымды сипаттағанда, оны тіркестік константа арқылы инициалдауға болады. Мысалы:

```
char c[]= "Атырау, Алтай - жеріміз";
```

Әдеттегі жиым қолданылатын жағдайлар сияқты бұл жиым аты **c** осы жиымның 1-элементіне сілтейтін нұсқауыш болып табылады.

```
c ==&c[0];
*c ==’0’;
*(c+1) == c[1] == ‘n’;
```

Сөз тіркестерін анықтау үшін нұсқауыштарды мынадай түрде сипаттауға болады:

```
char *c1 = "\n студенттер саны";
```

осы сипаттауға эквивалентті болып келесі сипаттау есептеледі:

```
static char c1[]="\n студенттер саны";
```

Осы қарастырылған екі сипаттау да **c1** тіркесінің нұсқауыш екенін білдіреді. Компьютер жадының қажетті көлемін айқын көрсетуге де болады. Мысалы, сыртқы сипаттауда келесі жолдың мынадай түрде жазылғаны көрсетілген.

```
char c[25]="Білім - өмір шырағы";
```

Элементтердің саны жолдың ұзындығынан бір символ артық болуы керек, яғни оның ең соңында ‘\0’ символы болуы тиіс.

Статикалық немесе сыртқы жиымдағы бұрынғы қарастырылған әдеттегі жиымдар оларды қолдану кезінде автоматты түрде 0-мен инициалданған болатын. Ал сөз тіркестерін пайдалану кезінде де

статистикалық немесе сыртқы жиымдар солар тәрізді 0 символымен инициалданады.

Келесі мысалды қарастыралық:

```
#include <stdio.h>
#include <string.h>
main ()
{
 char msg[30];
 strcpy(msg, "Сәлем, Азат!");
 puts(msg);
}
```

Мұндағы msg сөзінен соң тұрған [30] саны компиляторға 29 символ үшін, яғни char типіндегі 29 айнымалыдан тұратын жиым үшін жады бөлуді қамтамасыз етеді (30-орын нөлдік символмен – \0 толтырылады). msg айнымалысының символдық мәні жоқ; ол тек char типіндегі 29 айнымалының алғашқысының адресін (компьютер жадындағы белгілі бір орын адресі) сақтайды.

Компилятор strcpy(msg, "Сәлем, Азат!") операторын кездестіргенде, екі түрлі әрекет орындайды:

- "Сәлем, Азат!" тіркесі соңына (\0) символын (ASCII коды 0) қосады.
- strcpy функциясын орындап, msg айнымалысы нұсқап тұрған жады аймағына сол сөз тіркесі символдарын біртіндеп көшіреді. Ол тіркесті көшіруді сөз соңындағы нөлдік символдан кейін барып аяқтайды.

puts(msg) функциясын орындағанда, оған msg мәні, яғни тіркес құрамындағы бірінші символ адресі беріледі. Одан кейін puts сол символдың нөлдік символ емес екенін анықтап, ары қарай адреске бірді қосып, келесі символы оқиды, т.с.с. тіркес соңына дейін жетеді. Нөлдік символға жеткен соң, puts жұмысты аяқтайды;

Осындай тәсіл тіркес ұзындығына шек қоймай, нөлдік символға дейінгі символдарды біртіндеп оқуды жүзеге асырады.

Символға нұсқауышты пайдалану

Екінші тәсіл – символдарға нұсқауыш жасау. Программаны келесі түрге келтірейік:

```
#include <stdio.h>
#include <string.h>
main()
{
    char *msg;
    msg = "Сәлем, Азат!";
    puts(msg);
}
```

msg алдындағы жұлдызша (*) компиляторға оның символға нұсқауыш екенін білдіреді, яғни msg белгілі бір символ адресін сақтай алатын айнымалы. Бірақ мұнда компилятор символдар үшін ешқандай орын бөлмейді және msg да ешқандай мәнге ие болмайды.

Компилятор strcpy(msg, "Сәлем, Азат!") операторын кездестіргенде, ол тағы екі түрлі әрекет орындайды:

- объектілік код файлы ішіндегі бір орынға соңына (\0) символы қосылған "Сәлем, Азат!" тіркесін (ASCII коды 0) жазып қояды.
- сол тіркестің алғашқы символы адресін msg айнымалысына меншіктейді.

strcpy функциясын орындалып, puts(msg) командасы бұрынғыша нөлдік символға дейінгі мәліметті көшіреді.

Енді *символдық тіркестерден тұратын жиымдарды* қарастыралық. Бұл жиымдардың әрбір жолы символдық жиым болып табылады. Мысалы, статикалық жиымның сипатталуы келесідей түрде жазылуы мүмкін:

```
static char
       *m[4]={"регистр","жады","курсор","элемент"};
```

бұл жиым символдық тіркестерге сілтейтін 4 нұсқауыш болып табылады. Сонымен, символдық тіркестер жиымдар болып табылатын болса, онда осы жиымдарға сілтейтін 4 нұсқауыш қарастырылады. 1-жолға сілтейтін 1-нұсқауыш болып **m[0]** есептеледі, **m[1]–** 2-жолға сілтейтін 2-нұсқауыш болып табылады. Сонымен, әрбір нұсқауыш соған сәйкес жолдың немесе қатардың ең бірінші символына сілтейді.

```
*m[0]=='р'; *m[1]=='ж'; *m[2]=='к'; *m[3]=='э';
```

Тіркестерден құрылған жиымдарды сипаттағанда, символдық тіркестер көлемін көрсетуге де болады және бұл сипаттауда тіркестердің ұзындығын келесідей түрде көрсетуге болады:

```
static char m[10];
```

Символдар тіркестерін енгізу/шығару үшін **printf()** және **scanf()** функцияларының %s форматы қолданылады.

Тағы мысалдар келтірейік.

```
/* символдық тіркесті шығару */
#include <conio.h>
#include <stdio.h>
void main()
{char b[]="Сәзам, ашыл!";
 clrscr();
 printf("%s",b);
 getch();
}
```

Мұндағы b жиымының ұзындығы 13 символ, яғни сөздер ұзындығынан 1-ге артық.

Енді бір сөйлем енгізіп, соның соңғы сөзін экранға шығарайық.

```
#include <conio.h>
#include <stdio.h>
void main()
{
 char s,ss; // s – енгізілетін символ
  // ss – алдыңғы символ
 char a[80];  // сөз массиві
 int i,k;    // k – сөз ені
 clrscr();
 printf("Соңында нүкте бар сөйлем енгізіңдер:\n");
for(i='0',s=' ',k=0;i<=79;i++)
     {
      ss=s; s=getchar();
      if (s==' ') continue;
      if (s=='.') break;
      if (ss==' ') k=0;
      a[k]=s; k++;
     }
//нүктеден кейін шығу не тіркес біткесін шығу
if (i==80 || k==0)
  printf("сөйлем дұрыс емес \n");
else
  {
   a[k]='\0';   // жол соңы
   printf("ең соңғы сөз: %s",a);
  }
 getch();
}
```

мұнда символдар біртіндеп getchar() функциясы арқылы енгізіледі. Егер бос орын енгізілсе continue операторы келесі қадамға көшіреді. Нүкте енгізілсе цикл тоқталады, бірақ алыңғы k символда соңғы сөз сақталады. Егер символ нүкте де, бос орын да емес болса, онда алдыңғы символ қарастырылады. Егер ол бос орын болса, онда келесі сөз енгізіліп, k нөлге тең болады. Циклден шығу нүкте арқылы немесе 80 символ енгізілген соң орындалады.

Келесі мысалда сөз тіркесінің ұзындығы екі тәсілмен анықталады.

```
#include <conio.h>
#include <stdio.h>
#include <string.h>
```

```
void main()
{
 char st[80];
 int i;
 clrscr();
 puts("Сөз тіркесін енгізіп, Enter басыңыз:");
 gets(st);
 i=0;
 while (st[i++])
       ;
 printf("Енгізілген тіркес ұзындығы: %i\n",i-1);

 puts("Сөз тіркесін енгізіп, Enter басыңыз:");
 gets(st);
 printf("Енгізілген тіркес ұзындығы: %i\n",
         strlen(st));
 getch();
 }
```

Енді бір сөз тіркесін енгізіп, оның ішінде "т" символының неше рет кездесетінін табайық.

```
#include <conio.h>
#include <stdio.h>
#include <string.h>
void main()
{
   char str[80];
   int s=0;
   clrscr();
   puts("Сөз тіркесін (сөйлем) енгізіңіз:");
   gets(str);
   for (int i=0; i<strlen(str); i++)
      if (str[i] == 'т')  s+=1;
   printf("'т' символы %i рет кездеседі\n",s);
   getch();
 }
```

Ендігі мысалда енгізілген сөздің палиндром (алды-артынан оқығанда, мәні бірдей) екенін анықтайық.

```
#include <conio.h>
#include <stdio.h>
#include <string.h>
void main()
{
```

```
char str[80];
int k,s=0;
clrscr();
puts("Бір сөз (палиндром) енгізіңіз:");
gets(str);
k=strlen(str);
for (int i=0; i<k/2; i++)
   if (str[i] == str[k-i-1])  s+=1;
if (s==k/2)
puts("Сөз - палиндром.");
else puts("Сөз - палиндром емес.");
getch();
}
```

Сөз тіркестерін енгізу функциялары scanf(), gets(str).

Scanf() функциясы тіркестік айнымалылар мәнін %s форматымен енгізеді, бірақ ол тіркесті тек бірінші босорын таңбасына дейін ғана енгізе алады.

Ал **gets(str)** функциясы арасында босорыны бар тіркестерді енгізеді, енгізу ENTER пернесімен аяқталады.

Екі функция да тіркес соңына нөлдік байт қосып жазады. Оның үстіне тіркес –символдық жиым болып, ал жиым аты – оның компьютер жадындағы алғашқы адресіне сілтеме болғандықтан, тіркестік айнымалы атының алдына «&» символы жазылмайды.

Сөз тіркестерін шығару функциялары
cprintf(),puts(),cputs()

printf() – экранға формат арқылы сөз тіркесін шығарады;

cprintf() – экранға **printf()** сияқты формат арқылы сөз тіркесін шығарады, тек олардың түстерін **textcolor()** және **textback-ground()** функциялары арқылы өзгертуге мүмкіндік береді;

puts(str) – экранға сөз тіркесін шығарып, курсорды бірден келесі жолдың басына алып барады, мұндағы str – тіркестік константа немесе тіркестік айнымалы. Бұлар stdio.h тақырып файлы бойынша жұмыс істейді.

Екі функция да символдық жиымды нөлдік байтқа дейін шығарады. printf() функциясы символ тіркесі шығарылған соң, курсорды келесі жолға көшірмейді, ол үшін арнайы формат (\n) жазылуы тиіс. Ал puts( ) функциясы символдар шығарылған соң, автоматты түрде курсорды келесі жол басына көшіреді.

```
//puts функциясын пайдалану мысалы
#include <stdio.h>
#include <conio.h>
```

```
main()
{ char str1[]= "abc";
  char str2[]= "def\nghi\n";
  char str3[]= "jkl";
  puts(str1);
  puts(str2);
  puts(str3);
}
```

Нәтижесі:

```
abc
def
ghi
jkl
```

cputs(str) – экранға сөз тіркестерін шығарып, олардың түстерін **textcolor()** және **textbackground()** функциялары арқылы өзгертуге мүмкіндік береді, **conio.h** тақырып файлы бойынша жұмыс істейді.

Сөз тіркестерімен орындалатын басқа операциялар да стандартты функциялар арқылы атқарылады. Ол функциялар жұмыс істеуі үшін **string.h** тақырыптық файлы қажет.

Жалпы сөз тіркестеріне қолдану үшін **stdlib.h** немесе **string.h** тақырыптық файлдары қолданылады.

Сөз тіркестерімен жұмыс істейтін функциялар

1) **strlen(str)** функциясы str сөз тіркесіндегі символдар санын (соңғы нөлді есепке алмайды), яғни жолдың ұзындығын анықтайды, оның типі int, тақырыптық файлы <string.h>.

Мысалы. Бірнеше сөз тіркестерінің ұзындығын анықтайтын программа құру керек.

```
// strlen(str) функциясын пайдалану
#include <conio.h>
#include <stdio.h>
#include <string.h>
main ()
{
static char t[]="Студенттер жайлы хабарлама.";
clrscr();
printf("%d\n",strlen(t));
printf("%d\n",strlen("Студенттер жайлы
                      хабарлама."));
```

```
printf(”%d\n”,strlen(”аль-Фараби ат.КазҰУ”));
printf(“%d\n”,strlen(“”));
getch();
}
```
Мұның нәтижесі:
27
27
19
0

2) strcat(stroka1,stroka2) функциясы қатарларды біріктіру үшін қолданылады. Ол **stroka1** және **stroka2** тіркестерін біріктіріп, нәтижені **stroka1** айнымалысына меншіктейді, **stroka2** тіркесінің мәні өзгермейді
Мысалы:
```
// strcat(str1, str2) функциясын пайдалану
#include <conio.h>
#include <stdio.h>
#include <string.h>
main ()
{
char str1[50]="Си тілін оқимыз, ";
char str2[]="жақында емтихан тапсырамыз.";
clrscr();
printf("%s\n",strcat(str1,str2));
puts(str1);   // қатарды экранға шығару
puts(strcat("Егер жақсы оқысақ, ",str2));
getch();
}
```
Мұның нәтижесі:
```
Си тілін оқимыз, жақында емтихан тапсырамыз.
Си тілін оқимыз, жақында емтихан тапсырамыз.
Егер жақсы оқысақ, жақында емтихан тапсырамыз.
```

Келесі мысалда студенттің аты пернелерден енгізіліп, ол екінші тіркеспен біріктіріледі.
```
// strcat(str1, str2) функцияларын пайдалану
#include <conio.h>
#include <stdio.h>
#include <string.h>
main ()
{
```

```
char name[80];
char stud[]= " - КазҰУ студенті";
clrscr();
puts("атын енгіз:");
gets(name);
strcat(name,stud);
puts(name);
getch();
}
```

3) **strcmp(stroka1,stroka2)** функциясы екі сөз тіркесін салыстыру үшін қолданылады. Егер олар бірдей болса, функцияның мәні 0-ге тең болады, әйтпесе ол екі тіркестің айырмасын береді. Егер **stroka1<stroka2** болса, нәтиже 0-ден кіші, ал **stroka1> stroka2** болса, нәтиже 0-ден артық болады. Көбінесе бұл тәсіл екі тіркестің бірдей еместігін анықтау үшін ғана қолданылады.

Мысалы:

```
main ()
{
  printf("%d\n",strcmp("Сәлем","Сәлем"));
  printf("%d\n",strcmp("Azat","Izat"));
  printf("%d\n",strcmp("Абайда","Абайла"));
 getch();
}
```

Мұның нәтижесі:

```
0
-8
-7
```

Алғашқы екі сөз бірдей, нәтижесі – 0, келесі екі сөздің алғашқы әрпі әр түрлі, олардың ASCII-кодтарының айырмасы – -8 (А - 65, І - 73), ал 3-жолы -7 (д – 164, л – 171, олардың кодтарының айырмасы 164-171=-7).

```
// strcmp(str1, str2) функциясын пайдалану
#include <conio.h>
#include <stdio.h>
#include <string.h>
#define NAME "Ритчи"
main ()
{
 char f[20];
 puts("Си тілінің авторы кім:");
 gets(f);
```

```
 while(strcmp(f,NAME)!=0)
  {puts("басқа кім болуы мүмкін:");
   gets(f);
  }
puts("Жауап дұрыс!");
getch();
}
```

Нәтижесі:

```
Си тілінің авторы кім:
Керниган
басқа кім болуы мүмкін:
Ритчи
Жауап дұрыс!
```

4)strcpy(str1,str2) функциясы сөз тіркесінің көшірмесін алу үшін қолданылады, мұнда str2 айнымалысындағы сөз тіркесі str1 айнымалысына көшіріледі. Мысалы:

```
// strcpy(str1, str2) функциясын пайдалану
#include <conio.h>
#include <stdio.h>
#include <string.h>
main ()
{
 char str1[21];
 strcpy(str1,"Хал қалай, Азат?");
 puts(str1);
 strcpy(str1,"Тамаша!");
 puts(str1);
 getch();
}
```

Нәтижесі:

```
Хал қалай, Азат?
Тамаша!
2-мысал:
// strcpy(str1, str2) функциясын пайдалану
#include <conio.h>
#include <stdio.h>
#include <string.h>
#define stroka "көшіру функциясы"
main ()
{
 char *ptr=stroka;
```

```
 char res[25];
 clrscr();
 puts(ptr);
 puts(res);
 strcpy(res,ptr);
 puts(ptr);
 puts(res);
 getch();
}
```

Нәтижесі:

```
көшіру функциясы

көшіру функциясы
көшіру функциясы
```

Мұнда **ptr** айнымалысы **көшіру функциясы** сөзін береді, **res** айнымалысы бос жол береді, ал келесі жолы екеуі де **көшіру функциясы** сөзін береді.

5) **strstr(str1,str2)** функциясы 2-ші көрсетілген жолды 1-ші жолдың ішінен іздейді.

6) **strset(str,ch)** функциясы берілген қатардағы барлық символдарды көрсетілген символға (`char ch`) ауыстырады.

7) **strtod(str1,str2)** функциясы берілген қатарды **double** типті санға ауыстырады.

8) **strchr(str,c)** функциясы берілген қатардағы коды көрсетілген символдың позициясын анықтайды.

9) **strrev(str)** функциясы берілген қатардың барлық символдарын керісінше бейнелейді.

10) **strpbrk(str1,str2)** функциясы 2-ші қатардың кез келген символын 1-ші қатардан іздейді.

Бақылау сұрақтары

1. *Тіркестік айнымалылардың сипатталу тәсілдері қандай?*
2. *Тіркестік айнымалы қандай идентификатормен (атаумен) және қалай анықталады?*
3. *Бір тіркестік айнымалыға немесе тұрақтыға қанша символ жазуға болады?*
4. *Тіркестік айнымалының ұзындығы қалай анықталады?*
5. *Тіркестік өрнектер дегеніміз не?*
6. *Тіркестік айнымалылар мен тұрақтыларға қандай амалдар қолданылады?*
7. *Тіркестің ішкі символдарын қалай бөліп алуға болады?*
8. *Си тілінде сөз тіркестерін өңдейтін қандай функциялар бар? Оларды қалай пайдаланады және олар қалай жазылады**?***

ТАПСЫРМАЛАР

1. Бір топтағы фамилиялары бірдей студенттерді табыңдар.

2. Топ студенттері фамилиялары мен аттарының бірінші әрпін шығарыңдар.

3. Әрбір сөзден кейін бір бос орын қалдырылған сөйлемдер берілген. Құрамында дауысты әріптер ең көп кездесетін сөзді табыңдар.

4. Әрбір сөзден кейін бір бос орын қалдырылған сөйлемдер берілген. Құрамында берілген әріптен басталатын сөздердің санын анықтаңдар.

5. Әрбір сөзден кейін бір бос орын қалдырылған сөйлем берілген. Сөйлемдегі ең ұзын сөзді табыңдар.

6. Әрбір сөзден кейін бір бос орын қалдырылған сөйлем берілген. Сөйлем ішіндегі жақшалардың дұрыс қойылғанын тексеріңдер.

7. Әрбір сөзден кейін бір бос орын қалдырылған сөйлемдер берілген. Сөйлем ішіндегі жақшаға алынған мәтіндерді өшіріңдер.

8. Берілген сөз тіркесіндегі әрбір нүктені көп нүктемен (яғни үш нүктемен) алмастырыңдар.

9. Алдыңғы есеп шартындағы қатар келген нүктелердің әрбір тобын бір нүктемен алмастырыңдар.

10. Дүкендегі кассир көмекшісіне арналған программа құрыңдар. Программа тауардың бағасын, мөлшерін, сатып алынған тауарлар бағасының қосындысын есептеп, сатып алушының берген ақшасының мөлшерін сұрап, оған қайтарылатын соманы да анықтайды.

11. Әрбір сөзден кейін бір бос орын қалдырылған сөйлем берілген. Барлық сөздерді керісінше жазып шығыңдар.

12. Пернетақтадан енгізілген сөздердегі әріптердің санын есептейтін программа жазыңдар. Тапсырманы do … while циклында орындаңдар.

13. Натурал n саны және S_1, S_2, ... S_n символдары берілген. Осы символдардың арасында неше рет + символы кездесетінін анықтаңдар.

14. Натурал n саны және S_1, S_2, ... S_n символдары берілген. Осы символдардың арасында * символдарының санын есептеңдер.

15. Натурал n саны және S_1, S_2, ... S_n символдары берілген. Осы символдардың арасында қандай символдар көп: + не * символы ма?

16. Сөз тіркесі енгізілгеннен кейін оның құрамында бір символ қалғанша, тіркесті цикл сайын бір символға қысқарта отырып, сөздердің барлық нұсқаларын экранға шығаратын программа жазыңдар.

17. Енгізілген сөз тіркесіндегі сөздердің санын анықтайтын программа жазыңдар. Бір сөз екіншісінен бір бос орын арқылы айырылады деп санау керек.

18. Мәтіні пернетақтадан енгізілген телеграмманың бағасын есептейтін программа жазыңдар.

19. Берілген сөзде бірінші және соңғы әріптердің қайсысы көп кездесетінін анықтайтын программа жазыңдар.

20. “*а*” әрпімен аяқталатын атау септігіндегі зат есім берілген. Осы сөзді септеп, басып шығарыңдар.

21. Берілген сөздің жұп нөмірлі орындарында қанша “*о*” әрпі бар екендігін анықтайтын программа жазыңдар.

22. Студенттің фамилиясы, есімі және әкесінің аты бос орындармен бөлініп берілген. Студенттің аты-жөнінің инициалдарын (алғашқы әріптерін) басып шығаратын программа жазыңдар.

23. Сөз тіркесіндегі *а* әрпін өшіретін программа жазыңдар.

24. Мәтіндегі соңғы әріппен бірдей әріптерді жоятын программа жазыңдар.

25. *Z, X* сөздері берілген. *Z* сөзінде кездесетін барлық әріптерді *X* сөзінен өшіріп тастайты программа жазыңдар.

26. Берілген сөздегі әр түрлі әріптердің санын есептейтін программа жазыңдар.

27. Сөздердің әрбір үшінші әрпін жоятын программа жазыңдар.

28. Берілген мәтіндегі “*Айна*” сөзін “*Асыл*” сөзіне өзгертетін программа жазыңдар.

29. Пернетақтадан енгізілген символды жазылған сөз тіркесінен өшіретін программа жазыңдар. Өшіру процесін жеке функция етіп қарастырыңдар.

30. Берілген мәтіндегі кездесетін “*а*” әрпін “*о*” әрпіне ауыстырыңдар.

31. Енгізілген сөз тіркесін керіге айналдыратын, яғни символдарды кері тәртіпте орналастыратын программа құрыңдар.

32. Енгізілген сөздің палиндром болатынын/болмайтынын анықтайтын программа құрыңдар.

33. Пернетақтадан енгізілген символдарды ASCII кестесінде нөмірлерінің өсу реті бойынша сұрыптайтын программа жазыңдар.

34. Бос орындармен бөлініп жазылған үш сөзден тұратын сөйлемдегі ең қысқа сөздің ұзындығын есептеп шығаратын программа жазыңдар.

35. Экранға жылжымалы жолды шығаратын программа жазыңдар.

36. Сөйлемдегі жүйелі жақшаға алынған мәтіндерді және жақшаның өзін өшіретін программа құрыңдар.

37. Ұзындығы 25 символдан артпайтын сөз тіркесін алып, одан мүмкіндігінше бірнеше жаңа сөз құрастырыңдар.

38. Сөз тіркесіндегі кездесетін “а” әрпін “ә” әрпімен ауыстыратын программа құру керек.

39. Пернелерден енгізілген сөздің ұзындығын анықтайтын программа жазу қажет. Программаны while do циклі арқылы ұйымдастырып, программа жұмысын аяқтауды ‘999’ тіркесін енгізу арқылы жүзеге асыру керек.

40. Берілген тіркес құрамындағы сөздер бос орын арқылы бөлініп жазылған деп есептеп, олардың ішіндегі ең ұзын сөзді табу керек.

41. Берілген мәтінде өзің қалап алған сөз қанша рет кездесетінін анықтау керек.

42. Берілген мәтін сөздерінде ең көп кездесетін әріпті табу қажет.

43. Берілген мәтін сөздерінің ең жиі ұшырасатын алғашқы әрпін анықтау керек.

44. Берілген сөз тіркесіндегі “а” әрпінің санын анықтайтын программа құрыңдар.

45. Берілген мәтін сөздері арасындағы бос орынды үтірмен алмас-тырыңдар.

46. Берілген мәтіннің сөздері бір-бірінен бірнеше бос орын таңбасымен бөлініп жазылған, сол сөздердің арасына тек бір бос орын таңбасын қалдырып, қайта жазып шығыңдар.

47. Берілген мәтіннің неше сөзден тұратынын табу керек (сөздер арасындағы бос орын тұратынын пайдаланыңдар).

48. Берілген мәтінде “ас” тіркесі қанша рет кездеседі?

49. Берілген мәтіндегі “а” әрпімен аяқталатын сөздерді экранға шығару керек.

50. Енгізілген сөз тіркесі сан болатынын немесе болмайтынын анықтау қажет

51. Енгізілген үш сөздің ішіндегі ең қысқасының неше символдан тұратынын анықтаңдар.

52. Енгізілген мәтіннің жұп орында тұрған сөздерінің ішінде қанша 'е' әрпі кездесетінін табыңдар.

53. Енгізілген сөзде қандай әріптер қанша рет кездесетінін анықтау керек.

54. Сөзбен енгізілген сан аттарын цифр түрінде жазып шығатын программа құрыңдар.

55. Сөз тіркесі түрінде санмен енгізілген мәліметті цифрсыз жазылған сөз тіркесіне айналдырыңдар, мысалы, '56' – 'елу алты', '-125' – 'минус бір жүз жиырма бес', т.с.с.

Есептер

1 вариант

1. Құрамына сандар кіретін сөз тіркесінің ұзындығын – L анықтап, егер L жұп сан болса, онда тіркестегі барлық екілік сандарды өшіріп тастаңдар.
2. Берілген сөз тіркесінің дәл ортасында тұрған сөзді керісінше жазып шығыңдар.

2 вариант

1. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L жұп сан болса, онда тіркестегі алғашқы сөзді өшіріңдер, ал тақ сан болса, соңғы сөзді өшіру керек.
2. Берілген сөз тіркесінде палиндром сөз бар екенін анықтап, ол жайлы мәлімет беру керек.

3 вариант

1. Берілген сөз тіркесінің ішінде ДОС сөзіне кіретін символдардың нешеу екенін анықтаңдар.
2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L >10 болса, соңғы сөзді өшіріп тастаңдар.

4 вариант

1. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L тақ сан болса, онда тіркестің дәл ортасындағы символды анықтау керек, ал жұп болса – тіркес ортасындағы екі символды анықтау керек.

2. Берілген сөз тіркесінің ішіндегі барлық ! белгісін ? белгісіне алмастырыңдар.

5 вариант

1. Берілген сөз тіркесінің ішіндегі бос орын таңбаларын астын сызу (_) таңбасына алмастырыңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L >10 болса, соңғы сөзді өшіріп тастаңдар.

6 вариант

1. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L 3-ке қалдықсыз бөлінетін болса, тіркестегі екінші сөзді өшіріп тастаңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L 5-ке қалдықсыз бөлінетін болса, тіркестегі барлық жақша түрлерінің санын анықтаңдар.

7 вариант

1. Екі сөйлемнен тұратын сөз тіркесінің сөйлемдерінің орындарын алмастырыңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L 3-ке қалдықсыз бөлінетін болса, тіркестегі екінші сөзді өшіріп тастаңдар.

8 вариант

1. Берілген сөз тіркесінің алғашқы нүктесіне дейінгі символдарды “с” әрпіне алмастырыңдар.
2. Берілген сөз тіркесіндегі бірінші және соңғы сөзді керісінше жазып шығыңдар.

9 вариант

1. Берілген сөз тіркесінің алғашқы сөзі мен екінші сөзін керісінше жазып шығыңдар.

2. Бірнеше сөйлемнен тұратын сөз тіркесіндегі екінші сөйлем ішіндегі “Е” әрпінің санын анықтаңдар.

10 вариант

1. Берілген сөз тіркесінде жақшалар бар. Алғашқы жақшалар ішіндегі сөзді анықтаңдар.
2. Берілген сөз тіркесіндегі ең ұзын сөздің енін тауып, оны керісnше жазып шығыңдар.

11 вариант

1. Сөз тіркесі берілген. Оның ішіндегі леп белгілерін нүктемен алмастырып, нүктелер санын анықтаңдар. Леп белгісі жоқ болса, ол туралы мәлімет беріңдер.

2. Құрамына сандар кіретін сөз тіркесінің ұзындығын – L анықтап, егер L жұп сан болса, онда тіркестегі барлық екілік сандарды өшіріп тастаңдар.

12 вариант

1. Сөз тіркесі берілген. Оның ішіндегі нүктелерді үш нүктемен алмастырып, нүктелер санын анықтаңдар, нүкте жоқ болса, ол жайлы мәлімет беріңдер.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L тақ сан болса, онда тіркестің соңғы сөйлемін анықтау керек, ал жұп болса – тіркес ортасындағы символды анықтау керек.

13 вариант

1. Сөз тіркесі берілген. Оның ішінде үтірлер бары белгілі. Алғашқы үтір мен соңғы үтірдің қай позицияда тұрғанын және оларды арасында неше символ бар екенін анықтаңдар.

2. Бірнеше сөйлемнен тұратын сөз тіркесіндегі соңғы сөйлем ішіндегі “А” әрпінің санын анықтаңдар.

14 вариант

1. Сөз тіркесі берілген. Оның ішінде нүктелер бары белгілі. Сол тіркесте неше нүкте бар екенін және бірінші нүктемен екінші нүкте арасында неше символ орналасқанын анықтау керек.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L жұп сан болса, онда тіркестегі алғашқы сөзді өшіріңдер, ал тақ сан болса, соңғы сөзді өшіру керек.

15 вариант

1. Сөз тіркесі берілген. Оның ішінде қатар орналасқан бірдей символдар бар екенін анықтау керек. Ондай символдар жоқ болса, ол туралы мәлімет беру керек.

2. Берілген сөз тіркесінің ішінде ДОС сөзіне кіретін символдардың нешеу екенін анықтаңдар.

16 вариант

1. Берілген сөз тіркесінде неше бос орын бар екенін анықтап, сол тіркестегі ең ұзын сөздің енін табу керек.

2. Берілген сөз тіркесінде “ба” сөзі неше рет кездесетінін анықтау керек, ондай тіркес жоқ болса, ол туралы мәлімет беріңдер.

17 вариант

1. Берілген сөз тіркесінде неше арифметикалық амалдар таңбасы бар екенін табыңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L тақ сан болса, онда тіркестің дәл ортасындағы символды анықтау керек, ал жұп болса – тіркес ортасындағы екі символды анықтау керек.

18 вариант

1. Берілген сөз тіркесінде неше нүкте бар екенін және нүктелер арасында неше символ орналасқандарын анықтау керек.

2. Берілген сөз тіркесінің ішіндегі барлық ! белгісін ? белгісіне алмастырыңдар.

19 вариант

1. Берілген сөз тіркесіндегі барлық "А" әріптерін алып тастаңдар да, "Е" әріптерінің орнына "Э" әріптерін жазып шығыңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L 5-ке қалдықсыз бөлінетін болса, тіркестегі барлық жақша түрлерінің санын анықтаңдар.

20 вариант

1. Берілген сөз тіркесіндегі екінші сөйлемді экранға шығарыңдар және оның неше символы бар екенін анықтаңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L 5-ке бөлінетін сан болса, онда тіркестің дәл ортасындағы символды анықтау керек, ал жұп болса – тіркес ортасындағы сөзді анықтау керек.

21 вариант

1. Берілген сөз тіркесіндегі (кириллица) бас әріптерді кіші әріптерге айналдырыңдар.

2. Берілген сөз тіркесінің ұзындығын – L анықтап, егер L 3-ке бөлінетін сан болса, онда тіркестің бірінші сөйлемін экранға шығарыңдар, әйтпесе тіркестің дәл ортасындағы символды анықтау керек.

22 вариант

1. Берілген сөз тіркесінің ішіндегі бос орын таңбаларын астын сызу (_) таңбасына алмастырыңдар.

2. Берілген сөз тіркесіндегі "Г" әріптерінің барлығын "G" латын әріптеріне айналдырыңдар.

23 вариант

1. Берілген сөз тіркесіндегі (кириллица) кіші әріптерді бас әріптерге айналдырыңдар.

2. Екі сөйлемнен тұратын сөз тіркесінің сөйлемдерінің орындарын алмастырыңдар.

8. ТҰТЫНУШЫ ФУНКЦИЯСЫН ПАЙДАЛАНУ

Си тілінде стандартты функциялармен қатар тұтынушы өзі құрастырған функцияларымен де жұмыс істеу мүмкіндігі жасалған. Ол функцияны алдын ала main функциясына дейін толық анықтау керек немесе оны алдын ала қысқаша сипаттап алып, main функциясынан кейін толық анықтауға болады. Сондықтан, тұтынушы функциялары декларациясы екі түрде: қысқаша сипаттау түрінде (main функциясына дейін) және анықтау түрінде (main функциясынан кейін немесе дейін) толық берілуі мүмкін.

Функцияны алдын ала сипаттау барысында программалық файлдың басында оның прототипі көрсетіледі, өйткені main функциясында оның айнымалылары типтері белгілі болуы тиіс. Ол келесі түрі жазылады:

```
<нәтиже_типі> <функция _ аты>(<тип> <айнымалы1>,
    <тип> <айнымалы2>, …<тип> <айнымалыN>);
```

Прототиптегі жай жақшаларға алынған айнымалылар идентификаторларын көрсетпесе де болады, өйткені тілдің компиляторы оларды өңдемейді.

Параметрлер тізімімен берілген fun функциясын сипаттаудың мысалы:

```
float fun(int, float, int, int);
```

Функция прототипі компиляторға программа мәтінінде негізгі программадан (main()) кейін оның толық анықтамасы келтірілетінін білдіреді.

Функцияның толық анықталуының жазылу түрі төмендегідей:

```
<нәтиже_типі> < функция _ аты >(параметрлер
                                  тізімі)
  {
    функция коды
  }
```

Нәтиже_типі *return* операторының көмегімен функцияның шақырылу нүктесіне қайтарылатын мәннің типін көрсетеді. Егер функция типі көрсетілмесе, қайтарылатын мән **int** типті деп саналады. Параметрлер тізімі үтірмен ажыратылған типтер мен параметрлер аттарының тізбегінен тұрады. Егер функцияның параметрлері болмаса да, жай жақшалар бәрі бір міндетті түрде көрсетілуі тиіс.

Return операторы берілген функциядан бірден шығып, нәтиженің оны шақырушы функцияға қайтарылуын қамтамасыз етеді, яғни бұл оператор функция жұмысы нәтижесін қайтару қызметін атқарады. Функция тұлғасында бірнеше *return* операторы болуы немесе бірде бір рет кездеспеуі де мүмкін екендігін айта кеткен жөн. Мұндай жағдайда

шақырушы функцияға қайту оның тұлғасындағы соңғы оператор орындалғаннан кейін барып жүзеге асырылады.

Екі бүтін санның ішіндегі кішісін анықтайтын функцияның мысалы:

```
int  mini(int x, int y)
{int t;
if (x<y) t=x;
else t=y;
return t;
}
```

Mini функциясын келесі түрде де жазуға болады:

```
mini(int x, int y)
{
return (x<y)? x:y;
}
```

Екі бүтін санның ішіндегі үлкенін анықтайтын функцияның мысалы:

```
int  maxi(int x, int y)
{
 if (x>y)
 return (x);
 else
 return (y);
}
```

Егер қайтарылатын мәннің типі көрсетілмесе, ол *int* типті болып есептеледі.

Си тіліндегі мәнді қайтаратын барлық функциялар өрнектердің оң жағында жазылуы тиіс, өйтпеген жағдайда ешқандай да нәтиже қайтарылмайды. Бірақ функция нәтижесінің адресі қайтарылатын жағдайда, ол сол жақта болуы керек.

Егер функция ешқандай мән қайтармайтын болса, онда ол **void** (бос) типті функция ретінде сипатталуы тиіс. Мысалы, дисплей экранына бір толық жолды шығару үшін келесі функцияны жазамыз:

```
void lin(char a)
{
int k;
for(k=0; k<80; k++)
printf("%c", a);
}
```

Егер функцияда ешқандай параметр болмаса, онда функцияны декларациялау барысында жай жақша ішіне **void** сөзін жазған дұрыс. Мысалы, негізгі функция тақырыбының көбінесе былай жазылуы мүмкін:

`void main(void)`.

Си тілінде әрбір функция – программаның жеке бөлігі, оны орындау үшін осы функцияны шақыру керек. Мысалы, *goto* операторы арқылы басқаруды кез келген функцияның тұлғасына беруге болмайды.

Функция келесі түрде шақырылады:

`<функция_аты>(аргументтер_тізімі);`

мұндағы аргументтер ретінде тұрақтыларды, айнымалыларды, өрнектерді (олардың мәндері функцияны шақырудан бұрын программада анықталады) қолдануға болады. Функцияны шақыру тізімінің аргументтері сол функцияның анықталу параметрлерінің тізімімен саны жағынан, орналасу ретімен, сәйкес параметрлері типтерімен толықтай үйлесуі тиіс. Аргументтер жоқ болған жағдайда да, функция атынан кейінгі жай жақшалардың болуы міндетті.

Айнымалылардың әрекет ету аймағы

Айнымалылардың әрекет ету аймағы – программаның ағымдағы әрекеттеріне қандай мәліметтердің қатынасуға болатынын анықтайтын қағида. Айнымалылардың үш типі болады: ауқымды, локальді (жергілікті) және формальды. Жергілікті айнымалылардың әрекет ету аймағы – олар сипатталған программа бөлігі, яғни олар сол функция тұлғасында ғана белгілі болып табылады. Программаның осы бөлігінен шыққан соң, жергілікті айнымалылар мен олардың мәндері жоғалады.

Формальды айнымалылар – тұтынушы функциясының тақырыбында көрсетілген параметрлер. Формальды параметрлер функция тұлғасында жергілікті айнымалылар сияқты қолданылады. Формальды параметрлердің әрекет ету аймағы – функция тұлғасы болып табылатын блок.

Ауқымды айнымалылар программадағы функциядан тыс сипатталады. Олар программаның кез келген жерінде қолданылуы мүмкін, сондықтан ауқымды айнымалыларды алдын ала сипаттау және бастапқы мәндерді меншіктеу керек. Ауқымды айнымалылардың әрекет ету аймағы – олар сипатталғаннан бастап, программа соңына дейін болып саналады.

Си тілінде әрбір айнымалы компьютер жадының келесі төрт класының біріне жатуы тиіс, олар – *автоматты* (auto), *сыртқы* (extern), *статикалық* (static), *регистрлік* (register). Айнымалы класын көрсету үшін оның типі спецификациясының алдына қажетті түйінді сөз (auto, extern, static, register) қоса жазылады. Мысалы, **register int a;**

Айнымалы үшін жады класы нақты көрсетілмеген жағдайда, ол auto класына жатқызылады.

Си тілінде функцияны шақыру барысында аргументтер өздерінің осы программадағы мәндеріне сәйкес әрекеттер атқарады, яғни функцияға аргументтердің нақты мәндері емес олардың көшірмелері жіберіледі.

Компьютер жадында (стекте) функцияның формальды параметрлері үшін орын бөлінеді және функцияны шақыру кезінде сол бөлінген орындарға нақты аргументтер мәндері орналастырылады. Содан кейін функция осы мәндерді пайдаланады, бірақ функциядан шыққан кезде олар жоғалып кетеді.

Қажет болған жағдайда, функцияны оған берілген аргументтерді өзгерту үшін қолдануға болады. Ол үшін шақырылатын функцияға аргумент ретінде айнымалы мәнінің орнына оның адресін жіберсе болғаны. Ал аргумент негізгі (оригинал) мәнді пайдалануы үшін "*" нұсқауыш операциясын қолдану қажет.

1- мысал. x және *y* мәндерінің орындарын ауыстыратын функцияның анықталуы:

```
void z1(int *x, int *y)
{ int t;
  t=*x;
  *x=*y;
  *y=t;
}
```

Осы функцияны пайдаланатын (шақыру) программа бөлігі:

```
int a=2, b=3;
void z1(int*, int*);
…
printf("\n a=%d, b=%d", a, b);
z1(&a, &b);
printf("\n a=%d, b=%d", a, b);
…
```

Мұнда функцияны шақыру кезінде, мәндер өзгеріске ұшырайды, яғни экранға төмендегі мәндер шығарылады:

```
   a=2, b=3
   a=3, b=2
```

2-мысал. unsigned char типті *r* параметрімен шақырылатын **rus** функциясы анықталған. Егер функцияның параметрі орыс алфавитінің әрпі болса, бірге тең бүтін мәнді, ал кері жағдайда 0 мәнін шығарады.

```
int rus (unsigned char r)
{if (r>='A' && c<=' ')
 return 1;
 else
return 0;
}
```

Функциялар өздерін өздері *рекурсивті* түрде шақыруы да мүмкін.

3-мысал: *n*!=1*2*3*…**n* санының факториалын есептейтін рекурсивті функцияны қарастырайық:

```
fac(int n)
{
  int b;
  if (n==1) return 1;
  b=fac(n-1)*n;
  return n;
}
```

Функцияны рекурсивті шақыру кезінде ол функцияның жаңа көшірмесі құрылмайды, бұл шақыру жергілікті айнымалылар мен параметрлердің жаңа көшірмелерін құрады. Рекурсивті функциядан шығудың дұрыс жолдарын қарастыру қажет, өйтпеген жағдайда мұндай функциямен жұмыс істеу барысында біраз уақыттан кейін ол "тоқтап қалуы" мүмкін.

Кез келген басқа объектілер сияқты функцияларда да нұсқауышты пайдалануға болады, мысалы, **type t, type z** параметрлі **type** типін қайтаратын функциядағы **p** нұсқауышын келесі түрде сипаттауға болады:

```
type (*p)(type1 t1, type2 t2);
```

Айтылғандарды тұжырымдай келе функцияда бір және екі өлшемді жиымдарды (массивтерді) пайдалануды қарастырайық.

Функцияға нақты параметр ретінде бір өлшемді жиымды жіберуге болады, ол үшін жиымның бастапқы (нөлінші) элементінің орнын көрсетсек жеткілікті.

Бір өлшемді жиымды пайдалану мысалы:

```
int min_index(int sp[], int ras)
{ int i, mindx, m;
mindx=0;m=sp[mindx];
for(i=0; i<ras; i++)
      if(sp[i]<m)
      {  m=sp[i]; mindx=i;
       };
return(mindx);
}
```

Бұл жерде Си тілінің артықшылығын айта кеткен жөн. Жоғарыда қарастырылған мысалдан көріп отырғанымыздай, компиляторға өңделіп отырған жиым мөлшерін анық көрсетудің қажеті жоқ. Функция тұлғасында жиымның бастапқы элементінің адресін көрсету жеткілікті. Бір өлшемді жиымды кездейсоқ бүтін сандармен толтыру үшін келесі түрдегі функцияны пайдалануға болады:

```
void init(int mas[],int ras);//функцияны сипаттау
{
  int k;
  for (k=0; k<ras; k++)
```

```
  mas[k]=rand();
}
```

Программаның қандай да бір бөлігінде жоғарыда аталған функцияны **init(spisok, dlina)** түрінде шақырсақ, бір өлшемді ұзындығы **dlina** болатын **spisok** жиымының элементтері кездейсоқ мәндерге ие болады. Енді **min_index** функциясын пайдаланатын программаны қарастырайық:

```
#include <stdio.h>
#include <conio.h>
#define dlina 150
main()
{int k, list[dlina];
 for (k=0; k<dlina; k++)
 list[k]=rand();
 k= min_index(list, dlina);
 printf("list[%3d]=%6d\n",k, list[k]);
}
```

Функцияның нақты параметрі ретінде екі немесе одан да көп өлшемді жиымдарды да пайдалануға болады.

Екі өлшемді жиымды пайдалану мысалы:

```
void minit(int matrix[][KO],int str);
{
 int i,j;
 for (i=0; i<str; i++)
 for (j=0; j<KO; j++)
     matrix[i][j]= rand();
}
```

Жоғарыда қарастырылған функцияда екі өлшемді жиымның бағаналарының саны тұрақты және ол ауқымды KO айнымалысымен анықталған.

*4- мысал. N*x*N* өлшемді бүтін сандар жиымын (50-ден көп емес) енгізіп, функция арқылы оның оң мәндерінің қосындысын табу керек.

```
#include <stdio.h>
#include <conio.h>
void summa(int,int a1[ ][50]);//функцияны сипаттау
void main(void)
{
 int a[50][50];
 int i,j,N;
 clrscr();
 puts("\n Жиым өлшемін N(<50) енгізіңіз \n");
```

```
 scanf(“%d”,&N);
 printf("\n Мәліметтерді енгізіңіз \n");
 for(i=0; i<N; i++)
 for(j=0; j<N; j++)
  {
   printf("\n a[%d][%d]=", i+1, j+1);
   scanf("%d", &a[i][j]);
  }
 summa(N,a);
}
void summa(int n,int a1[ ][50])//функцияны анықтау
{
  int i,j,s;
/* Жиымның оң элементтерінің қосындысын есептеу*/
for (s=0,i=0; i<n; i++)
{
  printf("\n");
  for (j=0;j<n;j++)
  if (a1[i][j]>0)
  s+=a1[i][j];
}
printf("\a Қосынды = %d, Press any key... ",s);
getch();
}
```

5- мысал. Берілген **s1** символдар тізбегінен k-сыншы символдан бастап ұзындығы **L** болатындай бағыныңқы **s2** тізбегін бөліп алатын **substr** функциясын қарастырайық:

```
substr(s1,s2,L,k,m)
char s1[],s2[]; int L,k,m;
{
  int i,j;
  if(L+k>=m)k=m-L-1;
  for (i=1,j=0;i<L+k; i++,j++)
  s2[j]=s1[i];
}
#include <stdio.h>
#include <conio.h>
void main(void)
{
 char str1[80],str2[80]; int L,k,m=0;
 clrscr();
 puts("\n сөз тізбегін енгізіңіз \n:");
```

```
 while((str1[m++]=getchar())!='\n')
 ;
 printf("\n");
 /* m өз мәнін сақтап қалады */
 printf("\n L және k мәндерін енгізіңіз:\n");
 scanf("%d %d",&L,&k);
 substr(str1,str2,L,k,m);
 printf("%s\n",str2);
 }
```

6-мысал. Берілген s1 символдар тізбегінің құрамында бағыныңқы s2 тізбегінің бар жоқтығын анықтайтын index функциясын қарастырайық. Егер s2 тізбегі s1 символдар тізбегінің құрамында бар болса, онда index функциясы s2 тізбегінің s1-дегі орнын (индексін) анықтайды:

```
index(s1,s2)
char s1[],s2[];
{
  int i,j,k;
  for (i=0,s[1]!='\0';i++)
  {
  for (j=i,k=0;s2[k]!='\0'&& s1[j]==s2[k];j++,k++)
  ;
  if(s2[k]=='\0')
  return(i);
  }
  return(-1);
}
char str1[]= "информатика";
char str2[]= "форма";
#include <stdio.h>
#include <conio.h>
void main(void)
{clrscr();
 printf("%d\n", index(str1,str2));
}
```

7-мысал. Функцияның параметрі ретінде оған нұсқауышты пайдалану мысалын қарастырайық. Функция cos(x)-тің туындысын есептейді.

```
double proiz(double x,double dx,double(*f)(double x));
double fun(double z);
int main()
{
 double x; /*туынды есептелетін нүкте*/
```

```
 double dx; /*ығысу*/
 double z;  /*туындының мәні*/
 scanf("%f,%f",&x,&dx);
 z=proiz(x,dx,fun); /* функцияны шақыру */
 printf("%f",z);
 return 0;
}
double proiz(double x,double dx,double(*f)(double z))
{
 double xk,xk1,pr;
 xk=fun(x);
 xk1=fun(x+dx);
 pr=(xk1/xk-1e0)*xk/dx;
 return pr;
 }
 double fun( double z)
 {
  return (cos(z));
 }
```

Жоғарыда қарастырылған **fun** функциясын кез келген функцияның туындысын табу үшін қолдануға болады. Ол үшін анықталған **fun** функциясының тұлғасын өзгерту керек немесе **proiz** функциясын шақырған кезде туындысы қарастырылып отырған функцияның атымен шақырса болғаны. Мысалы, sin(x) функциясының туындысын табу үшін жоғарыда қарастырылған **proiz** функциясын

z=proiz(x,dx,sin);

түрінде шақыру жеткілікті болып саналады.

Бақылау сұрақтары

1. *Тұтынушы функциясын сипаттау дегеніміз не?*
2. *Егер функцияның параметрлері жоқ болса, ол қалай сипатталады?*
3. *Тұтынушы функциясы мен стандартты функциялардың айырмашылығы неде ?*
4. *Функция аргументтерін беру жолдарын атаңыз?*
5. *"Жергілікті" және "ауқымды" айнымалылардың айырмашылығы.*
6. *"Жергілікті" айнымалылардың әрекет ету аймағы ұғымы.*
7. *"Ауқымды" айнымалылардың әрекет ету аймағын түсіндіріңіз.*
8. Return *операторының қолданылуы және атқаратын қызметі.*

ТАПСЫРМАЛАР

1. Үш натурал сандар берілген. Олардың ең үлкен ортақ бөлгішін (ЕҮОБ) анықтайтын функцияны құру керек.

2. *a,b,c,d* кесінділері берілген. Осы кесінділердің кез келген үшеуін қарастыра отырып, олардан үшбұрыш тұрғызуға болатын/болмайтынын анықтап, үшбұрыш тұрғызылатын жағдайда, оның ауданын табу керек. Үшбұрыштың ауданы $S = \sqrt{p*(p-x)(p-y)(p-z)}$, мұндағы $p=(x+y+z)/2$.

3. Натурал N саны берілген. Ол екі *x* және *y* бүтін сандарының квадраттарының қосындысына тең болатын болса $N=x^2+y^2$, онда *x, y* сандарын анықтайтын функцияны құру керек.

4. Нақты *x, y* ($x>0$, $y>1$) сандары берілген, $y^{k-1} \le x < y^k$ шартын қанағаттандыратындай k бүтін санын (оң, теріс немесе нөлге тең) табу керек.

5. Натурал *N* саны берілген ($N > 99$). Ондағы жүздіктер санын анықтаңыз.

6. Натурал *N* саны берілген ($N \le 99$). N^2 саны *N* санының цифрларының қосындысының кубына тең екендігін тексеру керек.

7. Натурал *N* саны берілген ($N > 10000$). *N* санының алғашқы *K* разрядтарының қосындысын анықтаңыз ($K \le 4$).

8. Натурал *n, m* сандары берілген, *n* санының соңғы *m* цифрларының көбейтіндісін табу керек.

9. Натурал *N* саны берілген. *N* санының жазбасынан басқа цифрлардың ретін өзгертпей 0 және 5 цифрларын алып тастау керек. Мысалы, 59015509 санынан 919 саны шығады.

10. Натурал *N* саны берілген. K^2-қа бөлінетін және K^3-қа бөлінбейтін барлық натурал *K*-ларды табу керек.

11. Натурал *n* және *m* сандары берілген, A(*n*,*m*) мәнін есептеу керек, мұндағы

$$A(n,m) = \begin{cases} m+1, & \text{егер } n = 0 \\ A(n-1,1), & \text{егер } n \neq 0, m = 0 \\ A(n-1, A(n,m-1)), & \text{егер } n = 0, m = 0 \end{cases}$$

12. Егер *N* цифрдан тұратын натурал санның цифрларының қосындысын *n*-ші дәрежеге шығарғанда, сол санның өзіне тең болатын болса, ондай сан Армстронг саны деп аталады (мысалы, $153=1^3+5^3+3^3$). Екі, үш және төрт цифрдан тұратын барлық Армстронг сандарын табу керек.

13. 1-ден *n*-ға дейін нөмірленген ($n=10$) *n* елді мекен бар. Кейбір қос елді мекендер жолдармен қосылған. Осы жолдар арқылы бірінші елді мекеннен *n*-шісіне жетуге болатынын немесе болмайтынын анықтау керек. Жолдар туралы ақпарат i-ші және j-ші елді мекендердің жолдармен қосылғанын білдіретін i және j (i<j) қос сандарының тізбегі түрінде берілген. Ол тізбектің аяқталғанының белгісі – қос нөл (00).

14. Екі үшбұрыштың төбелерінің координаталары берілген. Олардың қайсысының ауданы үлкен екенін анықтау керек.

15. Жазықтықтағы үш түзу $a_k x+b_k y=c_k$ (k = 1,2,3) теңдеулерімен берілген. Егер ол түзулер қос-қостан қиылысып, үшбұрыш құрайтын болса, сол үшбұрыштың ауданын табу керек.

16. Екі жай санның бір-бірінен айырмашылығы 2-ге тең болса, олар **"егіздер"** деп аталады (мысалы, 41 және 43 сандары). *[*n, *2*n*]* аралығындағы барлық "егіздерді" анықтау керек, мұндағы n – 2-ден үлкен бүтін сан.

17. Натурал N санын енгізіңіз. Келесі алгоритм бойынша Паскаль үшбұрышын құру керек:

C_0^0

C_1^0 C_1^1

C_2^0 C_2^1 C_2^2

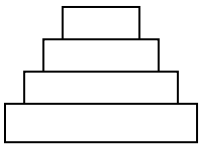
C_3^0 C_3^1 C_3^2 C_3^3

C_4^0 C_4^1 C_4^2 C_4^3 C_4^4

……..

18. $C_n^m = \frac{n!}{m!(n-m)!}$ анықтау керек, мұндағы $n!=1\cdot2\cdot3\cdot\ldots\cdot n$, n – санының факториалы.

19. **"8 ферзі туралы есеп".** Шахмат тақтасында 8 ферзіні олар бір-бірін "жемейтіндей" етіп орналастыру керек (барлығы 92 орналастыру бар).



20. **"фон Нейман әдісі".** Нақты N саны берілген. Оларды кемімейтіндей етіп төмендегідей түрде реттеу керек: А және В екі жиымын құрып, бастапқы сандарды А-ға жазу керек; көрші екі санды реттеп (A_1 және A_2, A_3 және A_4 және т.с.с.) оларды В-ға жазу керек; В-дан екі-екіден реттелген көрші сандарды алып, оларды реттелген төрттіктерге жинақтап, қайтадан А-ға жазу керек; одан кейін әрбір көрші реттелген төрттіктерді В-дан алып, оларды реттелген сегіздіктерге жинақтап А-ға көшіру керек, т.с.с.

21. Әрқайсысы 100 элементтен тұратын X және Y жиымдары берілген. Осы екі жиымның әрбір элементін бір-бірімен орын алмастыруды ($x_k \leftrightarrow y_k$) функция түрінде ұйымдастыратын пограмма құру керек, мұндағы k=1,2,...,100. Аралық мәндерді сақтайтын қосымша айнымалыларды қолдануға болмайды.

22. N саннан тұратын бүтін сандар тізбегі берілген. Олардың арасында бірдей екі сан кездеспейді. Қалған сандар өсу ретімен орналасатындай етіп, сол реттілікке көнбейтін сандарды өшіріп тастау керек.

23. 1,2,3,4,5,6,7,8,9 цифрлары жазылған 9 кәртішке бар (әрбіреуіне бір цифрдан жазылған). Осы кәртішкелерден екі сан құрылған. Бірінші сан – бөлшектің алымын, ал екіншісі – бөлшектің бөлімін құрайды. Әрбір кәртішке міндетті түрде пайдаланылуы тиіс. Бөлшектің бөлімі 5 цифрдан тұрады, олай болса бөлшекті алымын қалған төрт цифр құрайды. Бөлшектің мәні дәл 1/2-ге тең болатындай барлық комбинацияларды табыңыз. Мысалы, бөлшектің алымы – 6729, бөлімі – 13458 (барлық кәртішкелер пайдаланылған), 6729/13458 =1/2.

24. 1,2,3,4,5,6,7,8,9 цифрлары жазылған 9 кәртішке бар (әрбіреуіне бір цифрдан жазылған). Екі натурал N және M сандарын енгізіңіз. Мысалы N=2, M=3. N/M бөлшегін (2/3) дәл құрайтын кәртішкелердің комбинацияларын құру керек, мысалы, 4/6, 12/36, 123/369, т.с.с. (сондайлар бар болатын болса). Егер ондай комбинация саны бірден көп болса, олардың бәрін шығару қажет.

9. ҚҰРЫЛЫМДАРДЫ ПАЙДАЛАНУ

Си тіліндегі құрылымдар өзара логикалық байланысқан әртүрлі типті мәліметтерді байланыстырады. Мәліметтердің құрылымдық типтері келесі сипаттаумен анықталады:

```
struct  құрылым_ аты
{
  элементтерді сипаттау;
};
```

Құрылымға компьютер жадынан орын бөлу үшін құрылымдық айнымалыны анықтап алу керек:

```
struct құрылым_аты айнымалы_аты;
```

Құрылымдарды анықтау барысында олардың элементтеріне бастапқы мәндерді меншіктеуге болады. Құрылым элементтерінің мәндерін енгізу үшін ағымдық енгізу операторы *cin*>> немесе форматпен енгізу операторы – *scanf* қолданылады.

Құрылымдармен келесі амалдарды орындауға болады:

1) Құрылымның адресін алу амалы. Құрылымның адресін алу үшін құрылымдық айнымалыға нұсқауыш (&) амалы қолданылады.

2) Құрылымның элементін пайдалану үшін нүкте (.) амалы (тікелей ену) немесе (->) (нұсқауыш бойынша ену) амалы қолданылады.

Құрылымдық айнымалы float, int, char типті айнымалылар сияқты пайдаланылуы мүмкін. Мысалы:

```
struct gr /* құрылым аты */
{char fio[10];      /* құрылым элементі */
 int est[25]:       /* құрылым элементі */
 int nomer;         /* құрылым элементі */
}
gruppa1;                /*құрылымдық айнымалы аты */
struct gr gruppa2;  /* құрылымдық айнымалыны хабарлау*/
```

Егер құрылымды сипаттаушы берілген файлдағы барлық функциялардың алдында орналасса, онда ол құрылымды осы файлдағы барлық функциялар пайдалана алады. Құрылымдық айнымалыны анықтау барысында оған алғашқы мәндерді меншіктеуге болады (құрылым өрістеріне мән беруге болады). Мысалы:

```
  struct date { int day,month,year;};
 d[5]={{1,3,1980},
       {5,1,1990},
       {1,1,2002}};
```

Құрылымдарды пайдалану мысалы:

```
#include <stdio.h>
#include <string.h>
```

```
#include <conio.h>
struct Spisok {
char Fio[20];
char Grup[10];
int Ot[3];
float S_Bal;
} *sved;

 void Vvod(int nom,struct Spisok *sved)
{printf("\n Мәліметтерді енгізіңіз:");
scanf("%d",nom+1);
printf("\n фамилиясы - ");
gets(sved.Fio);
printf("топ нөмірі - ");
gets(sved.Grup);
float s=0;
for(int i=0;i<3;i++) {
printf("\n бағалары - ");
scanf("%d",sved.Ot[i]);
s+=sved.Ot[i];
}
sved.S_Bal=s/3.;
return;}

void main(void)
{struct Spisok Stud[50]; int i,N; char Bukva;
clrscr();
printf("\n 50-ден кіші бүтін сан енгізіңіз");
scanf("%d",&N);
for(i=0;i<N;i++) Vvod(i,&Stud[i]);
printf ("\n студенттердің тізімі:");
for(i=0;i<N;i++)
printf("\n %20s  %10s  %4.2f",Stud[i].Fio,
       Stud[i].Grup,Stud[i].S_Bal);
printf ("\n  Фамилиядағы әріпті іздеу");
scanf("%c",&Bukva);
printf("\n  Студенттер туралы мәлімет");
int kod_p=0;
for(i=0;i<N;i++)
if(Stud[i].Fio[0]==Bukva){ kod_p=1;
printf("\n %20s  %10s%4.2f",Stud[i].Fio,
        Stud[i].Grup,Stud[i].S_Bal);}
```

```
if(kod_p==0) printf("Ондай жазба жоқ!");
      printf ("\n Жазбаны іздеу");
            Vvod(-1,sved);
kod_p=0;
for(i=0;i<N;i++)
if(memcmp(sved,&Stud[i],sizeof(sved))==0) { kod_p=1;
printf("\n Іздеу: \n  %20s  %10s",Stud[i].Fio,
       Stud[i].Grup);}
 if(kod_p==0) printf("Ондай жазба жоқ!");
    getch();}
```

Си тілінде құрылымдарда өрістердің ерекше типі – биттік өрістерді қолдануға мүмкіндік жасалған. Биттер өрісі дегеніміз – компьютер жадында бүтін типті айнымалылардың аймағында орналасқан көрші разрядтар (биттік – екілік) топтары.

Ақпараттарды мәліметтер құрылымдарында сақтау үшін бірнеше бит жеткілікті болса, биттер өрістерін пайдаланған ыңғайлы.

Биттік өрісті сипаттаудың синтаксисі:

тип[аты]:ені;

Құрылымның элементтері бір немесе бірнеше құрылымдық айнымалы болуы мүмкін. Құрылымдармен жұмыс істеудің бірнеше мысалын қарастырайық.

1-мысал.

/* Программа sum функциясында st құрылымының avans және zarp элементтерін пайдалануды көрсетеді */

```
 #include <conio.h>
 #include <stdio.h>
 #define k 2
 #define PE  printf("\n айға алдын-ала берілетін жалақы
                    мөлшері%d...",k);
 #define PE1
 printf("\n================================");
 #define PE2  printf("\n");
 struct fund{
             char *mes_avans;
             float avans;
             char *mes_zarp;
             float zarp;
           };
main()
{
static struct fund st={"тамыз", 600000,
```

```
                        "қазан", 900000};
float sum(),res;
res=sum(st.avans,st.zarp);
printf(" \n Жалпы сома тең %8.2f теңге.",res);
PE2;PE1;getch();
}
float sum(x,y)
float x,y;
{
clrscr();PE;PE1;PE2;
return(x+y);}
```

2-мысал

```
/* Программа тұтынушының sum функциясына st құрылымының адресін нұсқауыш амалы арқылы жіберуді көрсетеді
*/
#include <stdio.h>
#include <conio.h>
struct spis{
             char *s_vans;
             float vans;
             char *s_za;
             float za;};
main()
{
 static struct spis str={"қыркүйек",600,
                         "қазан",1000};
 float sum(),res;
 res=sum(&str);
 printf("даналардың жалпы саны =%8.0f дана",res);
 getch();
}
float sum(t)
struct spis *t;
{clrscr();
return(t->vans+t->za);
}
```

3-мысал. Программа sum функциясына құрылымдар жиымының адресін жіберуді көрсетеді. st құрылымдар жиымының бастапқы адресі m нұсқауыш типті айнымалыға орналастырылады. Енді sum функциясына құрылымдар жиымының бастапқы адресі белгілі. Бұл адресті біле

отырып -> амалы арқылы бірінші құрылымның элементтерін пайдалануға болады.

```
#include <stdio.h>
#include <conio.h>
struct fund{
            char *mes_avans;
            float avans;
            char *mes_zarp;
            float zarp;
           };
main()
{
static struct fund st[2]={{"қыркүйек",500000,
                           "қазан",700000},
                           {"қараша",600000,
                           "желтоқсан",800000}};
float sum();
printf("жалпы сомасы= %8.3f ",sum(st));
getch();
}
float sum(m)
struct fund *m;
{
float res;
int i; clrscr();
for(i=0,res=0;i<2;i++,m++)
res+=m->avans+m->zarp;
return(res);
 }
```

4-мысал. Компьютерге студенттер жайлы ақпарат енгізу керек, студенттік топ жиымының әрбір элементі келесі мәліметтерден тұрады:

а) студенттің тегі және аты;

в) программалау пәнінен тапсырылған емтиханның нәтижесі.

Программа студент туралы ақпаратты келесі тәртіппен шығарады: оның топ тізіміндегі реттік нөмірі (енгізілген ақпараттан тұратын жиым индексі).

Си-программаның бір нұсқасының мәтіні:

```
#include <stdio.h>
#include <conio.h>
struct student
              {
char fio[10];/*студенттің тегін,атын сақтайтын өріс */
```

```
                int est1;   // баға
              } gruppa[25]; // топ тізімі жиымы
void main(void)
{
int i,j,k,priznak;
clrscr();
printf("\n 25 адамға дейінгі тізімді құру");
           /*---------Топ тізімі ----------*/
for (i=1;i<=25;i++)
{
printf("\n студент туралы мәліметті енгізіңіз %d
және\\n (0-енгізуді аяқтау )\\n",i);
printf("\n Тегі және инициалы: ");
fflush(stdin); // енгізу буферін тазалау функциясы
gets(gruppa[i].fio);
printf("\n программалаудан алынған баға: ");
fflush(stdin);
scanf("%d",&gruppa[i].est1);
k=i;  // енгізілген ақпараттың ағымдағы мәнін
//сақтаймыз
printf(" \n Белгі? ");
scanf("%d",&priznak);
k=i;
if(!priznak) break;
}
/*---------Тізімді қарап шығу ------------*/
for (i=0; i<25; i++)
{
printf("\n қай студент туралы ақпарат?\n");
scanf("%d",&j);
if(j>k)
{
printf("\n Мәлімет жоқ! ");
continue;   // мәлімет болмаған жағдайда
}           // циклдің келесі қадамына көшу
printf(" \n СТУДЕНТ: %s \n",gruppa[j].fio);
printf(" \ программалау пәнінен n көктемгі сессия,
       бағасы\: %d",gruppa[j].est1);
printf("\n Белгі? ");
scanf("%d",&priznak);
if(!priznak)break;  }
puts("\n ПРОГРАММА жұмысы аяқталды!
```

```
          Press any key...");
getch();
}
```

Бақылау сұрақтары

1. *Си тілінде құрылымдар мен жолдар қалай сипатталады?*
2. *Айырмашылықтарын түсіндіріңіз.*
3. *"Құрылымға нұсқауыш" терминін түсіндіріңіз?*
4. *Нұсқауыштарды құрылымдарға қатысты қолдану ерекшеліктерін атаңыз.*
5. *Құрылымдармен жұмыс істеу ерекшеліктерін түсіндіріңіз.*
6. *Құрылым элементтерінқорытып шығару мүмкіндіктерін көрсетіңіз.*

ТАПСЫРМАЛАР

Келесі мәліметтерден құралған төрт студент туралы ақпаратты енгізіңіз:
- тегі және инициалы;
- туған жылы;
- оқуға түскен жылы;
- бірінші семестрдің бағасы:
- физика;
- жоғарғы математика;
- информатика;

1. Алфавит бойынша реттелген студенттердің тізімін шығару керек.

2. Туған жылы бойынша реттелген студенттердің тізімін көрсету қажет.

3. Оқуға түскен жылы бойынша реттелген озат студенттердің тізімін шығару керек.

4. Сессияны 4 және 5-ке тапсырған студенттердің анкеталық мәліметтері шығарылуға тиіс.

5. Тегі Б әрпінен басталатын студенттердің тізімін және олардың барлық пәндер бойынша бағаларын көрсету керек.

6. Оқу озаттарының анкеталық мәліметтерін шығару қажет.

7. Тегі А әріпінен басталатын студенттердің тізімін және олардың туған жылдары көрсетілуі тиіс.

8. Сессияда 3 деген бағалары бар студенттердің анкеталық мәліметтерін шығару керек.

9. Тегі В және Г әріпінен басталатын студенттердің тізімін және олардың бағаларын шығаратын программа құру керек.

10. Сессияны 3-сіз тапсырған студенттердің тізімін және олардың туған жылдарын шығару керек.

11. Барлық студенттердің жалпы орта балын және осы орта балдан жоғары балл жинаған студенттердің тізімі көрсетілуі тиіс.

12. Барлық студенттердің жалпы орта балын және осы орта балға тең балл жинаған студенттердің тізімін шығару керек.

13. Сессияда “екілік” алған студенттердің анкеталық мәліметтерін шығару керек.

14. Информатика пәнін 5-ке тапсырған студенттердің анкеталық мәліметтерін экранға шығару қажет.

15. Физиканы 4-ке және жоғарғы математиканы 5-ке тапсырған студенттердің анкеталық мәліметтері шығарылуы тиіс.

10. ФАЙЛДАРДЫ ПАЙДАЛАНУ

Файл – сыртқы есте сақтау құрылғыларында (магниттік дискілерде) орналастырылған және мәлімет өңдеу, тасымалдау кездерінде біртұтас күйде қарастырылатын мәліметтер жиыны.

Файлдармен жұмыс істеу үшін оларды алдымен ашу керек, яғни файл туралы мәліметті – атын, адресін программаға белгілі ету қажет.

Си тілінде файл ашу *fopen*() функциясы арқылы орындалады. Ол сыртқы құрылғыдағы физикалық файлды, мысалы, A:\NUR.DAT, программадағы оның логикалық атымен байланыстырады. Логикалық ат дегеніміз – файлға нұсқауыш, яғни файл туралы мәлімет сақталатын жады аймағына сілтеме жасау. Файлға нұсқауыш сипатталуы тиіс.

Сонымен, файлдармен жұмыс істегенде нұсқауыштар қолданылады. Файлға нұсқауыш мынадай түрде сипатталады:

```
FILE  *fp;
```

мұндағы FILE типі – <stdio.h> тақырыптық файлында анықталған құрылым. Бұл нұсқауышты көрсетілген файлмен сол файл ашылғаннан бастап, байланыстырып қоюға болады. Ол үшін келесі функция пайдаланылады:

```
fopen ("файл адресі/аты","қатынасу типі");
```

ол файлға нұсқауыш мәнін қайтарады немесе қате болса, NULL мәнін береді.

Мысалы, мынадай оператор орындалуы нәтижесінде

```
fp = fopen ("ex1.txt", "w");
```

жұмыс бумасындағы ex1.txt файлы оған мәлімет жазу (write) үшін ашылады, ал программада бұл файлды **fp** нұсқауышы арқылы пайдаланамыз (яғни *fopen()* функциясы файлдың сыртқы атын оның программада қолданылатын ішкі логикалық атымен байланыстырады).

Сонымен, файлды ашатын fopen() функциясының жалпы жазылуы:

```
fp = fopen(name,mode)
```

мұнда **fp** – файлға сілтейтін нұсқауыш; **name** – файлдың адресін де көрсетуге болатын аты, көбінесе символдық тіркес арқылы жазылады;
mode – файл қандай режимде қолданылатынын көрсететін параметр, бұл да символдар тіркесімен төмендегідей түрде жазылады:

"r" – файлды оқу үшін ашу (файл бұрын ашылған болуы тиіс);
"w" – бос файлды информация жазу үшін ашу;
"a" – файл соңына мәлімет қосып жазу үшін оны ашу;
"r+" – файлдан информация оқу және оған информация жазу үшін ашу);
"w+" – бос файлдан информация оқу және оған мәлімет жазу үшін файл ашу (бұрын файл болса, ол өшіріледі);
"a+" – файлдан информация оқу және оның соңына информация қосып жазу үшін ашу).

"t" – файл мәтіндік (текстік) режимде ашылады, ол rt, wt, at, т.с.с. болып көрсетіле береді.

"b" – файл екілік режимде ашылады, rb, wb, ab, т.с.с. болып көрсетіле береді.

Келісім бойынша файл көбінесе мәтіндік режимде ашылады.

Файлмен жұмыс істеп болған соң, оны жабу үшін келесі функция қолданылады:

fclose(файл_нұсқауышы).

Файлға мәлімет жазу/оқу үшін жалпы енгізу/шығару функциялары тәріздес бірсыпыра функциялар пайдаланылады:

fprintf(), fscanf(), fputs(), fgets(), getc(), putc(), fgetc(), fputc().

Бұлардың ішіндегі **getc()/fgetc(), putc()/fputc()** функциялары әрекеттері ұқсас, айырмашылықтары тек **getc**() пен **putc**() макроанықтаулар да, ал **fgetc**() және **fputc**() – нағыз функциялар болып табылады.

Барлық файлдық функциялар прототиптері <stdio.h> файлында орналасқан.

Файлдардан мәлімет жазу/оқу әрекеттерін үш топқа бөлуге болады:

- символдарды енгізу/шығару операциялары;
- сөз тіркестері жолдарын енгізу/шығару операциялары;
- блок бойынша енгізу/шығару операциялары.

Символдарды енгізу/шығару операцияларында файлдан бір ғана символ оқылады немесе оған бір символ жазылады. Мысалы:

fgetc(FILE *fp); ашылған файлдан символ оқиды.

fputc(int ch,FILE *fp); **ch** символы кодын файлға жазады.

Сөз тіркестері жолдарын енгізу/шығару операцияларында бір мәлімет алмасуы кезінде файлдан сөз тіркесі жолдары оқылады немесе оған сөз тіркесі жазылады. Мысалы:

gets(char *S); файлдан мәлімет байттарын '\n' символы кездескенше біртіндеп оқиды да, оларды **S** тіркесіне орналастырып, '\n' символын '\0' (нөл-терминатор) белгісіне айналдырады.

fgets(char *S,int m,FILE *fp); **fp** түрінде сипатталған файлдан байттарды оқып, оларды '\n' символы кездескенше немесе *m* байт оқылып біткенше S тіркесі мәні ретінде жазып шығады.

Блок бойынша енгізу/шығару операцияларында мәліметтер алмасу олардың бір блогын толық қамтиды. Мысалы:

fread (void *ptv, int size, int n, FILE *fp);

мұнда **fp** файлынан әрқайсысы **size** байттан тұратын мәліметтің **n** блогы **ptv** нұсқауышы көрсетіп тұрған жады аймағына оқылады (оқылатын блок үшін алдын ала орын дайындап қою қажет).

fwrite (void *ptv, int size, int n, FILE *fp);

мұнда **ptv** нұсқауышы көрсетіп тұрған жады аймағында орналасқан әрқайсысы **size** байттан тұратын мәліметтің **n** блогы ашық тұрған **fp** файлына жазылады.

fprintf және fscanf функцияларын пайдалану

fprintf - файлға информация жазу үшін, ал **fscanf** – файлдан информация оқу үшін қолданылады. Олардың жалпы жазылу түрі:

fprintf(fp, "спецификация шаблоны",p);

мұндағы **p**–өрнек;

fscanf(fp, "спецификация шаблоны",адрес);

Мысал: Бүтін сандар тізбегін n=5 пернелерден енгізіп, оларды файлға жазу керек болсын.

```
// файл ашып, оған 5 сан жазу
#include <stdio.h>
#include <conio.h>
main()
{ int k,i,n=5;
  char fname[]="a:\\num.txt\0";
  clrscr();
  FILE *fp;
  fp=fopen(fname,"w");
  clrscr();
  printf("Енгізілген сандар %s файлына жазылады\n",
          fname);
  puts("Әр сан енгізілген соң, Enter басыңыз\n");
  for(i=0; i<n;i++)
  {scanf("%i",&k);
   fprintf(fp,"%d ",k);
  }
  fclose(fp);
  printf("Енгізілген сандар %s файлына жазылды\n",
          fname);
  getch();
}
```

Нәтижесі:

Енгізілген сандар a:\num.txt файлына жазылады

Әр сан енгізілген соң, Enter басыңыз
1
2
3
4
5
Енгізілген сандар a:\num.txt файлына жазылды
Файлдағы мәлімет: 1 2 3 4 5

Бес бүтін санды n=5 бұрыннан бар file1.txt файлына жазу керек, сол файлды мәлімет қосу үшін ашып, оған 5 сан жазып, сонан соң ондағы мәліметті оқып тексеру керек.

```
/* файлға мәліметтер қосу */
#include <math.h>
#include <stdio.h>
#include <conio.h>
main()
{ int i,k=0,s=0,n=5,y,x;
  char fname[]="file1.txt";
  clrscr();
  FILE *fp;
  fp=fopen("file1.txt","at");
  if ((fp=fopen(fname,"at")) == NULL)
   { printf("\nФайлды ашуда қате болды");
     getch();
     return(0);
   }
  puts("Enter арқылы бөліп, 5 сан енгізіңіз:");
  for(i=1; i<=n;i++)
   {scanf("%i",&x);
    fprintf(fp,"%d ",x);}
  fclose(fp);
  getch();
}
```

Нәтижесі:
Enter арқылы бөліп, 5 сан енгізіңіз:
11
22
33
44
55

Файлдағы мәлімет: 1 2 3 4 5 11 22 33 44 55

Дискідегі file1.txt файлын мәліметтер оқу үшін ашып, ондағы мәліметті оқып, файлдағы тақ сандардың қанша екенін және қосындысын табу керек.

```
/* файлдан мәліметтер оқу */
#include <math.h>
#include <stdio.h>
#include <conio.h>

main()
{ int i,k=0,s=0,n=5,y,x;
  char fname[]="file1.txt";
  clrscr();
  FILE *fp;
  fp=fopen(fname,"r");

  /* Оқу үшін файл ашу */
  fp=fopen("fp","rt");

   if ((fp=fopen(fname,"rt")) == NULL)
  {  printf("\n Файл ашуда қате болды");
     getch();
     return(0);
  }
  puts("Файлдан оқылған сандар:");
  while(!feof(fp))
   {fscanf(fp,"%i",&y);
    printf(" %i ",y);
    if (y%2==0)  {s+=y;k=k+1;}
   }
  fclose(fp);
  printf("\nТақ сандар қосындысы s = % i,
            олардың саны k=%i", s, k);
  getch();
}
```

fgets және fputs функцияларын пайдалану

Сөз тіркестерімен жұмыс істегенде **fgets** және **fputs** функциялары қолданылады:

fgets функциясының жалпы жазылу түрі:

```
fgets(нұсқауыш,MAXLEN,fp);
```

нұсқауыш – ЭЕМ жадындағы орынға сілтейтін нұсқауыш; **MAXLEN** – оқылатын тіркестің максимальды ұзындығы; **fp** - файл нұсқауышы.

Мысалы:

```
#define L 20
main()
{
  FILE *fp;
  char *st[n];
  fp=fopen("stroka","r");
  while(fgets(st,L,fp)!=NULL)
puts (st);
}
```

Файлдарға сөз тіркестерін жазу үшін **fputs** функциясы келесі түрде қолданылады:

```
    status=fputs(қатар нұсқауышы,fp);
```

status – бүтін сан, оның мәні **eof** функциясында жазылады, егер **fputs()** функциясы файлдың соңына шыққан болса немесе қате тапса, **fputs** функциясы жазылатын жолдың соңына **/0** символын жазбайды.

Төменде осы функцияларды пайдалану мысалы келтірілген.

```
//
#include <stdio.h>
void main ()
{
 int n;
 char str[50], str1[50], ch;
 FILE *fp;
 // Файлға мәлімет жазу
 fp = fopen("ex.txt","w");
 puts ("Бүтін сан енгізіңіз: "); scanf("%d",&n);
 fprintf (fp, "%d\n", n) ;
 puts ("Символ енгізіңіз: "); ch=getchar();
 putc (ch, fp);
 puts ("Сөз тіркесін енгізіңіз: "); gets(str);
 fputs(str, fp);
 fclose(fp);
 // Файлдан мәлімет оқу
 if((fp = fopen("ex.txt","r")) != NULL)
  {
    fscanf (fp, "%d", &n); printf ("n=%d\n", n) ;
    ch = getc (fp); putchar (ch);
    fgets(str1, 50, fp); puts (str1);
```

```
    fclose (fp);
  }
  else printf ("\nФайлдан мәлімет оқылмайды!");
}
```

Мұндағы **fgets()** функциясының екінші параметрі N – оқылатын символдар саны, оған '\0' белгісі де қосылады. Бұл функция өз жұмысын N-1 символын оқығаннан кейін немесе '\0' белгісі кездескенде аяқтайды. Екеуінде де сөз тіркесі соңына '\0' белгісі қосылады. **fgets()** функциясы оқылған сөз тіркесі адресін қайтарады немесе файл оқылып болғанда (не қате шықса), NULL белгісін береді.

fputs() функциясы әрекет дұрыс орындалса, соңғы оқылған символ кодын қайтарады, ал қате болса, EOF (файл соңы) белгісін береді. Бұл функция курсорды автоматты түрде келесі жолға көшірмейді.

Жоғарыдағы функциялар файл мәліметтерін біртіндеп, символдан соң келесі символды қарастыра отырып өңдейді. Си тілі файлдармен жиым сияқты жұмыс істеуге де мүмкіндік береді, яғни кез келген байтты жеке өңдеуге де болады. Файл ішіндегі белгілі бір орынды айқындау үшін мына функция қолданылады:

```
fseek(файлға нұсқауыш,бастапқы нүктеден ығысу,
        бастапқы нүкте);
```

Екінші аргумент типі *long*, оның мәні оң да, теріс те болуы мүмкін. Ол бастапқы нүктеден қанша орынға (байтпен) ығысу керек екендігін көрсетеді. Үшінші аргумент файлдағы бастапқы нүкте орнын анықтайтын код болып табылады. Осы код үшін мынадай мәндер тағайындалған:

0 - файл басы;
1 - ағымдағы позиция;
2 - файл соңы.

Дұрыс орындалғанда, *fseek()* функциясы 0 мәнін береді, ал егер қате (мысалы, файлдың сол жақ шекарасынан ары аспақшы болғанда) болса, онда -1 береді.

fwrite және fread функцияларын пайдалану

Құрылымдарды пайдаланатын файлдармен жұмыс істеу кезінде **fread()/fwrite()** функцияларын пайдаланған ыңғайлы. Олардың жазылуы:

```
fread(ptr,size,n,fp);
```

мұнда **fp** файлынан әрқайсысы **size** байттан тұратын мәліметтің **n** блогы **ptv** нұсқауышы көрсетіп тұрған жады аймағына оқылады (оқылатын блок үшін алдын ала орын дайындап қою қажет).

```
fwrite (ptr,size,n,fp);
```

мұнда **ptr** нұсқауышы көрсетіп тұрған жады аймағында орналасқан әрқайсысы **size** байттан тұратын мәліметтің **n** блогы ашық тұрған **fp** файлына жазылады.

Осы функцияларды қолданудың бір мысалын келтірейік:

```
typedef struct
{
  char author [30];
  char title [50];
  int pages;
} BOOK;
BOOK b1={"Kernighan", "C Language", 256},b2;
FILE *fp;
void main()
{...
 fp=fopen("struct.txt", "w+");
// файл әрі оқу, әрі жазу үшін ашылды
 fwrite(&b1, sizeof(BOOK), 1, fp);
 fseek(fp,0,0); // маркер файл басына
 fread(&b2, sizeof(BOOK), 1, fp);
 printf("Авторы – %s,аты – %s,беттер саны –
        %d\n",b2.author, b2.title, b2.pages);
}
```

Бақылау сұрақтары

1. *Файл дегеніміз не? Ол не үшін пайдаланылады?*
2. *Файл қайда орналасады және қалай белгіленеді?*
3. *Файл ашу функциясы қалай жазылады?*
4. *Файл атын программада қалай анықтаймыз?*
5. *Файлдың қолданылу режимдері қалай көрсетіледі?*
6. *Файлға мәлімет жазу/оқу функциялары.*
7. *Файлға жаңа элемент қалай қосылады?*
8. *Мәлімет оқылған файлға мәлімет жазуға бола ма?*
9. *Файлдан ақпарат оқу үшін не істеу керек?*
10. *Файлға ақпарат жазу үшін не істеу керек?*
11. *Символдар мен сөз тіркестерін файлға жазу үшін не істеу керек?*
12. *Құрылымдарды пайдаланатын файлдар қандай функцияларды пайдаланады?*

ТАПСЫРМАЛАР

1. Файлдан сандар оқып, солардың ішіндегі теріс сандар қанша екенін анықтайтын программа құру керек.

2. Файлдан сөз тіркесін оқып, солардың ішіндегі 6 символдан артық сөздерді экранға шығарып, басқа файлға жазып шығыңдар.

3. Файлдан сандар оқып, солардың арифметикалық ортасын файл соңына қосып жазу керек.

4. Файлдан сөз тіркесін оқып, солардың керісінше жазылған нұсқасын басқа бір файлға жазып шығыңдар.

5. Файлдан сандар оқып, солардың максимумын анықтайтын программа құру керек.

6. Файлдан сөз тіркесін оқып, солардың ішіндегі бос орын орнына сызықша жазып оны басқа бір файлға жазып шығу керек.

7. Файлдан сандар оқып, солардың минимумын анықтайтын программа құру керек.

8. Бір файлда екі сөйлем жазылған, соның екінші сөйлемін басқа файлға жазып шығу керек.

9. Файлдан сандар оқып, солардың көбейтіндісін анықтайтын программа құру керек.

10. Сөз тіркесінен тұратын файлдағы бас әріптерді кіші әріптерге айналдырып, басқа файлға жазып шығу керек.

11. Файлдан сандар оқып, солардың нешеуі 5-тен артық екенін анықтау керек

12. Сөз тіркесінен тұратын екі файл берілген. Осы екі файлдағы сөз тіркестерін біріктіріп, үшінші файлға жазып шығыңдар.

13. Файлдан сандар оқып, солардың қосындысын анықтау керек

14. Сөз тіркесінен тұратын екі файл берілген. Осы екі файлдағы сөз тіркестерінің орнын ауыстырып жазып шығу керек.

15. Файлдан сөз тіркесін оқып, солардың ішіндегі ең ендісін анықтап, соның неше символдан тұратынын анықтау қажет.

16. Файлдан сандар оқып, солардың тақтарын бір файлға, жұптарын екінші файлға жазып шығыңдар.

17. Файлдан бірнеше сөйлем оқып, сол сөйлемдердің “м” әрпінен басталатын бір сөйлемін ғана экранға шығарып, соның ұзындығын анықтайтын программа құру керек.

18. Файлдан сандар оқып, солардың барлығын да бірге арттырып, шыққан сандарды басқа бір файлға жазып шығыңдар.

19. Файлдан бірнеше сөйлем оқып, сол сөйлемдердің ең соңғы сөйлемін ғана экранға шығарып, соның ұзындығын анықтайтын программа құру керек.

20. Файлдан сандар оқып, соларды керісінше тәртіппен екінші бір файлға жазып шығыңдар.

21. Файлдан сандар оқып, солардың цифрларын экранға сөзбен шығаратын программа құру керек, мысалы, 0 орнына “нөл”, 1 орнына “бір”, т.с.с. 9 орнына “тоғыз” деп жазатын болуы тиіс.

22. Файлдан сөз тіркесін оқып, солардың бірінші сөзі мен соңғы сөзін алмастырып, екінші бір файлға жазып шығыңдар.

23. Файлдан сөз тіркесін оқып, соларды керісінше жазып шығатын программа құру керек.

24. Файлдан сандар оқып, солардың алғашқы жартысы мен соңғы жартысының орнын ауыстырып, нәтижесін жаңа файлға жазып шығу керек.

25. 10 бүтін саннан тұратын файл жасайтын программа құру керек. Сол файлдағы сандарды оқып, олардың қосындысын анықтаңыздар.

26. While операторы арқылы Char типті элементтерден тұратын файл жасау қажет. Циклден шығу шарты – z әрпін енгізу. Сол файлдың көшірмесін екінші бір файлға жазып, жазылған мәліметтерді экранға да шығару керек.

27. Integer типті N саннан тұратын файл жасап, сол файлдағы жұп сандарды экранға шығару қажет.

28. Бүтін сандардан тұратын файл жасап, сол сандарды басқа бір файлға кері тәртіппен жазып шығу керек.

29. Мынадай құрылымдағы бірнеше қатарлары бар файл жасау керек:
- реттік нөмірі;
- фамилиясы, аты-жөні;
- жалақысы.

Осы файлға бес адам туралы мәліметтер енгізіп, басқа файлға осылардың ішіндегі ең көп жалақы алатын адам туралы мәліметті көшіріп жазу керек.

30. Файлдағы мәтінді түгел оқып шығып, ондағы “о” әрпін “а” әрпімен алмастыратын программа құрыңыздар.

31. Файлдағы мәтіндік ақпаратты экранға және қағазға шығаратын программа жасау керек.

32. Файлда N бүтін сан жазылған. Соларды өсуі бойынша реттеп, екінші файлға жазып шығару керек.

33. Нақты сандардан тұратын бір өлшемді жиым элементтерін пернелерден енгізе отырып, бір файлға жазып шығып, сол жиымның жұп индексті элементтерін екінші файлға жазатын программа құрыңыздар.

34. Топтағы студенттердің үлгерімін бір файлға мынадай түрде жазу керек: рет нөмірі, аты-жөні, 5 сабақ аты, әр сабақтан алған 3 бағасы. Программаға енгізілетін мәліметтер пернелерде теріліп, файлға жазылуы қажет. Жақсы оқитын студенттер тізімін екінші бір файлға бөлек жазып шығу керек.

11. ГРАФИКАЛЫҚ РЕЖИМДЕ ЖҰМЫС ІСТЕУ

Си тілінде растрлық графика жұмыс істейді, оның тақырыптық файлы **graph.h**. График нүктелерден – пиксельдерден тұрады. *Пиксель – экранның адрестелетін ең кіші элементі*. Алдымен графика шығара алатын бейнережимді іске қосу керек. Мұнда экранның пиксельмен берілген мөлшері және түстер саны беріледі.

Графикада үш координаталық жүйе: *абсолюттік*, *салыстырмалы* және *масштабталған* жүйе қолданылады. Тіке және көлденең өстер бойынша пиксельдер саны экран типіне байланысты болады.

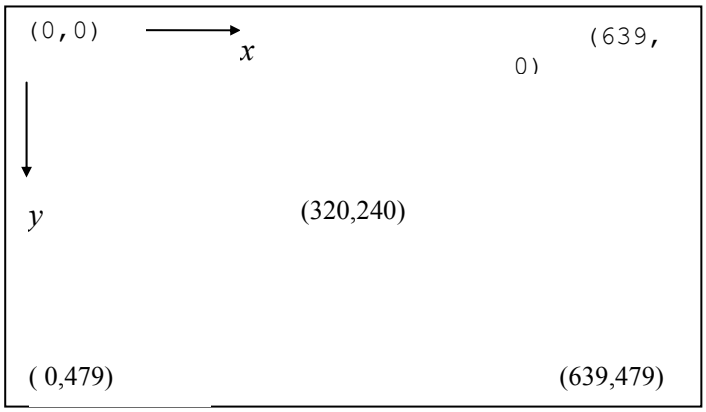
Абсолюттік координатада координаталар басы – (0;0) нүктесі сол жақ жоғарғы бұрышта болып саналады, *x* координатасы солдан оңға қарай, *y* координатасы жоғарыдан төмен қарай өседі.

Салыстырмалы режимде координаталар басы экранның кез келген нүктесіне ауыстырыла алады.

Масштабталатын режимде экран бетінде масштабталған координаталар беруге болады, онда *x* пен *y* өстері бойынша минимум және максимум мәндер енгізіп, жұмыс істеуге мүмкіндік бар.

Жалпы абсолюттік графикалық режимде

- әрбір пиксель берілген 16 түстің біріне боялады;
- (0,0) – экранның сол жақ жоғарғы бұрышы координатасы;
- (639,479) – оң жақ төменгі бұрышы координатасы болады;
- әр түрлі фигураларды экранға шығару үшін алдын ала графикалық режимді іске қосып алу керек.



Экрандағы нүктелер координаталары

Жалпы дисплей адаптерлері графикалық режимде 200, 350, 600 нүктелерден тұратын экран жолдарының әрқайсысында 640, 720, 800 нүктелер тізбегін бейнелей алады. Мұндағы нүкте деп отырғанымыз – көлемі 0,8x1 мм^2 шамасында болып келген (CGA) кішкентай тіктөрбұрыш, яғни пиксель.

Әрбір нүктенің координаталары екі бүтін санмен (X, У) анықталады. Дисплей экранына график салу үшін оның нүктелерінің координаталарын көрсету қажет. Координаталар басы (0,0) болып экранның сол жақ жоғарғы бұрышы есептеледі. *X* координаталары (бағаналар немесе позициялар нөмірлері) солдан оңға қарай, ал *У* мәндері (жолдар немесе қатарлар) жоғарыдан төмен қарай өсіп отырады. Мысалы, VGA адаптерінің экран бұрыштарының координаталарын *X*=0..799, *Y*=0..599 аралығында көрсету қажет. Экранда *X* өсі солдан оңға қарай, *Y* өсі жоғарыдан төмен қарай бағытталған, ал оның шеткі нүктелерінің координаталары суретте көрсетілген.

Олардың ең жоғарғы мәндері пайдаланылған экран адаптеріне тәуелді болады, яғни (0,0)..(320x200), (0,0)..(640x480), (0,0)..(800x600) аралықтарында және т.б. болуы мүмкін. Сонымен, графикалық режимде экрандағы кез келген объект көрініп тұрған нүктелер тобынан тұрады. Мәтіндік режимнен графикалық режимге көшкенде экран тазартылады. Графикалық режимде экраннан курсор көрінбейді. Дисплей экранының бетіне (кейін қағазға) нүкте, түзу немесе қисық сызық, шеңбер, эллипс және кез келген тұйық сызық сызып шығаруға болады. Сонымен қатар тұйық сызықтардың ішін әр түрлі түске бояп қою мүмкіндіктері де бар. Сызықтарды жылжыту, айналдыру және басқа орынға көшіру арқылы көрнекі бейнелер мен мультфильмдер жасауға болады.

График тұрғызу үшін оны шығару немесе бастау нүктесін көрсету қажет. Мәтіндік режимде ол курсор позициясы болып саналады, ал графикалық режимде көрініп тұратын курсор жоқ, бірақ экранда көрінбейтін курсор тәрізді сілтеме белгі CP (current pointer) бар. Негізінде оны да курсор деп қабылдауға болады.

Графикалық режимдегі жұмыстарды атқаратын Турбо Си нұсқасында бірсыпыра графикалық функциялар бар, енді біз солардың негізгілеріне тоқталып өтеміз.

Графикалық режимде жұмыс істеу былай басталады:

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>

void main ( )

{
  int gdriver = DETECT; //драйвер
```

```
  int gmode;              //режим аты
   int errorcode;      // қате коды
   initgraph (&gdriver,&mode,"C:\\TC\\bgi\\");
    errorcode = graphresult();
    if errorcode != grOk) //іске қосу қатесі
    {printf("Қате: %d\n", errorcode);
     puts("Аяқтау үшін УТЕУК басыңыз");
     getch ();
     return;
    }
     ... Ары қарай программа мәтіні...
  getch();
  closegraph ( );
}
```

11.1. Графикалық режим орнату, одан шығу, мәтін жазу, сызық салу функциялары

Графикалық режим драйверін іске қосу функциясы

```
initgraph(&Driver,&Mode,Path);
```

Driver параметрі бейнелік жүйе драйверін анықтайды, *Mode* параметрі – бейнелік жүйе жұмыс режимін береді, ал *Path* параметрі – драйвер файлының орнын көрсетеді. Көбінесе *Driver* параметрі мәні ретінде *detect* болатын бүтін константа қолданылады. Мұндайда initgraph функциясының өзі графикалық драйвер типін анықтап, ең дұрыс режимді таңдап алады. Тақырыптық файлы: <graphics.h>

Драйвер дұрыс оқылған соң, *initgraph()* функциясы 4 К көлемінде (келісім бойынша) ішкі графикалық буфер ұйымдастырады да, экран түсі, сызықтар түсі анықталып, дисплей адаптері графикалық режимге кіреді. Экран тазаланып, курсор сол жақ жоғарғы бұрышқа орнатылады.

Егер BGI-файлдар ағымдағы бумада болса, *initgraph()* функциясының үшінші параметрі ретінде бос орын мәнін беруге болады

```
initgraph(&Driver,&Mode," ");
```

Графикалық режимнен шығу және оған бөлініп берілген жады бөлігін босату үшін, яғни бейнеадаптер буферін тазартып, бұған дейінгі мәтіндік режимді қалпына келтіру мақсатында мына функция қолданылады

```
closegraph();
```

Графикалық режимде сызық түстерін, тұйық сызықтар ішін түрлі түске бояуға болады. Ол үшін мәтіндік режимдегідей түстер кодтары және олардың ағылшынша атаулары қолданылады.

Графикалық режимнен уақытша мәтіндік режимге ауысу мақсатында мынадай функция қарастырылған

```
restorecrtmode();
```

Ол *initgraph()* функциясын пайдаланғанға дейінгі болған мәтіндік режимді қайта орнатады; буферде (экран көрінісі де) сақталған мәтін қалпына келмейді, өйткені ол *initgraph()*функциясы арқылы өшірілген болатын.

Графикалық режимге қайтып оралу мына функция арқылы атқарылады

```
setgraphmode(gm);
```

функция аргументі ретінде қолданылған драйверге қатысты режимнің бүтін сан түріндегі нөмірі қарастырылады.

Курсор орнын анықтау функциялары

```
getx();  gety();
```

Функциялары курсор тұрған орын координаталарын *x (y)* береді.

```
maxx(); getmaxy();
```

getmaxx() функциясы экранның оң жақтағы ең шеткі нүктесінің *x* координатасын анықтайды, ал getmaxy() функциясы — экранның ең төменгі *y* нүктесі координатасын анықтайды.

Сызықтар түсін тағайындау функциясы

```
setcolor(col);
```

мұндағы **col** – түс атауы немесе түс коды. Сызық және мәтіндер түсі осы функциямен беріледі. Ал сызық типі **setlinestyle()** функциясы арқылы тағайындалады.

Түстер бүтін сан түріндегі кодпен немесе константа түрінде бас әріппен жазылатын ағылшынша түс атауларымен беріледі.

Түстердің стандартты белгіленулері төмендегідей:

Көмескі түстер	*Кодтары*	*Ашық түстер*	**Кодтары**
Қара (BLACK)	0	Қара қошқыл (DARKGRAY)	8
Көк (BLUE)	1	Көкшіл (LIGHTBLUE)	9
Жасыл (GREEN)	2	Ақ жасыл (LIGHTGREEN)	10
Көгілдір (CYAN)	3	Ақшыл көк (LIGHTCYAN)	11
Қызыл (RED)	4	Қызғылт (LIGHTRED)	12
Күлгін (MAGENTA)	5	Қызғыш (LIGHTMAGENTA)	13
Қоңыр (BROWN)	6	Сары (YELLOW)	14
Сұр (LIGHTGRAY)	7	Ақ (WHITE)	15

Мысалы: **setcolor(YELLOW); setcolor(3); setcolor(5);**

Экранның фоны түсін өзгерту функциясы

```
setbkcolor (түсі);
```

Мысалы: **`setbkcolor(BLUE);`** **`setbkcolor(14);`**

Мәтін шығару функциялары

Графикалық режимде экранға мәтін шығарарда символдар көлемін, бағытын (тік, көлденең), бекітілген бірнеше қаріп түрінің бірін таңдау мүмкіндіктері бар. Осындай **символдар параметрлері мынадай функция** арқылы беріледі:

```
settextstyle(Қаріп,Бағыты,Көлемі);
```

outtextxy және outtext функциялары арқылы шығарылатын мәтіндердің қаріп түрін, көлемін және бағытын тағайындайды. **`Қаріп`** *(шрифт)* параметрі ретінде төмендегі константалардың бірін пайдалануға болады.

Константа	Мәні	Қаріп типі
DEFAULT_FONT	0	Стандартты. Әрбір символ 8 х 8 пиксел көлемдегі квадрат ішіне шығарылады.
TRIPLEX_FONT	1	Triplex шрифті (TRIP.CHR файлы)
SMALL_FONT	2	Майда шрифт (LITT.CHR файлы)
SANSSERIF_FONT	3	SansSerif тік шрифті (SANS.CHR файлы)
GOTHIC_FONT	4	Готикалық шрифт (GOTH.CHR файлы)

Орыс шрифтері тек стандартты түрде ғана (default_font) шығарылады. **`Бағыты`** параметрі берілген мәтінді тік немесе көлденең беруді тағайындайды, оның мәндері:

HORIZ_DIR	0	Солдан оңға қарай
VERT_DIR	1	Төменнен жоғары қарай

`Көлемі` 1-ден 10-ға дейін өзгере алады. Бұл шама DEFAULT_FONT стандартты қарпі (8х8 пиксел) үшін әр символды неше есе үлкейту керек екендігін көрсетеді (стандартты матрица 8*8 болғандықтан, егер көлемі 4 болса, онда символдар 32*32 пикселдік матрицаға дейін үлкейеді). Ал қалған қаріптер үшін бұл параметр сызықтық емес, экспоненциалдық масштабтау шкаласын көрсетеді. Символдың негізгі нұсқасы көлемі 4-ке тең болып саналады, сондықтан ол 7 болса, онда символдар 2 есе өседі; егер 8 болса - 3 есе; ал 9 болса - онда 4 есе ұлғаяды. Символдар әрқашанда үздіксіз жіңішке сызықтармен жазылады.

Экранға мәтін шығару функциясы

```
outtext ("Мәтін");
```

қостырнақшаға алынған мәтінді курсор тұрған орыннан бастап экранға шығарады. Мәтін ішінде басқару символдары болмауы тиіс, мысалы, \n.

Шығарылатын символдар түсі setcolor, шрифт типі — settextstyle функцияларымен беріледі. Мысалы:

```
setcolor(4); moveto(250,10);outtext("Омаров Марат");
```

Экрандағы көрсетілген орынға мәтін шығару функциясы

```
outtextxy(x,y,"мәтін");
```

Курсорды алдымен x, y нүктесіне орналастырып алып барып, мәтінді экранға шығарады. Мәтін шығарылған соң, курсор бұрынғы орнында **(x,y)** қалады.

Шығарылатын символдар түсін – setcolor(), қаріп типін – settextstyle() функциясымен беруге болады.

Мысалы:

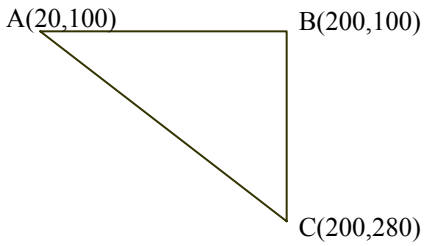
```
setcolor(4); outtextxy(250,2,"Омарова Айман");
```

Компьютер бейнежадына пиксел жазу функциясы

```
putpixel(x,y,Түсі);
```

координатасы (**x, y**) пиксел-нүктені ***Түсі*** санына байланысты бояп шығады. ***Түсі*** ағылшынша сөзбен немесе кодпен беріледі. Бұл функцияны цикл ішіне орнатып, сызықтар сызуға болады. Мысалы:

```
setcolor(BLUE);
for (x=20; x<=200; x++)
   putpixel(x,100, BLUE);
for (y=100; y<=280;y++)
   putpixel(200,y, BLUE);
line(200,280,20,100);
```



Түрлі сызықтар салу үш функция арқылы орындалады

```
1) line (x1,y1,x2,y2);
```

x1,y1 нүктесінен *x2,y2* нүктесіне дейін сызық сызады, бірақ мұнда курсор орны өзгермейді.

```
2) lineto (x,y);
```

Курсор тұрған нүктеден **x,y** нүктесіне дейін сызық салады, мұнда курсор бұрынғы орнынан жаңа орынға ауысады.

3) `linerel (dx,dy);`

Курсор тұрған **x,y** нүктесінен **x+dx,y+dy** нүктесіне дейін сызық салады, яғни сызық координаталары салыстырмалы түрде беріледі. Мұнда да курсор бұрынғы орнынан жаңа орынға ауысады.

Сызықтар осыған дейінгі *setcolor()* функциясы тағайындаған түспен және *setlinestyle()* функциясы арқылы орнатылған стильмен сызылады.

Мысалы, ромб сызудың екі тәсілін келтірейік.

1 мәсіл

```
setcolor (GREEN);
moveto(100,10); lineto(50,90);  lineto(100,170);
setcolor(RED);  lineto(150,90); lineto(100,10);
```

2 мәсіл

```
setbkcolor (WHITE); setcolor (GREEN);
line(100,10, 50,90); line(50,90, 100,170);
setbkcolor (RED);
line(100,170,150,90); line(150,90, 100,10);
```

11.2. Сызық стильдерін беру

Геометриялық фигуралар сызықтарының қалыңдығы мен түр сипатын беру функциясы

```
setlinestyle(tip,obr,tol);
```

сызықтар стилін береді, мұндағы **tip** – алдын ала анықталған константа, ол сызық ти-пін береді. **tol** – сызық қалыңдығын анықтайтын константа (**NORM_WIDTH** – қалыпты, **THICK_WIDTH** – қалыңырақ). Егер программалаушы анықтаған сызық типі қолданылатын болса, онда **obr** мәні төрт таңбалы оналтылық константа болып, ол ұзындығы 16 пиксел сызық кесіндісін кодтау үлгісі болуы тиіс.

tip константасының мүмкін мәндері (сызықтық элементтер үшін)

0	SOLID_LINE	(жай сызық)
1	DOTTED_LINE	(пунктир сызық – нүктелерден тұрады)
2	CENTER_LINE	(штрих-пунктир сызық – нүктелер мен сызықшалардан тұрады)
3	DASHED_LINE	(DOTTED_LINE сызығынан ұзыншалау келген пунктир сызық)
4	USERBIT_LINE	(программалаушы анықтаған сызық)

tol – *қалыңдығы* параметрінің мүмкін мәндері

1	NORM_WIDTH	(қалыңдығы бір пиксел)
3	THICK_WIDTH	(қалыңдығы үш пиксел)

obr параметрі **tip** константасы мәндері 4-ке тең болғанда ғана жұмыс істейді (қалған жағдайларда ол қарастырылмайды, сондықтан оны 0 деп алуға болады). Оның көмегімен кез келген периодты түрде қайталанатын сызық түрін бере аламыз, оның периоды 16 пиксел болуы тиіс. Егер сызықта жарықтанатын пиксел керек болса, үлгіде 1-ге тең бит, болмаса – 0-ге тең бит алу керек. Сонымен, **obr** мәні төрт таңбалы

оналтылық константа болып, ол ұзындығы 16 пиксел сызық кесіндісін кодтау үлгісі болуы тиіс.

Мысалы, шаблон для пунктир сызық үшін алынған үлгі мынадай болуы мүмкін: 0x3333, бұл мынадай биттер тізбегіне сәйкес келеді 0011 0011 0011 0011.

Графикалық режимде экрандағы тұйық сызық іші белгілі бір түспен боялуы мүмкін.

Штрихтау сызықтарын тағайындау және тұйық аймақты бояу функциясы

```
setfillstyle (stil,col);
```

аймақты бояу, толтыру стилін береді, мұндағы **stil** – алдын ала мәні анықталған константа, стильді береді, оның мәндері төменгі кестеде көрсетілген; **col** – түс кодына сәйкес бүтін сан немесе ағылшынша түс аты; ол **setcolor** функциясын анықтауда көрсетілген.

Бояу тәсілін анықтайтын **stil** параметрінің мүмкін мәндері

EMPTY_F1LL	0	аймақты фон түсімен толтыру
SOLID_FILL	1	көрсетілген түспен толтыру
LINE_F1LL	2	көлденең штрих сызықтармен толтыру
LTSLASH_FILL	3	45 градус оңға қиғаш жіңішке штрих сызықтармен толтыру
SLASH_FILL	4	45 градус оңға қиғаш қалың штрих сызықтармен толтыру
BKSLASH_FILL	5	45 градус солға қиғаш жіңішке штрих сызықтармен толтыру
LTBKSLASH_FJLL	6	45 градус солға қиғаш қалың штрих сызықтармен толтыру
HATCH_FILL	7	торсызықты штрихтармен толтыру
XHATCH_FILL	8	45 градус оңға қиғаш сирек штрих сызықтармен толтыру
INTERLEAVE_FILL	9	45 градус қиғаш жиі штрих торсызықтармен толтыру
WIDE DOT_FILL	10	сирек нүктелермен толтыру
CLOSE_DOT_FILL	11	жиі нүктелермен толтыру

Сызық стилі мен түсі көптеген функцияларда (**bar, bar3d, sector**, т.б.) қолданылады. Енді бір мысал келтірейік.

```
// Әр түрлі сызықтар салу мысалы
```

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
main ()
{
int gdriver=DETECT;
int gmode;
int x,y,xk=500;
initgraph (&gdriver,&gmode,"C:\\TC\\bgi");
setcolor(RED);  x=80; y=30;
moveto(x,y);   outtext("жай сызық - SOLID_LINE");
setlinestyle (SOLID_LINE,0,NORM_WIDTH);  y+=20;
line(x,y,xk,y);
y+=40; moveto(x,y);
outtext("пунктир сызық – DOTTED_LINE");
setlinestyle (DOTTED_LINE,0,NORM_WIDTH);
y+=20;  line(x,y,xk,y);
y+=40;  moveto(x,y);
outtext("штрих-пунктир сызық – CENTER_LINE");
setlinestyle (CENTER_LINE,0,NORM_WIDTH);
y+=20;  line(x,y,xk,y);
y+=40;  moveto(x,y);
outtext("штрихі ұзынша пунктир сызық");
setlinestyle (DASHED_LINE,0,NORM_WIDTH);
y+=20;  line(x,y,xk,y);
y+=40;  moveto(x,y);
outtext("қалың сызық");
setlinestyle (SOLID_LINE,0,THICK_WIDTH);
y+=20;  line(x,y,xk,y);
getch();
closegraph();
}
```

11.3. Тұйық сызықтар салу

Тіктөрбұрыш контурын салу үшін төмендегі функция

```
rectangle(xl,yl,x2,y2);
```

қолданылады, ол сол жақ жоғарғы бұрышы - *x1,y1,* оң жақ төменгі бұрышы - *x2,y2* болып келетін төртбұрыш сызады. Сызық типі setlinestyle функциясымен, ал түсі – setcolor функциясымен анықталады.

Іші боялған тіктөрбұрыш салу үшін

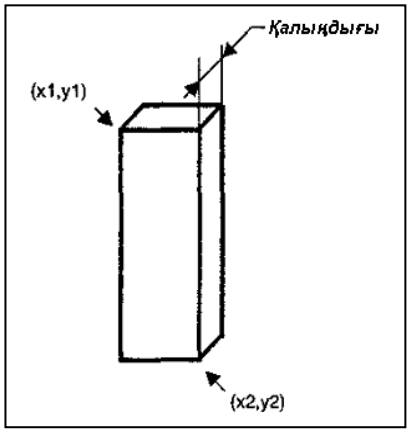
```
bar(xl,yl, x2,y2);
```

функциясы пайдаланылады, *x1, y1* сол жақ жоғарғы бұрыштың, ал *x2 , y2* – оң жақ төменгі бұрыштың координаталары. Мұның қабырғалары өстерге параллель болады, контуры көрсетілмейді. Оны бояу стилі *setfillstyle()* функциясымен анықталады.

Алдыңғы жақ беті боялған параллелепипед салу функциясы

bar3d(*x1,y1, x2,y2, Қалыңдығы, Ж_Беті*);

сыртқы контуры сызылған паралелепипед салады, мұндағы *x1,y1* параметрлері сол жақ жоғарғы бұрыштың, ал *x2,y2* – оң жақ төменгі бұрыш координаталары. *Қалыңдығы* параметрі – алдыңғы және артқы жақтарының ара қашықтығы, егер ол 0 болса, онда айналасы сызылған төртбұрыш салынады. *Ж_Беті* параметрі – жоғарғы жақ шеттерін сызуды (1 – TOP_ON) көрсетеді. Егер ол 0 (TOP_OFF) болса, сызықтар сызылмайды (бірінің үстіне бірі қойылған бірнеше паралелепипед салу кезінде қажет болады).



Мысалы, бірінің үстіне бірі қойылған бірнеше паралелепипед салу:

```
setbkcolor (WHITE);
setcolor (GREEN);
bar3d(230,50,250,150,15,1);
bar3d(220,150,260,180,15,1);
bar3d (300, 150, 340, 180, 15, 0);
bar3d (300, 50, 340, 150, 15, 1);
```

Көпбұрыш салу функциясы

drawpoly(НүктелерСаны,Координаталары);

түзу сызықтардан тұратын тұйық аймақты көпбұрыш сызады. *НүктелерСаны* параметрі көпбұрыш төбелері санын, *Координаталары* параметрі сол төбелер координаталарын жиым элементтері ретінде береді. Жиымның 0-ші және 1-ші элементтері алғашқы нүкте координаталары, 2-ші, 3-ші элементтері – екінші нүкте координаталары, т.с.с. Енді осы функцияны пайдаланып, алты бұрыш салайық.

```
// көпбұрыш салу
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
main ()
```

```
{
int gdriver=DETECT;
int gmode;
initgraph (&gdriver,&gmode,"C:\\TC\\bgi");
int x,y,t[14]={450,150, 500,350, 400,400,
               150,400, 50,150, 250,80, 450,150};
setcolor(WHITE);
drawpoly(7, t);
settextstyle(3,HORIZ_DIR,1);
outtextxy(458,135,"C(450,150)");
outtextxy(508,335,"D(500,350)");
outtextxy(400,400,"E(400,400)");
outtextxy(150,400,"F(150,400)");
outtextxy(38,138,"A   (50,150)");
outtextxy(243,55,"B(250,80)");
getch();
closegraph();
}
```

Іші боялған көпбұрыш салу үшін қолданылатын функция

```
fillpoly (НүктелерСаны,координаталары);
```

мұндағы **НүктелерСаны** – төбелер саны, **координаталары** – жиым элементтері түрінде берілген төбе координаталары. Әрбір төбе координатасы екі бүтін санмен бері-леді. Бұл функция төбелердің алғашқы нүктесін соңғы нүктесімен қосып, сызықтарды тұйықтап, ішін бояйды. Сызық типі мен бояу түсі *setfillstyle()* және *setfillpattern()* функцияларымен анықталады.

Шеңбер сызу функциясы

```
circle(x,y,r);
```

радиусы *r* (бүтін сан), центрінің координаталары (x, y) болатын шеңбер сызады.

Сызық түсі setcolor () функциясымен беріледі. Мысалы, қызыл түсті 5 шеңбер сызайық:

```
...
setcolor(RED);
for (r=5;r<=25; r+=5)
  circle(320,240,r); ...
```

Доға сызу функциясы

```
arc(x,y,БұрышБасы,БұрышСоңы,Радиус);
```

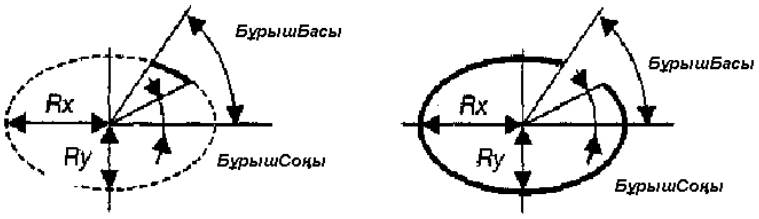
центрінің координатасы *(x, y)*, *радиусы* берілген доға сызады. **БұрышБасы,*БұрышСоңы*** параметрлері бұрышты градуспен горизонталь

x өсінен бастап, сағат тіліне қарсы бағытта береді. `Радиус` параметрі доға радиусын бүтін санмен береді. Бұрыштар мәні периодына сәйкес эквивалентті түрде [0..360] интервалындағы мәндерге келтіріледі. Мысалы, **`arc(x,y,-45,45,r)`** және **`arc(x,y,675,-315,r)`** шеңбердің бір ширегіне сәйкес бір доғаның екі түрде берілуін көрсетеді.

Эллипс доғасын сызу функциясы

`ellipse(x,y, БұрышБасы,БұрышСоңы,rx,ry);`

центрінің координаталары *(x, y)* эллипс немесе эллипс доғасын сызады. **`БұрышБасы,БұрышСоңы`** параметрлері доғаның басы мен соңын градуспен сағат тіліне қарсы береді. **`rx, ry`** параметрлері эллипстің көлденең және тік радиустарын береді. Эллипс өстері координаталар өстеріне параллель болады.



Іші боялған контурлы эллипс салу функциясы

`fillellipse (x,y, rx,ry);`

мұндағы *x, y* – центр координаталары; *rx, ry* – пикселмен берілген эллипс жарты өстері радиустары. Эллипс өстері координат өстеріне параллель болады. Эллипс ағымдағы түспен боялып шығады.

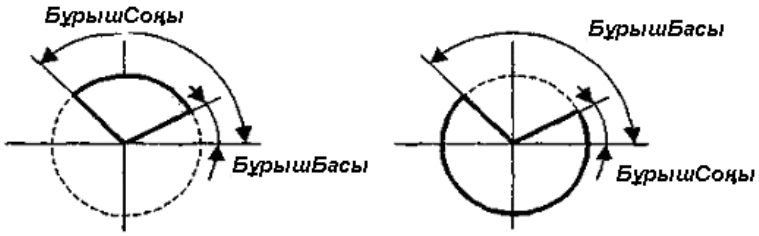
Іші боялған дөңгелек сектор сызу функциясы

`pieslice(x,y,БұрышБасы,БұрышСоңы,Радиус);`

радиусы *Радиус,* центрі **`(x,y)`** нүктесіндегі дөңгелек сектор сызады. **`БұрышБасы, БұрышСоңы`** параметрлері шеңбер секторының бастапқы және соңғы бұрыштарын градуспен сағат тіліне қарсы анықтайды. Егер **`БұрышБасы`** = 0, ал **`БұрышСоңы`** = 360 болса, онда **`pieslice`** функциясы шеңбер сызып шығады. Бұрыштарды [0..360] шегіне (диапазонына) келтірген соң, сектор мәні кіші бұрыштан мәні үлкен бұрышқа қарай сызылады, сол себепті OX өсінің оң жақтағы бағытын кесіп өтетін сектор салуға болмайды. Сектор контуры (доға мен екі радиус) рисуется после закраски сектор боялған соң сызылады, ал сызық

типі мен қалыңдығы *setlinestyle()* функциясымен анықталады. Егер контурсыз сектор салу керек болса, мынадай тәсілді пайдалануға болады:

```
setcolor (BLACK); setbkcolor (BLUE) ;
setwritemode (XOR_PUT); setfillstyle (WIDE_DOT_FILL,
RED);
pieslice(200,100,45,90,50);
```



Іші боялған контурлы эллипс секторын салатын функция

```
sector (x,y,бұрыш_басы,бұрыш_соңы,rx,ry);
```

Бұл функция *pieslice()* функциясы тәрізді жұмыс істейді.

```
// Жазуы бар секторлар sector2.cpp
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <graphics.h>
void main ()
{
   int gd=DETECT,gm,i,x,y;
   initgraph (&gd,&gm,"c:\\TC\\bgi");
   setcolor(BLACK); setbkcolor(BLUE);
   setwritemode(XOR_PUT);
   setfillstyle(WIDE_DOT_FILL,RED);
   pieslice(200,100,45,90,50);
   setbkcolor(BLUE); setcolor(RED);
   setfillstyle(1,3);
   x=getmaxx()/2; y=getmaxy()/2;
   pieslice(x,y,270,360,100);
   setfillstyle(1,2);
   pieslice(x,y,0,270,100);
   settextstyle(1,0,2);
   moveto(x-20,y-40); outtext("75%");
   moveto(x+20,y+20); outtext("25%");
   getch();
   closegraph();
```

```
}
```

Тұйық сызықпен қоршалған аймақтың ішін бояу функциясы

```
floodfill (x,y, шекара);
```

мұндағы *x, y* – боялатын аймақ ішіндегі нүкте координатасы. Тұйық аймақты қоршаған сызық контурында тесік болмауы тиіс, әйтпесе бояу бүкіл экранды сол түске бояп жібереді. Контур түсі *шекара* түсімен бірдей болуы тиіс. Бояу түсі мен типі *setfillstyle()* функциясымен орнатылады. Енді бір мысал келтірейік.

```
//боялған шеңберлер, эллипстер салу – kr_krug.cpp
#include <conio.h>
#include <stdlib.h>
#include <graphics.h>
void main ( )
{
   int gd=DETECT,gm,r,x=120,y=240;
   initgraph(&gd, &gm,"C:\\TC\\BGI");
   setcolor(RED); //сызықтар жасыл түсті
   setbkcolor(BLUE); //фон көк түсті
   for(r=0; r<80; r++)//концентрлі
     circle(x,y,r); //80 қызыл шеңбер салу
   setfillstyle(SOLID_FILL,RED);
   x+=200;  circle(x,y,80);  //қызыл шеңбер
   floodfill(x,y,RED); // іші де қызыл
   x+=200;  fillellipse(x,y-150,80,50);//қызыл эллипс
   setfillstyle(SOLID_FILL,GREEN);
   fillellipse(x,y,80,50); //жасыл эллипс
   setfillstyle(SOLID_FILL,YELLOW);
   fillellipse(x,y+150,80,50); //сары эллипс
   getch();
   closegraph();
}
```

Терезе ашу функциясы

Графикалық режимде экран ішінде өз координаталық жүйесі бар төртбұрышты басқа терезе ашу қолданылатын функция

```
setviewport (x1, y1, x2, y2, clip);
```

мұндағы ***x1, y1*** – терезенің сол жақ жоғарғы бұрышы координаталары; ***x2, y2*** – оң жақ төменгі бұрышы координаталары; ***clip*** – қию параметрі. Егер ***clip*** параметрі 1 болса, онда терезеге сыймайтын бейне элементтері қиылып алынып тасталады; ал егерде ол – 0 болса, терезе шекаралары есепке алынбай,

бейне толық экранда көрсетіледі. Бұл функция дұрыс орындалса, графикалық курсор терезенің координаталар басына орналасады.

Терезені тазалау функциясы

```
clearviewport ();
```

Тағы бір мысал келтірейік.

```
//Диагональ сызу,жаңа терезелер ашу primer3.cpp
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <graphics.h>
void main ()
{
int gd=DETECT,gm,x,y;
initgraph (&gd,&gm,"c:\\TC\\bgi");
//жасыл экранда қалыңдығы 3 пиксел диагональ сызу
setbkcolor(GREEN) ; setcolor(RED);
setlinestyle(SOLID_LINE,0,3);
x=getmaxx(); y=getmaxy();
printf("       xmax=%d ymax=%d ",x,y);
line(0,0,x,y); // бір перне басып,экран тазалау
getch();
cleardevice(); // терезе ашу, оны тазалау
setviewport(100,100,600,400,1);
clearviewport();  setcolor(GREEN);
rectangle(10,30,450,255);
circle(230,145,55);
settextstyle(DEFAULT_FONT,HORIZ_DIR,3);
//орыс әріптері тек DEFAULT_FONT шрифтінде
setbkcolor(BLUE);
setcolor(WHITE);
//Мәтін терезенің сол жақ жоғарысында
outtextxy(1,1,"Жаңа терезе");
getch();
closegraph();
}
```

Бақылау сұрақтары

1. *IBM PC компьютерлері бейнемониторының қандай түрлері бар? Олардың айырмашылығы неде?*
2. *Графикалық режимнің атқаратын қызметі қандай? Оны программада қалай іске қосады?*
3. *Графикалық режим қандай функция арқылы орнатылады?*

4. *Адаптерлердің мынадай типтерінің CGA, EGA, VGA бір-бірінен айырмашылығы неде?*
5. *Монитордың мәтіндік және графикалық режимдерінің мүмкіндіктері неге әртүрлі болады?*
6. *Драйвер деп нені айтады? Графикалық драйвер ше? Ол қандай қызмет атқарады?*
7. *Экранның түсін және оған шығарылатын жол символдарының түсін қалай өзгертуге болады?*
8. *Графикалық режимде курсор бола ма?*
9. *Символдарды қалай жыпылықтатып қоюға болады?*
10. *Курсорды экранның кез келген нүктесіне қалай көшіреміз?*
11. *Графикалық режимде экранда қалай терезе құруға болады?*
12. *Графикалық режимде экранды қалай тазартуға болады?*
13. *Clrscr, Clreol, Delline және Insline функцияларының ортақ қасиеттері мен айырмашылықтары.*

ТАПСЫРМАЛАР

1. Уақытты есептеп отыратын сиқыршымен кездестіңдер делік және ол сендерге жұлдыздар сырын ашып береді деп ойлайық. Неге аспанда жұлдыздар жымыңдайды? Аспандағы әрбір жұлдыз — бір адамның өмірі. Сендерге көне жұлдыздарды дәл уақытында өшіріп, жаңасын дер кезінде жағып отыру керек. Уақыт есептеушінің жұмысын көрсететін программа жазу керек. Есеп модель құрудан басталады: экрандағы әрбір жұлдыз нүктемен көрсетіледі және барлығы 20 жұлдыз жанып тұрсын делік (кездейсоқ түрде таңдап алынған): жанып тұрған бірінші нүктені өшіріп, оны кез келген басқа орыннан жандыру керек, содан кейін екіншісін, т.с.с. Жиырмасыншы жұлдыз өшкеннен кейін қайтадан біріншісіне көшу керек.

2. “Жаңбыр“ программасын жазыңдар — экранда (кездейсоқ күйде) 50 нүкте жоғарыдан төменге қарай қозғалып келе жатқандай болуы тиіс. Нүктелердің бірі экранның төменгі шекарасына жеткенде, ол қайтадан экранның жоғарғы бөлігінде пайда болуы керек.

3. “Жаңбыр” программасын кейбір нүктелер шапшаң, кейбіреулері баяу қозғалатындай етіп өзгертіңдер.

4. “Жаңбыр” программасын нүктелер өз қозғалысын төменгі сол жақ бұрыштан бастап, жоғарғы оң жақ бұрышта аяқталатындай етіп өзгертіңдер.

5. Қағазға оралған “кәмпиттің” бейнесін: диагональдары жүргізілген горизонталь тіктөртбұрыш пен оның екі жағына жалғасып жатқан теңқабырғалы үшбұрыштар салыңдар. Тіктөртбұрыштың сол жақ төменгі төбесінің координатасы (x, y), биіктігі a-ға, ұзындығы $2a$-ға тең болсын. Үшбұрыштың биіктіктері $a/2$. Тікбұрыштың диагональдары мен қабырғаларынан пайда болған қарама-қарсы екі үшбұрышты бояңдар.

6. Үш тісті және жарты дөңгелек пішінді аркасы бар мұнара салыңдар. Мұнараның сол жақ төменгі төбесінің координатасы (*x*, *y*), табанының ұзындығы *a*. Қалған өлшемдерін *a* арқылы өрнектеп табыңдар. Мұнараны бояңдар.

7. Куб салыңдар. Кубтың алдыңғы сол жақ төменгі төбесінің координатасы (*x*, *y*), бүйір қыры *a*. Жоғарғы бүйір бетін бояңдар, оңжақ бүйір бетінде диагональ жүргізіңдер.

8. Алтыбұрышты тік пирамида салыңдар. Табанының алдыңғы сол жақ төбесінің координатасы (*x*, *y*), табанындағы қыры *a*. Қалған өлшемдерін *a* арқылы өрнектеп табыңдар. Пирамиданың екі бүйір жақтарын бояңдар.

9. Тіктөртбұрышты қиық пирамида салыңдар. Пирамида табанының төменгі сол жақ төбесінің координатасы (*x*,*y*), табанындағы қыры *a*, жоғарғы жақтың қыры — *a*/2. Қалған өлшемдерін *a* арқылы өрнектеп табыңдар. Пирамиданың оңжақ бүйір бетін бояңдар.

10. Қарайтылған “Е” әрпі тәрізді фигура салатын программа құрыңдар. Әріп құрайтын нүктелердің абсциссалары мен ординаталары жиым элементтері ретінде берілген. Сурет салу үшін циклге енгізілетін тек қана бір LINE операторын пайдаланыңдар.

11. Экранда боялған шырша салатын программа жазыңдар. Сурет элементтерін жиымда сақтап, өшіріп және содан кейін қайтадан экранға шығарыңдар.

12. Боялған 7 шыршаның суретін салатын программа жазыңдар. Шыршалар бір-біріне ұқсас, горизонталь бойында бір қалыпта орналасады, шыршалардың биіктігі солдан оңға қарай сызықтық түрде өсуі керек. Шыршаларды бойлап Қызыл Телпек оларды уақытша “үсімесін” деп жауып шығатын болсын.

13. Графикалық режімде ESC пернесін басқанша монитор экранына кездейсоқ күйде нүктелерді шығаратындай программа құрыңдар.

14. Графикалық режімде ESC пернесін басқанша монитор экранында (100, 100) — (300, 200) тіктөртбұрышының нүктелерін кездейсоқ күйде жоятындай программа құрыңдар.

15. Графикалық режімде ESC пернесін басқанша монитор экранының центрі (200, 200) нүктесімен дәл және радиусы 80 болатын дөңгелектің нүктелерін кездейсоқ күйде жоятындай программа құрыңдар.

16. Шахмат тақтасының бейнесін салыңдар.

17. Экранда қарайтылған **М** әрпі пішіндес фигура салатын және бағыттауыш тілсызық (→, ←, ↑, ↓) пернелер арқылы басқарылып,

фигураның көлемін + және – пернелерінің көмегімен өзгертетін программа құрыңдар.

18. Циклдік операторды пайдаланып 10 басқышы бар саты салыңдар, оның жұп және тақ нөмірлі басқыштарының көлемі әр түрлі болатын программа құрыңдар.

19. Шеңберге іштей бесбұрышты жұлдыз салыңдар.

20. Қызыл экранға 0..9 аралығындағы 1000 кездейсоқ санды ақ түспен шығару қажет. Мұнан соң экран түсін жасылға бояп, сандарды сары түспен шығарыңыздар.

21. Экранды барлық (8) фон түстеріне 5 секунд кідіріспен бояп шығыңыздар. Экранның сол жақ жоғарғы бұрышына оның нөмірін жазып қою керек.

22. Қара экранға 1-ден 16-ға дейінгі сандарды 16 түрлі түске бояп шығару қажет. Әрбір сан бөлек жолда орналасатын болсын.

23. Алдын ала тазартылған экранды ақ түске бояп, жеті атаңыздың аттарын жеті түрлі түспен шығарыңыздар.

24. Қара экранға өзгеріп отыратын кездейсоқ түстер арқылы 200 “+” белгісін кездейсоқ түрде берілетін координаталық нүктелерге басып шығару керек.

25. Экранның жоғарғы жағына екі терезе салып, оның біріншісінің ішіне өз атыңызды, екіншісіне фамилияңызды жазып қойыңыз. 10 секундтан кейін терезелер ішін тазартып, өз аты-жөніңіздің орнына досыңыздың аты мен фамилиясын орналастырыңыз.

26. Экранның сол жақ шетіне көк түсті терезе салып, оны оң жаққа қарай қадамын бір символ етіп жылжытып отыратын программа құру керек.

27. Экранға біртіндеп үлкейе бастайтын шағын қызыл терезе салу қажет. Үлкею қадамын *X* координатасы үшін 3, ал *У* координатасы үшін 1 етіп алыңыздар.

28. Экранға біртіндеп бір символ аумағына дейін кішірейе бастайтын барынша үлкен сары терезе салу қажет. Кішірею қадамын *X* координатасы үшін 3, ал *У* координатасы үшін 1 етіп алыңыздар.

29. Экранға сиып тұратындай етіп 9 терезе салыңыздар да, олардың ішіне әртүрлі физика формулаларын жазыңдар.

12. СИ ПРОГРАММАСЫН ОРЫНДАУ ОРТАСЫ

12.1. Турбо Си редакторы терезесі

Turbo Си программалау жүйесі Си тілінде программа құрастырып, оны орындауға мүмкіндіктер береді. Жалпы тұтынушы мынадай әрекеттерді орындай білуі керек:

- программаның мәтінін жазып, дискіде программа файлы ретінде сақтау;
- программаны ***компиляциядан*** өткізіп, егер синтаксистік қателері бар болса, оларды түзету;
- программаны орындап, нәтижесін алу.

Сонымен, кез келген программа оны теру, компиляциялау, құрастыру, атқарылушы модульді жасау және орындап нәтиже алу сатыларынан өтуі тиіс.

Си тілінің біріктірілген (интегралданған) ортасында программа орындау төмендегідей қадамдардан тұрады:

1) компилятор қажет файлдарды іздеп тауып алуы үшін ортаның параметрлерін тағайындау;
2) программалық файлды редактор орасына жүктеу немесе теру;
3) атқарылатын модульді (орындалатын файлды) жасау;
4) программаны іске қосу және орындау;
5) программада қате болса, оны жөндеп түзету.

Программа құру үшін, ең алдымен Турбо Си біріктірілген ортасымен жұмыс істей білу керек.

Турбо Си ортасы – арнайы көп терезелі мәтіндік редактор. Ол көбінесе **C:\TC\BIN\tc.exe** файлын іске қосу арқылы немесе соның жарлықтарының бірін іске қосу жолымен жүктеледі:  

TC.EXE файлының жұмыс істеу барысында экранда Турбо Си ортасының негізгі терезесі ашылады (1-сурет).

Турбо Си ортасын автоматты түрде жүктеп, орыс (қазақ) әріптерін теру мүмкіндігін беретін C.bat пакеттік файлының мәтінін мынадай түрде құрастыруға болады:

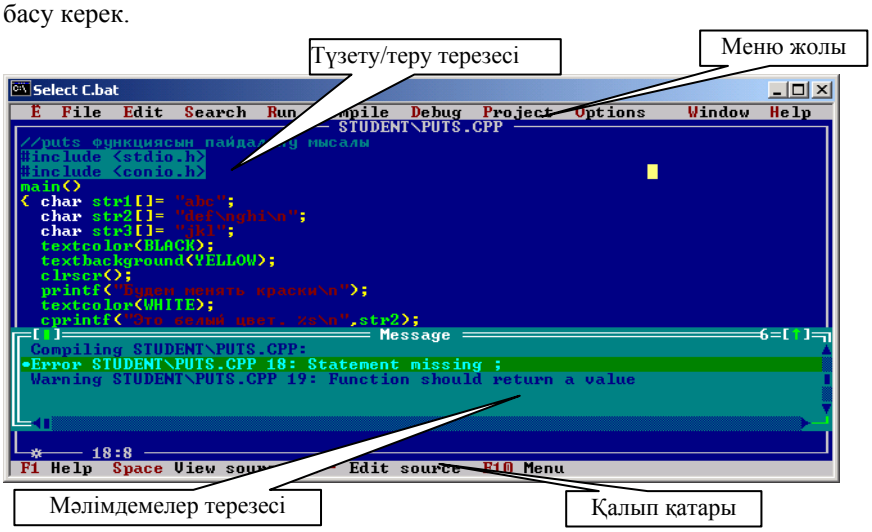
```
C:
cd TC\BIN
rk.com
tc.exe
```

C.bat файлын немесе оның жарлығын шерту арқылы да Турбо Си ортасы ашылады. Ол іске қосылған соң, орыс әріптеріне және ағылшын әріптеріне ауысу қос Shift пернелерін (ол басқаша да болуы мүмкін) қатар басу арқылы орындалады. Турбо Си редакторы DOS ортасында жұмыс

істейтін көкшіл экранға шығады. Ол экран төрт бөліктен тұрады (12.1-сурет):

- меню жолы,
- түзету/теру терезесі,
- мәлімедеме хабарлар терезесі,
- қалып қатары.

Терезені үлкейту үшін **Alt+Enter** пернелерін басу керек, редактор терезесін экранды толық алатындай етіп үлкейту үшін, оның оң жақ жоғарғы бұрышындағы тілсызық [↑] батырманы шерту керек. Ал терезені жабу үшін – сол жақ жоғарғы шеттегі төртбұрышты батырманы [□] шертеміз. Турбо Си-ден шығу үшін *Alt* және *X* (латын) пернелерін қатар басу керек.



12. 1-сурет. Турбо Си біріктірілген ортасы

Терезе нөмірі оның оң жақ жоғары жағында орналасады. Керекті терезеге (1 – 9) көшу үшін:

- Alt+0 пернелерін басқанда шығатын терезелер тізімінен керектісін таңдау арқылы;
- Alt+5 деп терезе нөмірін 5-ті енгізіп көшу;
- F6 пернесі арқылы терезелердің бірінен біріне көшуге болады.

F5 пернесі арқылы терезені үлкейтуге немесе аздап кішірейтуге болады.

Меню жолында 11 команда бар, олар әр түрлі қызметтер атқарады, әр менюді таңдағанда, оның ішкі командалары ашылады. Солардың кез келгенін таңдап орындай аламыз.

12.2. Меню командалары

Программа іске қосылып, терезе ашылғаннан кейін, курсор жұмыс алабында тұрады.Меню қатарына F10 пернесі арқылы шығып, ESC арқылы жұмыс алабына ораламыз. Меню қатарының командаларын және төменгі сатылы командаларының қажеттісін ← ↑ → ↓ бағыттауыштар арқылы таңдай аламыз. Команданы орындау үшін **Enter** пернесін басамыз.

Осы әрекеттерді тышқан қолтетігімен де қалыптағыдай етіп орындауға болады.

Командалар тізімі

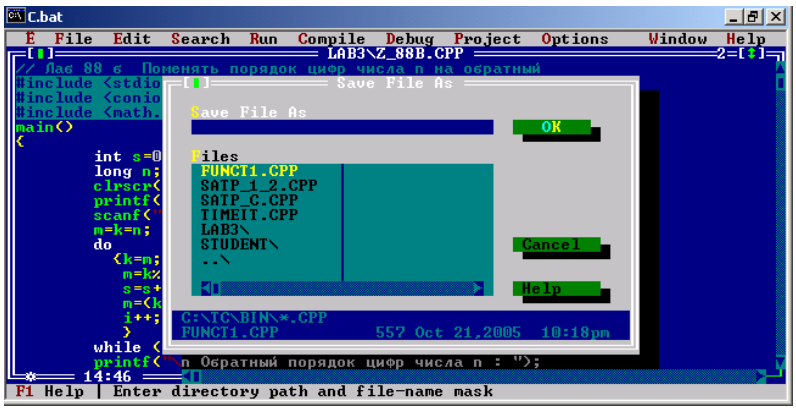
•**Ё** – қосымша әрекеттер орындау.
•**File** (файл) – файлдармен жұмыс істеу.
•**Edit** (түзету) – ашық терезедегі мәтінді түзету режимдерін орындау.
•**Search** (іздеу) – іздеп табу әрекеттерін орындау.
•**Run** (атқару) – программаны орындау.
•**Compile** (компиляция) – программаны компиляциядан өткізу.
•**Debug** (отладка) – программаны жөндеу.
•**Options** (варианттар) – орта параметрлерін тағайындау.
•**Windows** (терезе) – тереземен жұмыс істеу.
• **Help** (көмек) – анықтамалық жүйеден көмек алу.

Қосымша әрекеттер жұмыс істеп тұрған файлдарды анықтау, ішкі Ассемблерді іске қосу сияқты әрекеттерді орындайды.

File менюінің ішкі командалары:

Open F3 Бұрын жазылған файлды ашу {A:\LB1.C}
New Жаңа файл ашу
Save F2 Файлды дискіге жазып сақтау
Save as … Файлды басқаша түрде жазып сақтау
Save all Барлығын да жазып сақтау
Change dir Директорийды өзгерту
Print Ашық терезедегі файлды қағазға басу
Dos shell DOS-қа (Turbo Си-ден) уақытша шығу
Exit Alt-X Turbo Си-ден шығып кету

Файлды дискіге өзіңіз қойған атпен сақтауға арналған *Басқаша сақтау* (Сохранить как...) терезесі



Edit менюінің ішкі командалары:

Undo (болдырмау) Alt + Bksp – соңғы орындалған команданың әрекетін болдырмай алып тастайды.

Redo Shift +Alt + Bksp – Undo командасының кері қайтарған командасы әрекетін қайтадан іске қосады.

Cut (қиып алу) Shift+Del – белгіленген бөлікті буферге қиып алады (бұрынғы орнында қалмайды).

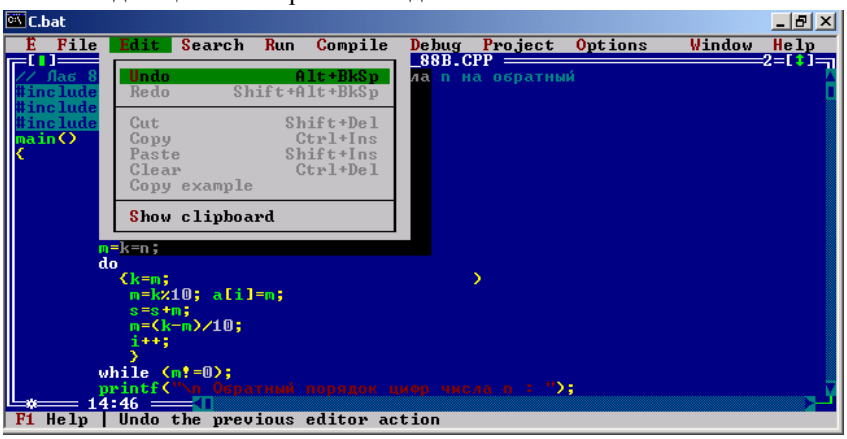
Copy (көшіру) Ctrl+Ins – белгіленген бөлікті буферге көшіреді (бұрынғы орнында сақталады).

Paste (кірістіру) Shift+Ins – курсор орналасқан жерге буфердегі ақпаратты кірістіріп қояды, яғни енгізеді.

Clear (өшіру) Ctrl+Del – белгіленген бөлікті өшіру.

Copy Examples – мысалды көшіру.

Show Clipboard (буферді ашу) редактор терезесінен буферге алынған мәтінді сақтайтын терезені ашады.



Find (табу) – табуға қажетті сөзді енгізу мүмкіндігін беретін сұхбат терезені ашады.

Replace (орын алмастыру) Alt+S+R – іздейтін мәтін мен оны алмастыратын мәтінді енгізу мүмкіндігін беретін сұхбат терезені ашады.

Search Again (қайтадан іздеу) Ctrl+L – **Find** немесе **Replace** командаларының соңғы әрекетін қайталайды.

Go to line number (нөмір қатарына бару) – курсорды нөмірі көрсетілген қатарға орналастырады.

Previous error Alt+F7 – алдыңғы қате орнына бару.

Next error Alt+F8 – келесі қате орнына бару.

Run менюінің ішкі командалары:

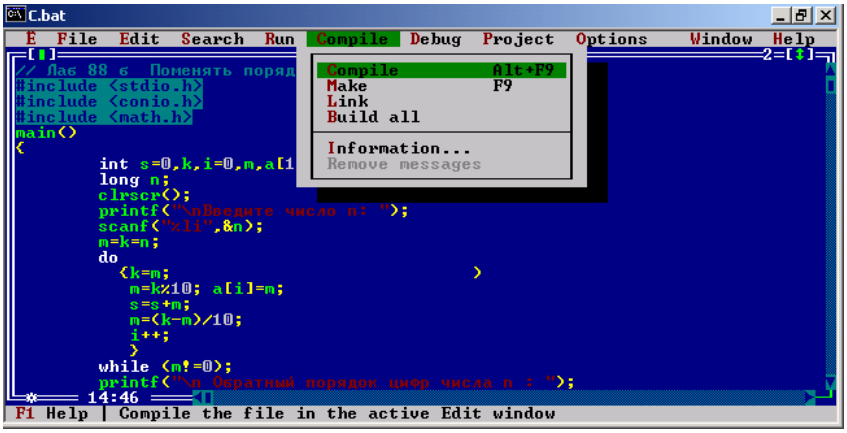
Run (орындау) Ctrl+F9 – орнатылған параметрлерді қолдана отырып, редактор терезесіндегі екпінді программаны орындайды.

Program reset Ctrl+F2 (сброс программы) – жөндеушінің орындап жатқан әрекетін тоқтатып, программаға бөлінген орынды босатып, барлық ашық файлдарды жабады.

Go to cursor (курсорға өту) F4 – екпінді терезедегі программаны курсор тұрған орындағы қатарға дейін орындайды.

Trace into (қосалқы программаға кіріп, қадамдық орындау) F7 – программадағы операторларды қадам бойынша көрсете отырып орындайды.

Step over (программаны қадам бойынша орындау) – программа мәтінінің бір жолына сәйкес келетін кезекті операторды біртіндеп орындайды.



Compile менюінің ішкі командалары:

Compile Alt+F9 – екпінді терезедегі программаның қатесін тексереді. Синтаксистік қате жөнінде хабарлама береді де, курсор қате жіберілген

орынға орналасады. Қате жоқ болса, компилляцияның сәтті болғаны жөнінде хабарлама береді.

Make (Программаны жинақтау). Егер негізгі программада немесе негізгі модульде қолданылатын жеке модульдердің мәтінінде объектілік файлды алғаннан кейін өзгеріс болса, соған сәйкес модульдер қайта тексеріледі, одан кейін негізгі программа немесе негізгі модульден тұратын файл қайта құрылады.

Build all (Программа құру) – бұл команда орындалғанда, негізгі программа және негізгі модульде қолданылатын барлық модульдер қайта компиляциядан өткізіледі.

Debug (жөндеу) менюі ішкі командалары:

Inspect… (Alt+F4) – Inspector терезесін ашады, ол объект-ілер мәнін талдауға көмектеседі.

Evaluate/modify… (Ctrl+F4) үш өрісі бар терезе ашады: Expression, Result, New value — олар айнымалы мәндерін көріп, оларды өзгерту мүмкіндігін береді.

Call stack (Ctrl+F3) – программада қолданылған функциялар тізбегін – стекті көрсететін қосалқы программа терезесін көрсетеді.

Watches суырылып шығатын менюді ашып, жаңа өрнектер енгізіп, оның нәтижесін көрсете алады.

Breakpoints... – түзету режимінде тоқтау нүктесімен жұмыс істеу мүмкіндігін беретін терезе ашады.

Project командалары қажет болғанда, жобалар ашу, толықтыру және жабу ісін атқарады.

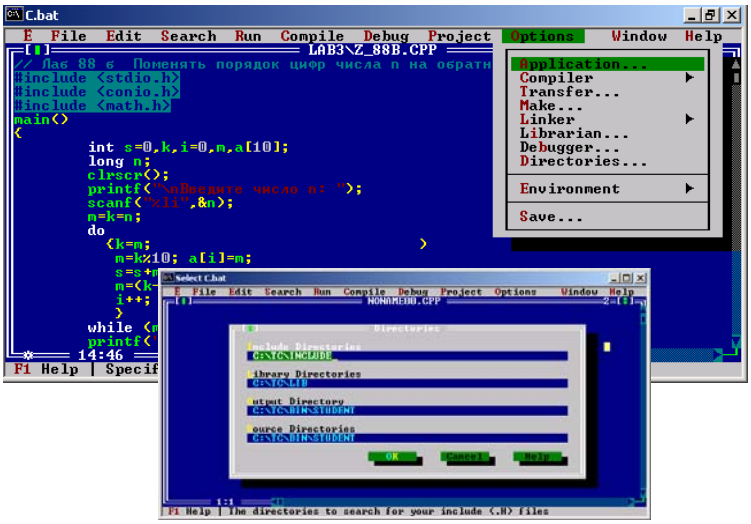
Жоба — бір-бірімен байланысты файлдар жиыны, бірнеше объектілік файлдар бірден компиляциядан өткізіліп, атқарылатын бір программа жасайды. Жоба көп файлды программалар кезінде керек. Кейде бір файлмен жұмыс істегенде де қолданылады.

Модулдық программалау барысында көпфайлдық компиляциялаусыз жұмыс істеу мүмкін емес. Көлемі үлкен программалармен жұмыс істеу кезінде ол программаның бөліктерін бірнеше файлдарға сақтау анағұрлым ыңғайлы. Әрбір файл бүтіндей бір немесе бірнеше функцияны қосуға тиіс. Ол файлдардың аттары арнайы файл-жобаға жазылады, IDE ол ақпараттан мәтіндік файлдардың қайсысын орындалатын файлға (.EXE) біріктіру керек екендігінен хабардар болады.

Файл-жобалармен жұмыс істеуге қажетті бұйрықтардың барлығы *Project* мәзіріне қосылған.

Файл-жобаларды ұйымдастыру үшін ол файл-жобаны ашу керек. Ол үшін **Project\Open Project**… бұйрығы орындалады және IDE экранның төменгі жағындағы арнайы *Project* терезесін іске қосады. Қажетті файл-жобаны жүктейтін немесе берілген атпен жаңа файл-жоба құратын диалог

терезесі ашылады.Жаңа файл-жоба құрылған болса, *Project* терезесі бастапқыда бос болады. Жобаға файлдарды қосу немесе оларды жою **Project\Add item**… және **Project\Delete item** бұйрықтарымен орындалады. Меңзер *Project* терезесінде орналасқан жағдайда осы мақсатта Ins және Del батырмаларын басса да жеткілікті. Файлдарды жобаға қосу кезінде қажетті файлды таңдауға мүмкіндік беретін диалог терезесі ашылады. *Project* терезесі жобаға қосылған бір файлдан оларды редакциялау барысында келесісіне көшуді жеңілдетеді. Ол үшін *Project* терезесіндегі қажетті жолға меңзерді апарып ENTER пернесін басса жеткілікті. Borland C++ ортасында жұмыс істеу кезінде программа бір ғана файлдан тұрғанның өзінде жобаны қолданған ыңғайлы.



Options менюінің командалары Турбо Си ортасының келісім бойынша тағайындалатын параметрлерін көру және оларды өзгерту мүмкіндігін береді. Олардың көптеген мәндерін өзгертпей, қалдыруға болады. Мұнда түйінді сөздер түсін (16 түс) өзгерту мүмкіндігі бар (Options/Enviroment/Colors/Syntax).

Directories командасы тақырыптық файлдар каталогын (*Include Directories*), кітапханалық функциялар (*Library Directories*) каталогын және терілген файлдар мен олардың нәтижелік файлдары (*Output Directory*) қайда орналасатынын көрсетіледі.

Мысалы, егер **TC** программалары **C:\TC** каталогында орналасса, онда **Include Directories** өрісінде **C:\TC\INCLUDE** деп көрсеткен дұрыс болады, ал Library Directories өрісіне — **C:\TC\LIB** деп жазу керек. **Output Directory** жолына нәтижелік файлдар орналасатын каталогты,

мысалы, **C:\TC\BIN\STUDENT** деп көрсеткен ыңғайлы болып саналады немесе **C:\TC\BIN** каталогын қалдыру үшін — нүкте "." енгізе салу керек. Керекті параметрлер енгізілген соң, оларды **Options – Save...** командасымен есте сақтап қою қажет.

Window (Терезе) меню командалары терезені ашу, жабу, экранды жылжыту әрекеттерін орындау мүмкіндігін береді.

Size/Move	Ctrl+F5	Терезе көлемі мен орны
Zoom	F5	Терезені ұлғайту
Tile		Сатылы (каскадты)
Next	F6	Келесі терезе
Close	Alt+F3	Жабу
Output		Нәтижелік терезе
User screen	Alt+ F5	Тұтынушы экраны
List All	Alt+0	Басқа терезелерді ашу

Help (көмек) меню командалары жүйедегі анықтамалық ақпаратты оқу мүмкіндігін береді.

Contents (экранға шығарылған ақпарат жөнінде мәлімет) ағымдағы уақытта экранға шығарылған мәлімет жөнінде (екпінді терезе, таңдалған меню командасы, жіберілген қате) мәліметті сұхбат терезеге шығарады.

Index (түйінді сөздер) Shift+F1 – жүйеде бар барлық анықтамалық ақпарат тізімін алфавиттік ретімен түйінді сөздер бойынша шығарады.

Topic search (сөз бойынша іздеу) Alt+F1 – курсор орналасқан сөз жөнінде анықтамалықты шығарады. Егер сол сөз жөнінде анықтамалық жоқ болса, алғашқы символдарының саны көп сәйкес келетін түйінді сөздер тізімін береді.

Prevoius topic (алдыңғы тақырып) алдыңғы сұранысқа сәйкес келетін анықтамалықты шығарады. Жүйе 20 сұранысты сақтай алады.

12.3. Қателер коды және олардың мәліметтері

Турбо Си ортасы программа компиляциядан өткен кезде пайда болған қателер жайлы толық мәлімет береді. Қате кездескен кезде орта автоматты түрде бастапқы программа мәтінін экранға шығарып, курсорды қате табылған орынға орналастырады және редактордың жоғарғы жолында диогностикалық мәлімет пайда болады. F1 пернесінен басқа кез келген пернеге бассаңыз, жоғарғы жол бастапқы қалпына келіп, интегралданған орта редакциялау режиміне ауысады. Егер қате жайлы мәлімет шыққаннан кейін F1 пернесін бассаңыз, экранда қатені түзету жайлы нұсқаулар жазылған анықтама қызметінің терезесі пайда болады. Кейбір қателер бірден емес, программа мәтінін талдау барысында анықталады.

Қате нөмірі	Қате түсінігі
1	*Out of memory (Жады шекарасынан асып кету).*
2	*Identifier expected (Идентификатор көрсетілмеген).*
3	*Unknown identifier (Белгісіз идентификатор).*
4	*Duplicate identifier (Идентификатор қайталанған).*
5	*Syntax error (Синтаксистік қате).*
6	Error in real constant (*Нақты тұрақтыда қате бар*).
7	*Error in integer constant (Бүтін тұрақтыда қате бар).*
8	*String constant exceeds line (Тіркестік тұрақты берілген аймақтан тыс жатыр).*
9	*Too many nested files (Кіріктірілген файлдар саны тым көп).*
10	*Unexpected end of file (Файл соңы табылмады).*
11	*Line too long (Тым ұзын жол)*
12	*Type identifier expected (Тип идентификаторы қажет)*
13	*Too many open files (Ашылған файлдар саны көп)*
14	*Invalid file name (Файл аты дұрыс емес)*
15	*File not found (Файл табылмады)*
16	*Disk full (Диск толып кеттікен)*
17	*Invalid compiler directive (Компилятор директивасы қате)*
18	*Too many files (Файлдар саны тым көп).*
19	*Undefined type in pointer definition (Көрсеткіш сипаттауында тип көрсетілмеген).*
20	*Variable identifier expected (Айнымалының иденфикаторы жоқ).*

Программа орындалу кезінде шығатын қателер

Программа орындалу кезінде анықталатын кейбір қателер экранда Runtime error nnn at xxxx:yyyy (xxxx:yyyy адресі бойынша nnn кезенін орындаудағы қате) мәлімдемесінің шығуына әкеледі, мұнда nnn — қате нөмірі; xxxx:yyyy — адрес (сегмент немесе жылжу). Бұл мәлімдемеден кейін программа өз жұмысын тоқтатады. Программа орындалу кезінде шығатын қателер төртке бөлінеді: СОЖ (сұхбатты операциялық жады) анықтайтын қателер (1-ден 99-ға дейінгі қателер), енгізу-шығару қателері (100-ден 149-ға дейінгі қателер), дағдарысты қателер (критические ошибки) (150-ден 199-ға дейін) және фаталды қателер (200-ден 255-ке дейінгі қателер).

Операциялық жүйе анықтайтын қателер

Қате №	***Аты***
1	*Invalid function number (Функция нөмірі дұрыс жазылмаған).*
2	*File not found (Файл табылмады).*
3	*Path not found (Жол табылмады).*
4	*Too many open files (Ашылған файлдар саны көп).*
5	*File access defined (Файлға қол жеткізуге рұқсат жоқ)*
6	*Invalid file handle (Ретсіз файлдық канал).*
12	*Invalid file access code (Файлға қол жеткізу коды нақты емес).*
15	*Invalid drive number (Дискіжетек нөмірі ретсіз).*
16	*Cannot remove current directory (Ағымдағы каталогты өшіруге болмайды).*
17	*Cannot rename across drives (Атты ауыстырғанда әртүрлі дискіжетек аттарын көрсетуге болмайды).*

Енгізу-шығару қателері

Егер операторлардың біреуі {$1+} директивасымен компиляциядан өтсе, онда енгізу-шығару қатесі программаның орындалуын тоқтатады. {$1–} қалпында программа орындалуын жалғастырып, қате IORESULT функциясымен қайтарылады.

Қате №	***Аты***
100	*Disk read error (Дискіден оқу кезінде қате кетті).*
101	*Disk write error (Дискіге жазу кезінде қате кетті).*
102	*File not assigned (Файлға ат берілмеген).*
103	*File not open (Файл ашылмаған).*
104	*File not open for input (Файл енгізу үшін ашылмаған).*
105	*File not open for output (Файл жазу үшін ашылмаған).*
106	*Invalid numeric format (Сан форматы дұрыс емес).*

Дағдарысты қателер

Қате №	***Аты***
150	*Disk is write protected (Диск жазудан сақталған).*
151	*Unknown unit (Белгісіз модуль).*
152	*Drive not ready (Дискенгізгіш «дайын емес» қалып-күйінде).*

153	*Unknown command (Бейтаныс команда).*
154	*CRC error in data (Бастапқы мәндерде қате бар).*
155	*Bad drive request structure length (Дисктен мәлімет аларда құрылым ұзындығы дұрыс көрсетілмеген).*
156	*Disk seek error (Дисктен оқитын құрылғы бастарын дискке орнату операциясы кезінде қате кетті).*
157	*Unknown media type (Тасмалдауыш типі белгісіз).*
158	*Sector not found (Сектор табылмады).*
159	*Printer out of paper (принтерде қағаз бітті).*
160	*Device write fault (Құрылғыға жазу кезінде қате кетті).*
161	*Device read fault (Құрылғыдан оқу кезінде қате кетті).*
162	*Hardware failure (Аппаратура жұмыс істемей тұр).*

Фаталды қателер

Бұл қаталер программа жұмысының бірден тоқталуын жүзеге асырады.

Қате №	***Аты***
200	*Division by zero (Нөлге бөлу).*
201	*Range check error (Шекараларды тексеру кезінде қате шықты).*
202	*Stack overflow error (Стек толып кетті).*
203	*Heap overflow error (Мәлімет толып кетті).*
204	*Invalid pointer operation (Көрсеткішпен орындалатын белгісіз операция).*
205	*Floating point overflow (Жылжымалы нүктемен жұмыс жасау кезінде мәндер берілген аймақтан шығып кетті).*
206	*Floating point underflow (Жылжымалы нүктемен жұмыс жасау кезінде реті бұзылды).*
207	*Invalid floating point operation (Жылжымалы нүктемен жұмыс жасауға болмайтын операция) .*
208	*Overlay manager not installed (Оверлейді басқарудың ішкі жүйесі орнатылмаған).*
209	*Overlay file read error (Оверлейлік файлды оқу кезінде қате жіберілген).*
210	*Object not initialized (Объект инициалданбаған).*

211	*Call to abstract method (Абстракты ережені шақыру).*
212	*Stream registration error (Тіркеуге алу ағынында қате бар).*
213	*Collection index out of range (Терілген индекс диапозон шекарасынан тыс жатыр).*
214	*Collection overflow error (Коллекция толып кеткен).*

СӨЗДІК

А

Аргумент заданный по умолчанию	Келісім бойынша берілген аргумент
Арифметические операции	Арифметикалық амалдар
Арифметика указателей	Нұсқауыштар арифметикасы

Б

Библиотека	Кітапхана

В

Ввод	Енгізу
Ввод данных	Мәліметтерді енгізу
Ввод текста	Мәтінді енгізу
Вывод	Шығару
Вывод данных	Мәліметтерді шығару
Вывод текста	Мәтінді шығару
Выражение сравнения	Өрнектерді салыстыру

Д

Двоичный поиск	Екілік түрде іздеу
Динамическая память	Динамикалық жад
Динамическое распределение памяти	Жадты динамикалық түрде бөлу

З

Знаки	Таңбалар

И

Инициализация	Бастапқы мән беру, инициалдау
Интегрированная среда	Біріктірілген (интегралданған) орта
Именование переменной	Айнымалы атауы

К

Классы памяти	Жад кластары
Конкатенация строк	Сөз тіркестерін біріктіру
Константы с плавающей точкой	Жылжымалы нүктелі тұрақты
Константы символьные	Символдық тұрақты (константа)

Константы целочисленные	Бүтін тұрақты
Л	
Логические выражения	Логикалық өрнектер
М	
Массив	Жиым
Массив двумерный	Екі өлшемді жиым
Массив динамический	Динамикалық жиым
Метод	Әдіс, тәсіл
О	
Область имен	Атаулар аймағы
Объединение	Біріктіру
Операция декремента (--)	Декремент (бірге кеміту) операциясы (--)
Операция инкремента (++)	Инкремент (бірге арттыру) операциясы (++)
Операция приведения типов	Типтерді келтіру операциясы
Оператор присваивания	Меншіктеу, тағайындау
Оператор составной	Құрама амал
Оператор цикла	Цикл операторы
Оператор цикла с предусловием	Шарты алдымен тексерілетін цикл операторы
Оператор цикла с постусловием	Шарты соңынан тексерілетін цикл операторы
Опертор выбора	Таңдау операторы
Оператор условный	Шартты оператор
Освобождение памяти	Жадты босату
Отношение	Қатынас
П	
Память автоматическая	Автоматты жад
Память динамическая	Динамикалық жад
Память статическая	Статикалық жад
распределение	Жадты бөлу (үлестіру)
Переменная автоматическая	Автоматтық айнымалы
Переменная внешняя	Сыртқы айнымалы
Переменная временная	Уақытша айнымалы
Переменная глобальная	Ауқымды айнымалы
Переменная локальная	Жергілікті айнымалы
Переменная ссылочная	Сілтеме айнымалы
Перестановки	Орын ауыстыру
Перечисление	Ретімен санау

Последовательность	Тізбек
Преобразование	Түрлендіру
Преобразование типов данных	Мәліметтер типтерін түрлендіру
Приведение типов	Типтерді келтіру
Приоритет операции	Амалдар үстемділігі
Присваивание	Меншіктеу
Р	
Рекурсия	Рекурсия, функцияның өзін өзі шақыруы
С	
Случайные	Кездейсоқ
Символ новой строки (\n)	Жаңа жолға көшу символы
Символ константы	Тұрақты символы
Символические	Символдық
Сортировка	Сұрыптау, іріктеу
Сравнение строк	Тіркестерді салыстыру
Ссылка	Сілтеме, сілтеу
Строка	Тіркес, сөз тіркесі
Структура	Құрылым
Т	
Таблица	Кесте
Тип данных	Мәліметтер типтері
Тип данных без знака	Таңбасыз мәліметтер типі
Тип данных основной	Негізгі мәліметтер типі
Тип данных с плавающей точкой	Жылжымалы нүктелі
Тип данных символьный	Символдық мәліметтер типі
Тип данных целочисленный	Бүтін мәіметтер типі
У	
Указатель	Нұсқауыш
Указатель на функции	Функцияға нұсқауыш
Управляющие последовательности языка Си	Си тілінің басқару тізбектері
Ф	
Форматирование	Пішімдеу, форматтау
Функция определенная пользователем	Тұтынушы анықтаған функция
Функция со многими аргументами	Көп аргументті функция
Функция сравнения	Салыстыру функциясы
Ц	
Целочисленная константа	Бүтін мәнді тұрақты
Цикл	Цикл, қайталау

Цикл вложенный	Қабаттасқан цикл

Ш

Шаг цикла	Цикл қадамы, адымы

Я

Язык программирования	Программалау тілі

ТЕСТ СҰРАҚТАРЫ

1. Идентификатор дегеніміз не?

1. программадағы объектінің аты
2. динамикалық жады
3. жиыннан тұратын массив
4. программаның берілу жолы
5. компиляторға арналған сөздер

2. Символдық типтегі шамаларға компилятор жадында қанша байт орын бөлінеді?

1. 1
2. 2
3. 10
4. 8
5. 27

3. Жиым (массив) дегеніміз не?

1. ұяшықтарда орналасқан мәліметтер
2. бір атаумен аталған бір типті мәндер тізбегі
3. өлшемді анықтайтын тип
4. динамикалық жадыны пайдаланатын көрсеткіш
5. сілтемені анықтайтын типтер жиынтығы

4. Қандай да бір әрекеттер тізбегін орындайтын операциялар мен сипаттамалардың айқындалған тізбегін ... деп атайды.

1. процедура
2. рекурсия
3. функция
4. дұрыс жауап жоқ
5. 1,2

5. Функцияның сипаттамасы басындағы параметрлер қалай аталады?

1. формальды

2. фактілі
3. аргументті
4. 2,3 жауап дұрыс
5. дұрыс жауап жоқ

6. Си тілінде кез келген программаның орындалуы қай функциядан басталады?
1. random
2. main()
3. randomize()
4. clrscr()
5. Әр программада әр түрлі

7. Өзін-өзі шақыратын функция не деп атаймыз?
1. процедура
2. функция
3. рекурсия
4. 1,2
5. дұрыс жауап жоқ

8. Кітапханалық функциялар қайда орналасқан?
1. кітапханалық файлдарда
2. Си ішінде
3. компьютерде
4. дұрыс жауап жоқ
5. Internet-те

9. Си алфавиті:
1. араб цифрі: 0-9
2. арнайы белгілер
3. түйінді сөздер
4. операциялар таңбалары
5. барлығы дұрыс

10. Латын әріптерінен, цифрлардан тұратын тек әріптерден басталуы тиіс таңбалар тізбегін ... деп атайды.
1. тұрақты
2. айнымалы
3. идентификатор
4. дұрысы жоқ
5. функция

11. a[3] = {1,3,5} жиымы берілсін делік. Осы жиымды экранға шығаратын программаны көрсетіңіз...

1. int i, a[3] = {1,3,5};
for(i=0;i<3;i++)
printf(“ %d”,a[i]);

2. int a[3] = {1,3,5}, i;
for(i=0;i<3;i++)
printf(“ %f”,a[i]);

3. int a[i] = {1,3,5}, i;
for(i=0;i<3,i++)
printf(“ %k”,a[i]);

4. int a[3], i;
for(i=0;i<3;i++)
printf(“ %i”,a[i+1]);

5. int a[3], i;
for(i=0;i<3;i++)
printf(“ %c”,a[i]);

12. Си тілінде тұрақтыларды сипаттау үшін қандай түйінді сөз қолданылады?

1. int
2. var
3. const
4. procedure
5. дұрысы жоқ

13. return (өрнек) операциясының қызметі:

1. функцияны шақырған операторға өрнектің мәнін қайтарады
2. С тілінде ондай оператор қолданылмайды
3. функция мән қайтармаған кезде қолданылады
4. программаға әсемдік береді
5. қызметі анықталмаған, заң бойынша программа соңында тұруы крек

14. rand() функциясының тақырыптық файлда орналасқан прототипі

қандай?
1. <math>
2. <string>
3. <stdlib>
4. <cmath>
5. rand

15. Функцияның ішінде ғана белгілі, соның ішінде анықталатын айнымалы қалай аталады
1. ауқымды (глобальді) айнымалы
2. жергілікті айнымалы
3. белгісіз айнымалы
4. белгілі айнымалы
5. рекурсивті айнымалы

16. Егер float пен double типтері араласса, нәтижесі – ... болады;
float;
long ;
double ;
int ;
short;

17. Кез келген блоктан немесе функциядан тыс хабарланған айнымалы...
1. ауқымды (глобальді) айнымалы
2. жергілікті айнымалы
3. программалық айнымалы
4. базалық айнымалы
5. унарлық айнымалы

18. ... компиляторға функция арқылы берілетін аргументтердің санын, типін, ретін анықтауға көмектеседі
1. функцияның идентификаторы
2. функцияның прототипі
3. файлдың протатипі
4. жергілікті протатип
5. аумақтық идентификатор

19. Функцияның тақырыбында void түйінді сөзі не үшін қолданылады?
1. функция мән қайтаратынын көрсету үшін

2. функцияның тақырыбында void түйінді сөзі қолданылмайды
3. дұрыс жауабы жоқ
4. функцияның ешқандай мән қайтармайтынын көрсету үшін
5. функция мән сақтайтынын көрсету үшін

20. Қандай айнымалылар өзі сипатталған функциядан шыққаннан кейін де өз мәнін сақтайды?
1. static секілді хабарланған жергілікті айнымалылар
2. аумақты айнымалылар
3. extern секілді хабарланған жергілікті айнымалылар
4. static секілді хабарланған ауқымды айнымалылар
5. жергілікті айнымалылар

21. Блок дегеніміз не?
1. функцияның екінші аты
2. айнымалыларды сипаттауы бар құрама оператор
3. кездейсоқ оператор
4. қатені анықтайтын оператор
5. дұрыс жауап жоқ

22. Айнымалы дегеніміз не?
1. белгілі бір атауы бар жады аймағы, онда анықталған типтің мәліметтері сақталады
2. анықталмаған тип мәліметтерінің жадыда сақталуы
3. дұрыс жауабы жоқ
4. жады класстарының бірнеше бөліктерге бөлінуі
5. операндадан, операция белгісінен, жақшадан тұратын, анықталған типтің мағынасын білдіруі

23. Шартты операция форматын көрсетіңіз
1. операнд_1? операнд_2:операнд_3
2. операнд_1:=операнд_2
3. 1,2
4. if (өрнек)оператор_1:[else оператор_2;]
5. if (тұрақты өрнек)1:[оператор тізбегі]

24. Жады класстарын беру үшін қандай спецификатор қолданылады
1. auto
2. extern
3. static
4. register

5. барлығы дұрыс

25. Си тілінде қайталау операторының қандай түрлері бар?
1. while (өрнек) оператор;
2. do ... while (өрнек)
3. дұрыс жауап жоқ
4. 1, 2, 5
5. for (инициализациялау; өрнек; модификация) оператор;

26. Тармақталу операторы дегеніміз не?
1. Алгоритмнің белгілі бір шарттың орындалуына немесе орындалмауына байланысты тармақталып, бірнеше жолдарға бөлінуі
2. дұрыс жауап жоқ
3. бір ғана шарттың орындалуын тексеретін оператор
4. шарттардың орындалмауын тексеретін оператор
5. операторлардың тармақталуын тексеретін оператор

27. #include директивасын қолданған кезде бұрыштық ("<",">") жақшаның орнына тырнақшаларды қолдануға болады ма, болса оның мәні неде?
1. болмайды
2. болады, бірақ одан ештеңе өзгермейді
3. болады, бұл уақытта файлды іздеу әрекеті бастапқы файлы бар каталогта жүргізіледі, сосын стандартты каталогтан іздейді
4. болады, онда файлды стандартты каталогтан іздейді
5. болады, бұл жағдайда файлды стандартты каталогтан іздейді, сосын бастапқы файлы бар каталогты қарастырады

28. Бастапқы ".h-файлдың" құрамына
1. типтің, тұрақтылардың, кіргізілген функциялардың, шаблондардың, тізімдердің анықталуы
2. функцияның, мәліметтің, шаблонның, аттың сипатталуы
3. атау кеңістігі
4. процедуралардың директивалары
5. көрсетілген жауаптың бәрі дұрыс

29. #define директивасы не үшін қолданылады
1. тұрақтының мәнін беру үшін
2. макростар үшін
3. шартты компиляторды басқаруға арналған символ үшін

4. символдық константа үшін
5. берілген жауаптың бәрі дұрыс

30. Құрамында функциялары немесе анықталған мәліметтері бар бастапқы (тақырыптық) файлдардың кеңейтілуі қандай және ол файлдар қалай аталады?
1. ".htp", ".htp-файлдары"
2. ".h", ".h-файлдары"
3. ".c" ".c-файлдары"
4. ".f" ".f- файлдары"
5. ".exe", ".exe- файлдары"

31. Сөз тіркесінің (жолдың) ұзындығы қай функцияның көмегімен анықталады?
1. strlen
2. src
3. return
4. template
5. main()

32. 10 нақты саннан тұратын массвті сипаттау қай вариантта дұрыс берілген?
1. int a[10];
2. float a(10);
3. float a[10];
4. float a[1..10];
5. int a[1..10];

33. Операндтардан, операция таңбаларынан, жақшалардан тұратын .... мәндерін есептеу үшін қолданылады. Көп нүктенің орнындағы сөзді табыңыз...
1. өрнектер
2. айнымалылар
3. тұрақтылар
4. логикалық типтер
5. жады класы

34. Айнымалы дегеніміз не?
1. программа орындалуы барысында әр түрлі мәндер қабылдайтын шама
2. жады класының бірнеше аймақтарға бөлінуі

3. программалар тізбегі
4. алгоритмнің белгілі бір шартының орындалуы
5. дұрыс жауабы жоқ

35. Таңдау операторы (нұсқауы) қай сөз арқылы беріледі?
1. switch
2. return
3. for
4. if
5. class

36. Егер long пен float типтері араласса, нәтижесі – ... болады;
char;
short;
int ;
float;
double.

37. Бөлгендегі қалдық табу операциясы қалай беріледі?
1. %
2. /
3. *
4. <<
5. &

38. Блок ішінде анықталған айнымалы қалай аталады?
1. ауқымды
2. тұрақты
3. жергілікті
4. тұрақсыз
5. сілтеуіш

39. Блоктың ішінде сипатталған идентификатор
1. глобальды көрініске ие
2. локальді идентификатор
3. оператордың белгісі
4. функцияның прототипінде көрсетілген параметр
5. көрсетілген типтің айнымалысы құрылғаннан бастап, ол жойылғанша өмір сүреді

40. *a* атты бүтін айнымалының дұрыс сипатталуы:

1. **int *a*;**
2. *a* int;
3. integer *a*;
4. float *a*;
5. char *a*;

41. Блоктан тыс анықталған айнымалы қалай аталады?
1. ауқымды (глобальды)
2. жергілікті
3. символды
4. тұрақты
5. жергіліксіз

42. Қай нұсқада sizeof жазылуы дұрыс?
1. sizeof тип
2. sizeof [тип]
3. sizeof (тип)
4. sizeof: тип
5. sizeof_ тип

43. Логикалық терістеу қай вариантта дұрыс берілген?
1. -
2. ~
3. =
4. !
5. --

44. extern спецификаторы нені білдіреді?
1. айнымалы программа ішінде анықталғанын
2. айнымалы программадан тыс анықталғанын
3. айнымалы анықталмағанын
4. айнымалы автоматты түрде анықталғанын
5. дұрыс жауабы жоқ

45. 12 бүтін саннан тұратын *a* массивінің дұрыс сипатталуы:
1. float *a*[12];
2. int *a*[12];
3. float *a*[10];
4. char *a*[12];
5. int *a*(12);

46. Функция тұлғасы қандай таңбалармен қоршалып тұрады?
1. ( )
2. / /
3. { }
4. * *
5. " "

47. 0...255 сандар диапазоны қай типке жатады?
1. double
2. unsigned char
3. bool
4. float
5. signed char

48. Goto нұсқауының дұрыс жазылуы:
1. Goto 65
2. Goto m65
3. Goto белгі
4. Goto ?65
5. Goto printf

49. Келесі операциялардың қайсысы тренарлы (үш операндты)?
1. !
2. &
3. <=
4. !=
5. ?:

50. Си тілінде комментарийді қандай таңбалар ішінде жазады?
1. // //
2. (* *)
3. /* */
4. < >
5. { }

51. Программаның барлық объектілері үшін ортақ мәліметтерді сақтау үшін қандай класс қолданылады?
1. статикалық
2. тұрақты
3. рекурсивті
4. айнымалы

5. ауқымды (глобальді)

52. Егер short пен int типтері араласса, нәтижесі – ... болады
char;
short;
int;
float;
double.

53. Нұсқауыштарды сипаттау форматы қай вариантта дұрыс көрсетілген?
1. аты *типі
2. *аты типі
3. типі *аты
4. & аты типі
5. тип_аты

54. float *a*[10]; нені білдіреді?
1. 1-ден 10-ға дейінгі бүтін сандар тізбегін сипаттау
2. 1-ден 10-ға дейінгі нақты сандар тізбегін сипаттау
3. 10 нақты саннан тұратын *a* массивін сипаттау
4. 10 мәнді қабылдай алатын *a* жиымын сипаттау
5. 10x10 өлшемді *a* матрицасын сипаттау

55. switch операциясынан шығу үшін қай нұсқау қолданылады?
1. do
2. break
3. case
4. return
5. default

56. Типтерді түрлендіру мысалы. Программа орындалғанда экранға не шығады:

```
main ()
{ char ch;
  int i; float fl;
  fl=i=ch='A';
  printf(" %c  %d  %6.2f\n",ch,i,fl);}
```

1. **A 65 65.00 ;**
2. B 66 65.000;

3. B 63 63.000;
4. B 62 62.000;
5. A A 65.00;

57. *= операциясы нені білдіреді?
1. көбейту
2. меншіктеу
3. әр элементін көбейту
4. көбейтіп барып меншіктеу
5. комментарий

58. Лексема дегеніміз не?
1. сөз тіркестері
2. сөйлемдер
3. тілдің өзіндік мағынасы бар ең кіші бірлігі
4. программамның денесі
5. бетті ауыстыру

59. " int n;" жазуы нені білдіреді?
1. n класы берілгенін
2. n спецификаторы берілген
3. n айнымалысы нақты мән қабылдайды
4. n айнымалысы бүтін мән қабылдайды
5. дұрыс жауабы жоқ

60. string.h – бұл мынадай функциялар кітапханасы
1. сөз тіркесі функциялары
2. стандартты функциялар
3. енгізу/шығару базалық кітапханасы
4. графикалық функциялар
5. буфермен жұмыс істеу функциялары

61. graphics.h - бұл мынадай функциялар кітапханасы
1. сөз тіркесі функциялары
2. стандартты функциялар
3. енгізу/шығару базалық кітапханасы
4. графикалық функциялар
5. буфермен жұмыс істеу функцияры

62. Си тілінде айнымалылар программаның мынадай аймағында сипатталады

1. басында
2. аяғында
3. ортасында
4. кез келген жерінде
5. еш жерінде сипатталмайды

63. Си тілінде бір таңбамен белгіленген бас және кіші әріптер мәні...
1. әр түрлі болып саналады
2. бірдей болып саналады
3. компиляторға байланысты өзгереді
4. тілдің нұсқасына байланысты өзгереді
5. барлығы да дұрыс

64. Айнымалыларға, тұрақтыларға, мәліметтер типіне және функцияларға қойылған атау былай аталады...
1. идентификаторлар
2. түйінді сөздер
3. директивалар
4. нұсқауыштар
5. дұрыс жауап көрсетілмеген

65. Келесі сөздер ішіндегі нақты сандар типтерін анықтайтын түйінді сөздерді көрсету керек:
1)char, 2) int, 3) float, 4) double, 5) long, 6) long double, 7)short, 8) signed, 9) unsigned.
1. 1,4,6,7
2. 2,3,4,8
3. 1,7,8,9
4. 3,4,6
5. 3,4

66. Келесі сөздер ішіндегі бүтін сандар типтерін анықтайтын түйінді сөздерді көрсету керек:
1)char, 2) int, 3) float, 4) double, 5) long, 6) long double, 7)short, 8) signed, 9) unsigned.
1. 1,2,5,7,8,9
2. 5,6,7,9
3. 1,2,3,6,9
4. 2,5,7,9
5. 2,7,8,9

67. Директиваның дұрыс жазылған жолын көрсетіңіз:
1. #define PI = 3.1415
2. #define PI == 3.1415
3. #DEFINE = 3.1415
4. #define PI = 3.1415;
5. #define PI 3.1415

68. sizeof() операциясы мынаны анықтайды:
1. литерлік өрнекті
2. константалық тұрақты өрнекті
3. операнд типіне бөлініп берілетін байттар саны
4. операнд орналасатын байттар саны
5. логикалық өрнек

69. Программа нәтижесі нешеге тең болады?
```
void main()
{int i = 5;
  switch (i++) {case 5: printf("%d %d", 1 , i); break;
                 case 6: printf("%d %d", 2 , i); break;
                default: printf("%d %d", 3 , i); }}
```
1. 1 5
2. 2 6
3. 2 5
4. 3 6
5. 1 6

70. Программа нәтижесі нешеге тең болады?
```
void main()
{int i = 5;
  switch (++i) {case 5: printf("%d%d", 1 , i); break;
                 case 6: printf("%d%d", 2 , i); break;
                default: printf("%d%d", 3 , i); }}
```
1. 1 5
2. 2 6
3. 2 5
4. 3 6
5. 1 6

71. int a[5] массивінде қандай элемент жоқ?
1. a[1]
2. a[2]

3. a[3]
4. a[4]
5. a[5]

72. Төмендегі функция арқылы файлға мәлімет жазылады…
1. fread()
2. fscanf()
3. gets()
4. fprintf()
5. seek()

73. Файлдан мына функция көмегімен мәлімет оқылады...
1. fprintf()
2. fscanf()
3. fread()
4. fwrite()
5. seek()

74. Файлды одан мәлімет оқу үшін ашу (режимін көрсету керек)
1. r
2. w
3. a
4. r+
5. w+

75. Мәлімет жазатын файл ашу қажет, егер файл бұрыннан бар болса, ондағы мәлімет жойылады (режимін көрсету керек)
1. r
2. w
3. a
4. r+
5. w+

76. Мәлімет қосу: файл соңына мәлімет жазу үшін оны ашу қажет (режимін көрсету керек)
1. r
2. w
3. a
4. r+
5. w+

77. Файлдағы мәліметті оқып, оған мәлімет жазу: оқу және жазу (режимін көрсету керек)
1. r
2. w
3. a
4. r+
5.w+

78. Файлды жаңарту үшін оны ашу, егер файл бұрыннан болса, оның мәліметі жойылады (режимін көрсету керек)
1. r
2. w
3. a
4. r+
5. w+

79. Мәлімет қосу: файлды ашып, оны жаңартады; мәлімет бұрнғының соңына жазылады (режимін көрсету керек)
1. r
2. w
3. a
4. r+
5. a+

80. Төмендегі функция файлдан сөз тіркесін оқиды
1. fread()
2. fgets()
3. fgetc()
4. fputs()
5. fputc()

81. Төмендегі функция файлға сөз тіркесін жазады
1. fwrite()
2. fgets()
3. fgetc()
4. fputs()
5. fputc()

82. printf() функциясының сөз тіркесін шығару кезіндегі форматы
1. %d
2. %c

3. %s
4. %f
5. %x

83. printf() функциясының нақты сан шығару кезіндегі форматы
1.%d
2.%c
3.%s
4.%f
5.%x

84. printf() функциясының таңбасыз ондық бүтін сан шығару кезіндегі форматы
1. %d
2. %u
3. %o
4. %x
5. %f

85. printf() функциясының таңбасыз сегіздік бүтін сан шығару кезіндегі форматы
1. %d
2. %o
3. %x
4. %c
5. %f

86. printf() функциясының таңбасыз он алтылық бүтін сан шығару кезіндегі форматы
1. %d
2. %o
3. %x
4. %c
5. %f

87. printf() функциясының экспоненциал нақты сан шығару кезіндегі форматы
1. %d
2. %o
3. %e
4. %c

5. %f

88. Егер **char** мен **short** типтері араласса, нәтижесі – ... болады
1. char;
2. **short ;**
3. float;
4. long ;
5. int;

89. Мәліметтердің бір еселік дәлдікпен берілетін нақты санды типі былай сипатталады...
1. float
2. double
3. char
4. int
5. struct

90. Мәліметтердің екі еселік дәлдікпен берілетін нақты санды типі былай сипатталады...
1. float
2. double
3. char
4. int
5. struct

91. Мәліметтердің символдық типі былай сипатталады...
1. float
2. double
3. char
4. int
5. struct

92. Тақырып файлдары нұсқаулары мына жерде орналасады...
1. main() функциясы алдында
2. main() функциясы артында
3. main() функциясы ішінде (тұлғасында)
4. жеке файлда
5. керекті функция шақырылатын блокта

93. < және > символдарымен қоршалып тұратын тақырып файл аттары компиляторға бұл файл мынадай каталогта орналасқанын білдіреді...
1. LIB
2. BIN
3. INCLUDE
4. BGI
5. TVISION

94. “ және “ символдарымен қоршалып тұратын тақырып файл аттары компиляторға бұл файл мынадай каталогта орналасқанын білдіреді...
1. LIB
2. BIN
3. INCLUDE
4. в текущем
5.TVISION

95. Си программасы мәтіндері қандай типті файлдарда сақталады?
1. .txt, .doc
2. .cpp, .c
3. .pas, .bas
4. .h, .htm
5. .html, .xtml

96. Программалардың объектілік кодтары қандай типті файлдарда сақталады?
1. .exe
2. .obj
3. .cpp, .c
4. .com
5. .txt

97. Программалардың атқарылатын екілік кодтары қандай типті файлдарда сақталады?
1. .exe
2. .com
3. .obj
4. .cpp, .c
5. .txt

98. Түйінді сөздер (ключевые слова) дегеніміз не?
1. қарапайым идентификаторлар
2. арнайы қорға енгізілген сөздер
3. операциялар
4. функциялар
5. макрокомандалар

99. Си тілінің int типті бүтін саны 16 разрядты процессорда қандай орын алады?
1. 2 байт;
2. 4 байт;
3. 8 байт;
4. 6 байт;
5. 10 байт;

100. Си тілінің int типті бүтін саны 32 разрядты процессорда қандай орын алады?
1. 2 байт;
2. 4 байт;
3. 8 байт;
4. 6 байт;
5. 10 байт;

ҚОЛДАНЫЛҒАН ӘДЕБИЕТТЕР

1. Березин Б.И., Березин С.Б. Начальный курс С и С++ . –М: Диалог-МРТИ ,1999.-288с.
2. Керниган Б., Ритчи Д. Язык программирования Си.-М.: Финансы және статистика.,1992.-271с.
3. Касаткин А.И., Вольвачев А.Н. Профессиональное программирование на языке Си: От Turbo –С к Borland С++: Справочное пособие – Мн.:ВШ,1992.-240с.
4. Страуструп Б. Язык программирования С++. 2-е изд.:В 2т. Киев: Диа Софт,1993.
5. Фьюэр А. Задачи по языку СИ. М.:Финансы и статистика.1985.
6. Хэнкок Л., Кригер М. Введение в программирование на языке СИ. М.:Радио и связь.1986.
7. Берри В., Микинз Б. Язык СИ: введение для программистов. М.:Финансы и статистика.1988.
8. Уэйт М., Прама С., Мартин Д. Язык СИ. Руководство для начинающих. М.:Мир. 1988.
9. Больски М.Н. Язык программирования СИ. Справочник. М.:Радио и связь. 1988.
10. Юлин В.А., Булатова И.Р. Приглашение к СИ. Мн.:Высш. школа.1990.
11. Р.Уингер. Язык Турбо СИ. М.:Мир.1991.
12. Романовская Л.М., Русс Т.В., Свитковский С.Г. Программирование в среде СИ для ПЭВМ ЕС. М.:Финансы и статистика.1992.
13. Е.М.Демидович. Основы алгоритмизации и программирования. Язык СИ. Мн.:”Бестпринт“, 2001. – 440с.
14. М.А.Аксенкин, О.Н.Целобенок. Язык С. Мн.:”Універсітэцкае”, 1995. – 302с.
15. Г.П.Котлинская, О.И.Галиновский. Программиро-вание на языке СИ. Мн.:”ВШ”, 1991. – 155с.
16. В. В. Подбельский. Язык С++. М.:ФиС, 2001-. 559с.
17. Л.И.Климова.С++. Практическое программирование. М.:Кудиц-Образ, 2001. – 587с.
18. Г.Шилд. Программирование на Borland С++. Мн.:Попурри, 1999. – 800с.
19. В.В.Тимофеев. Программирование в среде С++ Builder 5. М.:БИНОМ, 2000.
20. Орысша-қазақша сөздік. Жалпы редакциясын басқарған Мұсабаев Ғ.Ғ., екі томдық, Алматы, 1978ж.
21. Балапанов Е., Бөрібаев Б., Бекбаев А., Дәулетқұлов А., Спанқұлова Л. Информатика терминдерінің қазақша -ағылшыншы - орысша, орысша – қазақша - ағылшынша, ағылшынша – орысша - қазақша сөздігі.- Алматы: Сөздік-Словарь,1998.-176 б.

МАЗМҰНЫ