1-лекция

Жоғары деңгейлі бағдарламалау тілдеріне шолу. Алгоритм түсінігі. Алгоритмдердің қасиеттері мен түрлері. С программалау тілінің негізгі түсініктері. С/С ++ тілінің алфавиті және негізгі түсініктері. Мәліметтер типтері

Дәріс беруші:

т.ғ.к., доцент м.а. Абдувалова Айнур Джумабаевна

Сұрақтар:

1. Алгоритм, программа ұғымы.
2. Алгоритм қасиеттері
3. Алгоритмнің өрнектелу жолдары
4. Алгоритмдердің бірыңғай құрылымдары
5. Сызықтық алгоритмдер
6. Тармақталу алгоритмдері
7. Циклдік алгоритмдер

Сұрақтар:

1. Жалпы түсініктер
2. Си тілінің алфавиті
3. Тілдің қарапайым объектілері
4. Стандартты функциялар

1. Алгоритм, программа ұғымдары

Алгоритм атауы атақты шығыс математигі абу Жафар Мұхаммед ибн Мұса әл-*Х*орезми (763-850 ж.) есімінің латынша Algorithmi (Алгорит-ми) болып жазылуынан шыққан. Ол санаудың ондық жүйесінде көпорынды сандармен ариф-метикалық амалдардың орындалу ережесін ұсынған. Бұл ережелер қосынды мен көбейтін-діні табуға арналған амалдарды орындауға қа-жетті тізбектен құрылған. Сол ереже осы күнге дейін қолданылып келеді.

Алгоритм – берілген есептің шығару жолын реттелген амалдар тізбегі түріне келтіру.

Алгоритмді орындаушының рөлін негізінен адам немесе компьютер, робот т. б. атқарады. Мысалы, $y = (ax+b)(cx-d)$ функциясын есептеу төмендегі қарапайым 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 y деп белгілеу.

Алгоритмге күнделікті тұрмыстан алып бір мысал келтіре кетейік. Студент болу үшін алгоритмнің мынадай қадамдарын орындау керек.

1. Орта мектепті бітіріп, тест тапсыру.

2. Керекті құжаттарды тест нәтижесімен бірге белгілі бір жоғары оқу орнына (колледжге, институтқа, университетке) өткізу.

3. Конкурстан өту.

Техникалық құрылғыларды дұрыс пайдалану үшін есептің шешу жолы, яғни орындалатын әрекеттердің тізбегі әрі түсінікті, әрі дәл болуы қажет.

Берілген мәселенің шешу жолдарының түсініктілігін оның алгоритмінің түсініктілігі деп қарастырады. Алгоритмде алдыңғы әрекеттің нәтижесі келесі әрекетте пайдаланылады.

Алға қойған мақсатқа жету немесе берілген есепті шешу бағытында атқарушыға біртіндеп қандай әрекеттер жасау қажеттігін әрі түсінікті, әрі дәл етіп көрсететін нұсқаулар тобын алгоритм деп атайды.

Төмендегі алгоритм анықтамаларын салыстырыңыздар:

1. Алгоритм – алғашқы берілген мәліметтерді пайдаланып нақты нәтижеге қол жеткізетін шекті командалар тізбегін орындауда атқарушыға түсінікті және нақты нұсқаулар.

2. Алгоритм дегеніміз – берілген мәндерді пайдаланып қажетті нәтижеге жетуді жүзеге асыратын әрекеттердің орындалу ережесі.

3. Алгоритм дегеніміз – алғашқы берілген мәліметтерді пайдаланып, қойылған мақсатқа жетуге немесе мәселені шешуге (есепті шығаруға) бағытталған әрекеттердің орындалуын жүзеге асыратын атқарушыға түсінікті және нақты нұсқаулар тізбегі.

Алгоритмді компьютерде орындау үшін оны программа түрінде жазып шығу керек.

Программа – алгоритмді машинаға түсінікті нұсқаулар тізімі (командалар) ретінде жазу.

Программа машинаға түсінікті командалардан тұрады. Осы командалар тізбегі орындалу барысында есептің нәтижесі шығады. Әрбір компьютер алдын ала жазылған программамен жұмыс істейді.

Программа дегеніміз – белгілі бір нәтиже алу үшін орындалатын командалдардың айқындалған тізбегі.

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; нәтижені экранға, қағазға басып шығару командасы; келесі командаларға көшу тәртібін орындау т.с.с.

Компьютердің жұмысы программалық принципке негізделген, яғни ол өзінің жадында сақталатын командалар тізбегін автоматты түрде орындау арқылы есеп шығарады.

Компьютер берілген тапсырманы орындауға дайын тұрған техникалық аспап болғандықтан, әрбір тапсырманы түсінікті түрде қысқаша жаза білу қажет. Тапсырма жоғарыда айтылған жекеленген командалардан тұрады. Машинаға түсінікті түрде жазылған тапсырмаларды немесе командалар жиынын да программа деп атауға болады.

Программа – арнайы мәтін арқылы компьютерге тапсырманың ретті кезегін хабарлайтын ережелер мен нұсқаулар тізбегі.

Алгоритмдік тіл – алгоритмдерді жазуға арналған символдар мен сол символдардан тұратын конструкцияларды құрастыру және түсіндіру ережелерінің жиыны.

Алгоритмдеу дегеніміз – есепті шығару алгоритмін құрастыру процесі.

Мәліметтер дегеніміз – белгілі бір процесс көмегімен тасымалдап, өңдеуге болатын, формальды түрде бейнеленген фактілер мен идеялар.

Программалау тілі компьютерлерде программаларды орындау ісін атқарады.

2. Алгоритм қасиеттері

Алгоритмнің мәнін ашатын негізгі қасиеттері немесе оған қойылатын талаптар болады. Олар:

1) детерминділік (анықтылық, бір мәнділік) – басқаша түсінуге жол бермей, тек қана көрсетілген әрекеттерді айқын түрде орындауға арналған нұсқаулар дәлдігі, яғни алгоритм анық, әрі дәл өрнектелуі тиіс;

2) оның модульдік (бөлікке бөліну) қасиеті, яғни алгоритмді шағын бөліктерге бөлу мүмкіндігі болуы қажет, яғни есептеу процесін жекеленген қарапайым операцияларға бөлу қасиетінің болуы;

3) оның нәтижелілік (шектеулілік) қасиеті алгоритм шектелген уақыттан соң нәтиже беруі тиіс, яғни алгоритм қадамдарының саны шексіз болмауы керек;

4) бір типтегі (біртектес) есептерге жалпы бір ғана алгоритм қолданылуы тиіс – жалпылық қасиеті.

3. Алгоритмнің өрнектелу жолдары

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

1) табиғи тіл арқылы жазу;

2) белгілі бір түйінді сөздер – терминдер (псевдокодтар - жалған кодтар) арқылы қысқаша тізбекті түрде жазу, мұны қарапайым алгоритмдік тіл деп те айтады;

3) график жолымен (блок-схема арқылы) жазу;

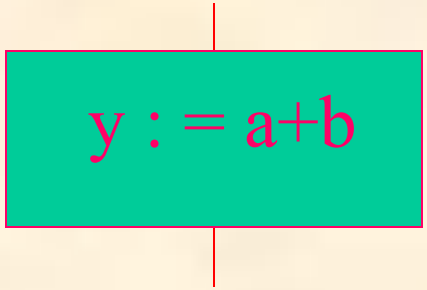
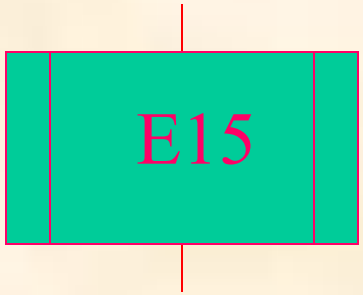
4) программалау тілдерінде жазу жолдарын жатқызуға болады.

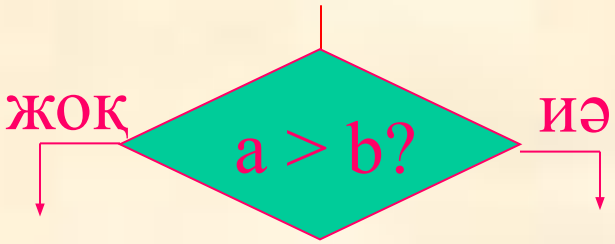
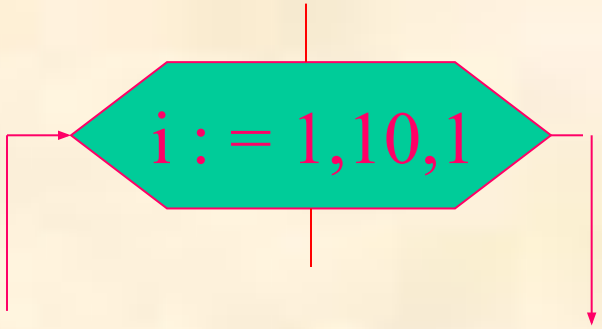
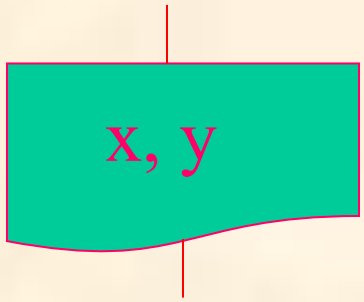
Алгоритмді табиғи тілде өрнектеу компьютерде қолданылмайды, өйткені онда дәлдік, нақтылық болмайды.

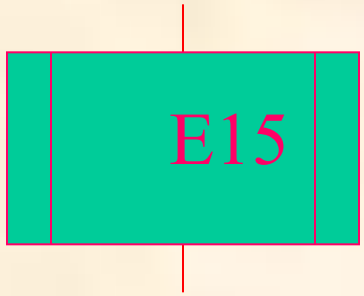
Ал алгоритмді екінші көрсетілген жолмен өрнектеу қарапайым алгоритмдік тіл деп аталып кеңінен қолданылып жүр. Мұны олардың ағылшын тіліне негізделіп жасалған программалау тілдеріне жақындығымен түсіндіруге болады.

Алгоритмдерді график жолымен жазу, онан кейін оны программалау тіліндегі программаға айналдыру істері мемлекеттік стандартпен бекітіліп ақпарат өңдеу жұмысында кеңінен қолданылып келеді.

Алгоритмдерді график жолымен жазу

Іс-әрекеттің аты	Блоктың түрі	Атқаратын жұмысы
Процесс		Математикалық өрнектерді есептеу
Бастау, аяқтау		Алгоритмдерді бастау, аяқтау
Қосалқы программа		Қосалқы программаларға кіру және шығу

Таңдау		Есеп шығару жолын таңдау
Модификация		Цикл (қайталау) басы
Құжат		Нәтижені баспаға (қағазға) шығару

Енгізу, шығару		Мәліметтерді енгізу, шығару
Түсініктеме беру		Схеманы, формулаларды түсіндіру
Қосалқы программа		Қосалқы программаларға кіру және шығу

Алгоритмдерді график арқылы бейнелеу 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 немесе операциялар символы деп те аталады. Блоктар бағытталған сызықтармен байланысып, бірінен соң бірі орналасады.

4. Алгоритмдердің бірыңғай құрылымдары

Кез келген алгоритмді (программаны) блоктардың өзара байланысуына қарай төмендегідей үш түрлі басқару құрылымын пайдалану арқылы жазып шығуға болатындығы дәлелденген:

- **сызықтық құрылым немесе тізбектелген әрекеттер тізбегі;**
- **тармақты құрылым немесе шартты тексеру;**
- **қайталау немесе циклдік құрылым.**

Осындай негізгі (канондық) құрылымдардан тұратын алгоритмді регулярлық алгоритм (программа) деп атайды, олардың бір ғана кіру нүктесі мен бір ғана шығу нүктесі болады. **Осы үшеуі құрылымдық программалаудың негізгі конструкциялары, яғни құраушылары болып саналады.**

Программадағы кез келген әрекетті (операторды) оның кіру нүктесі арқылы тауып орындауға болады (осы тәсілмен табылмайтын операторлар және шексіз циклдер болмауы тиіс). Мұндай алгоритмді – программаны басқару ісі жоғарыдан төмен қарай жүргізіледі. Түсініктеме мәтін (комментарий) қосылған осындай программалар оқуға және түсінуге жеңіл болып есептеледі.

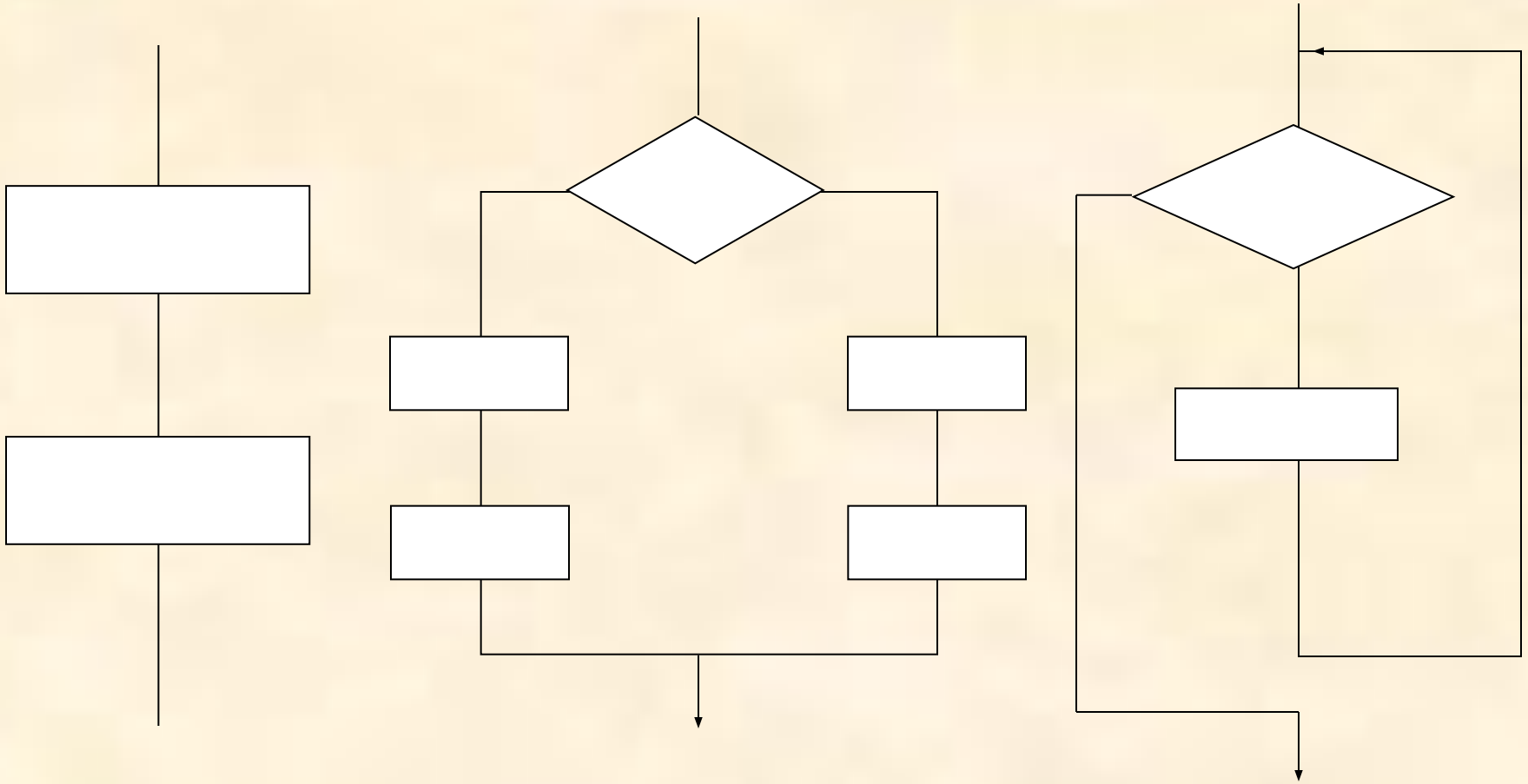
Оператор – тілдің қарапайым сөйлемі, ол белгілі бір әрекет немесе амал орындап, **;** таңбасымен аяқталады.

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

Тармақты құрылым – шартқа байланысты екі оператордың бірінің орындалуы.

Цикл – операторлар бөлігінің бірнеше рет қайталана орындалуы.

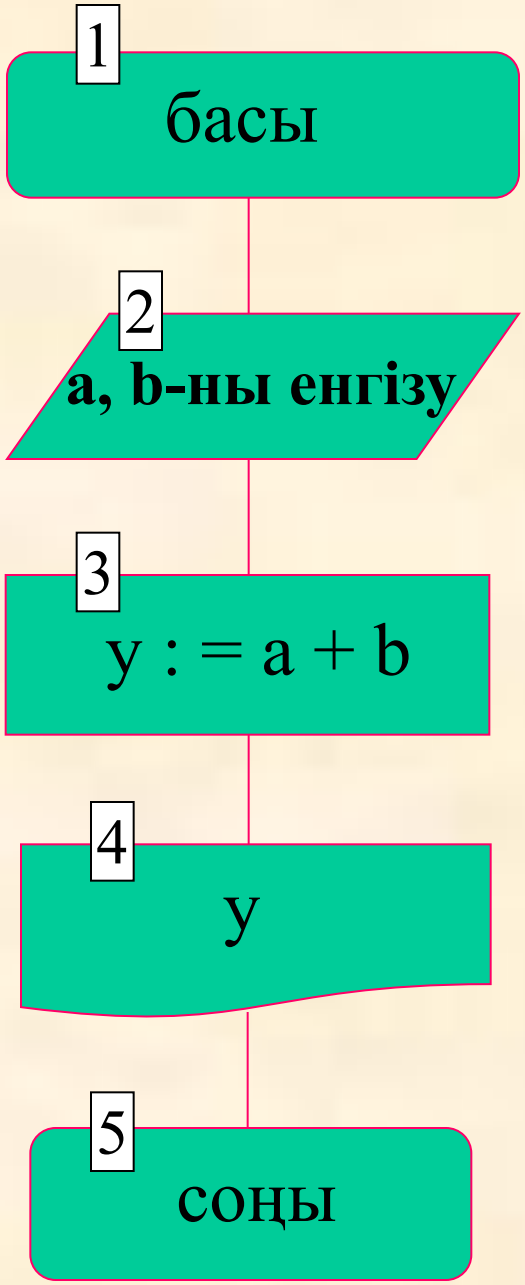
Төменде алгоритмдердің бірыңғай құрылым-дарының схемалық бейнеленуі көрсетілген



5. Сызықтық алгоритм

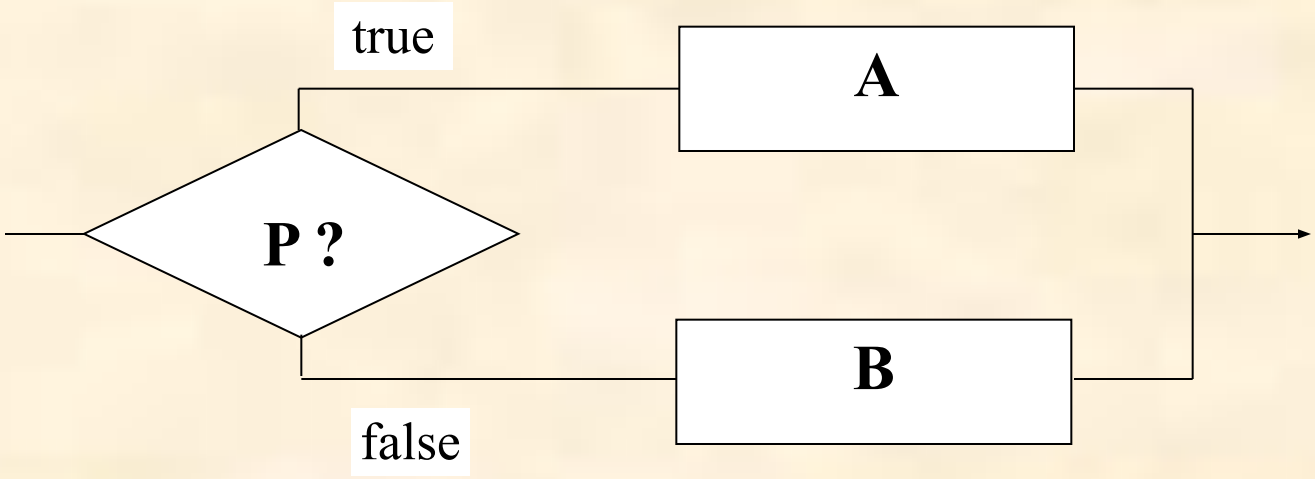
Мұнда *a* және *b*-ның сандық мәндерін ЭЕМ-ге енгізіп (2-блок), содан кейін қосу амалын орындап, ақырында *y*-ті қағазға басып шығарып, жұмысты тоқтатамыз.

y = *a* + *b* формуласы есептеу блогы (3-блок) арқылы өрнектеледі. Ал нәтижені қағазға басу үшін көпбұрышты құжат алу блогын (4-блок) пайдаланып, оның ішіне нәтиженің атауларын жазамыз.



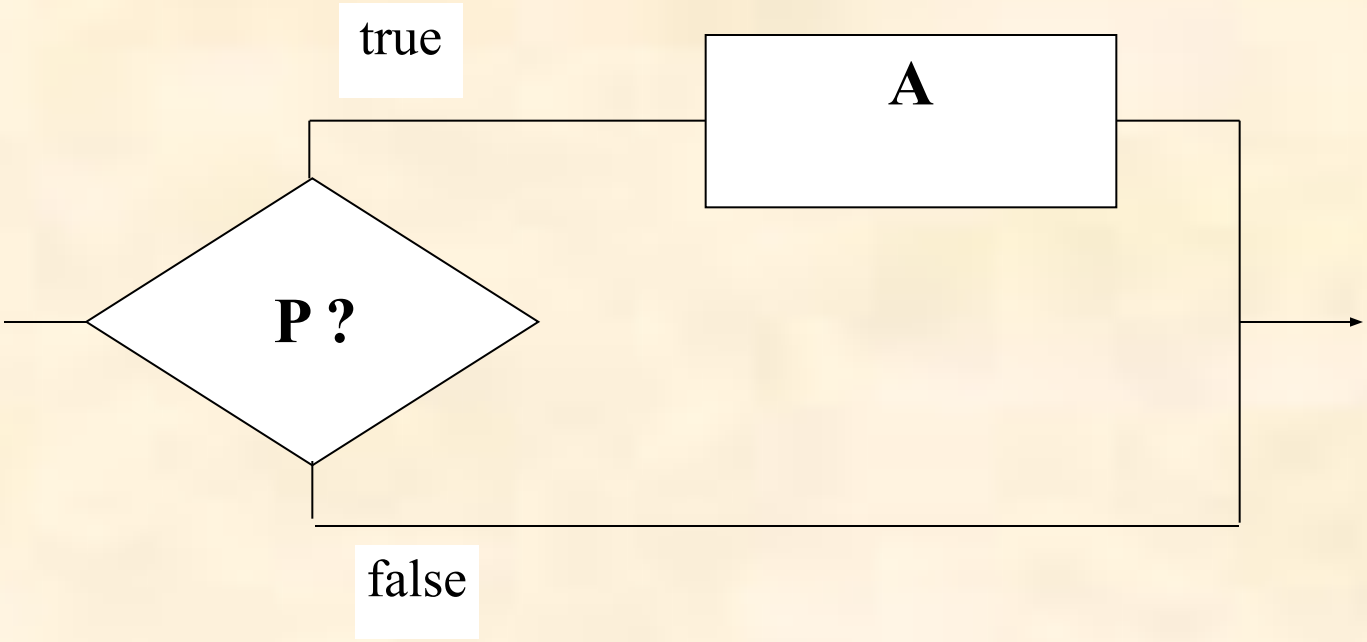
6. Тармақталу алгоритмдері

Тармақталу алгоритмінде арифметикалық теңсіздік (теңдік) түрінде берілген логикалық шарт тексеріледі. **P** шартының мәні ақиқат (true) немесе жалған (false) бола алатын логикалық өрнек түрінде болады. Егер ол орындалса – ақиқат болса, онда алгоритм бір жолмен, ал орындалмаса – екінші жолмен жүзеге асырылады, яғни есепті шығару жолы тармақталып екіге бөлініп кетеді.



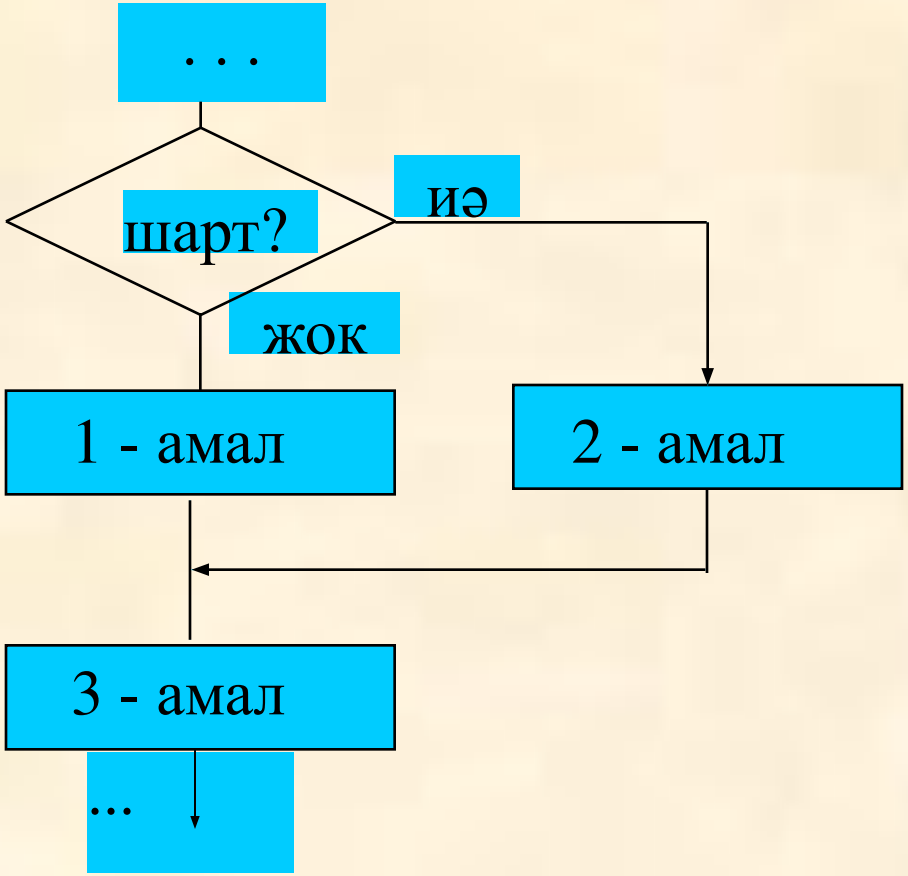
IF P then A else B;

Бұл құрылым толымсыз (қысқаша) түрде болуы мүмкін, онда логикалық өрнектің мәні жалған болғанда ешқандай әрекет орындалмайды. Мұндай құрылым түрі төменде көрсетілген:

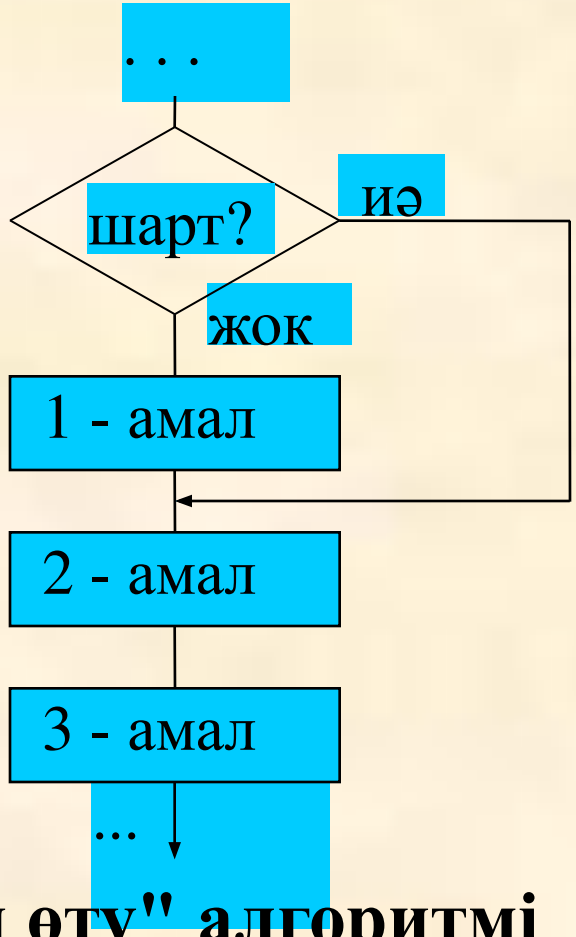


IF P then A;

Тармақталу алгоритмдері осы екі түрде кездеседі, олар "таңдау" және "аттап өту" мүмкіндіктерін іске асыруға көмектеседі.



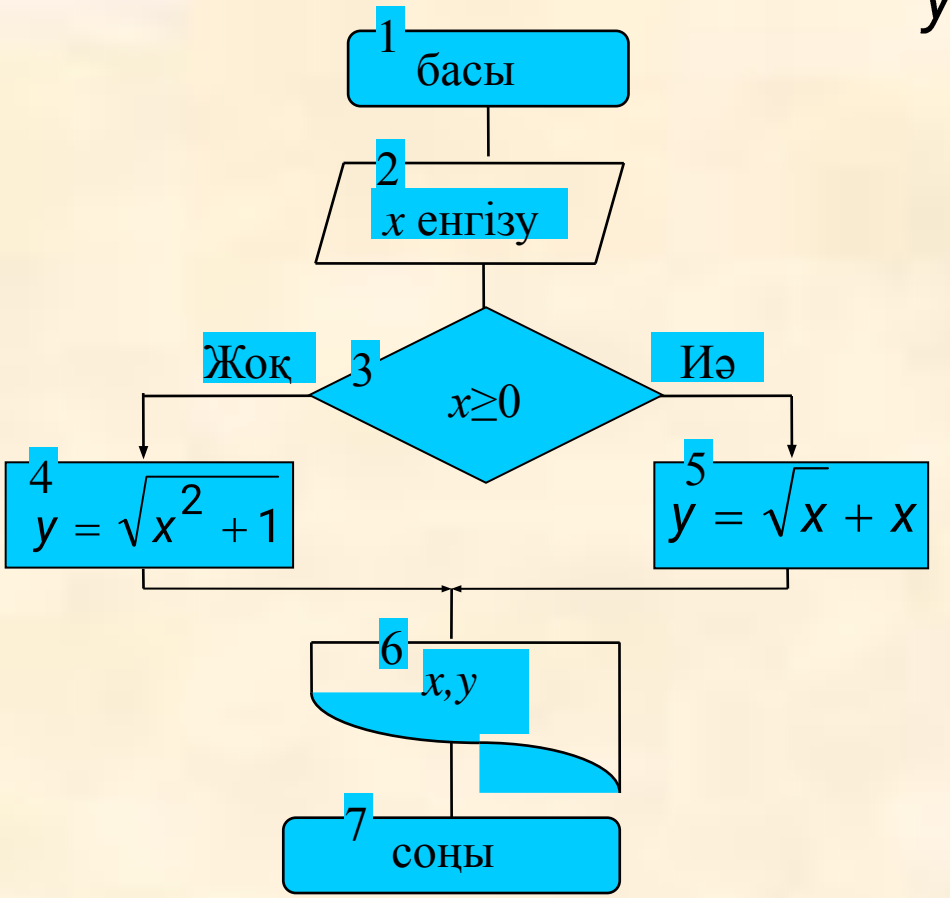
"Таңдау" алгоритмі



"Аттап өту" алгоритмі

1-мысал. *Y* функциясын төмендегі формула бойынша есептеу керек

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



Тармақталу алгоритмі

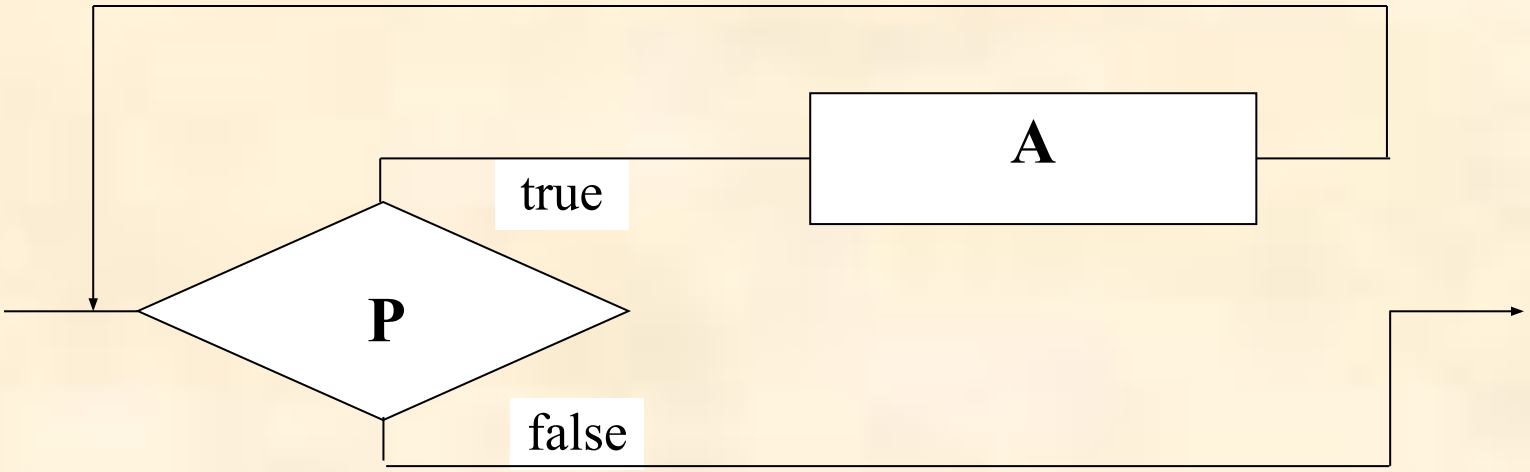
7. Циклдік алгоритмдер

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.

Бірнеше рет қайталанатын бөлігі бар алгоритмдер тобы циклдік алгоритмдерге жатады. Циклдік алгоритмдерді пайдалану оларды кейіннен программаларда цикл операторы түрінде қысқартып жазу мүмкіндігін береді.

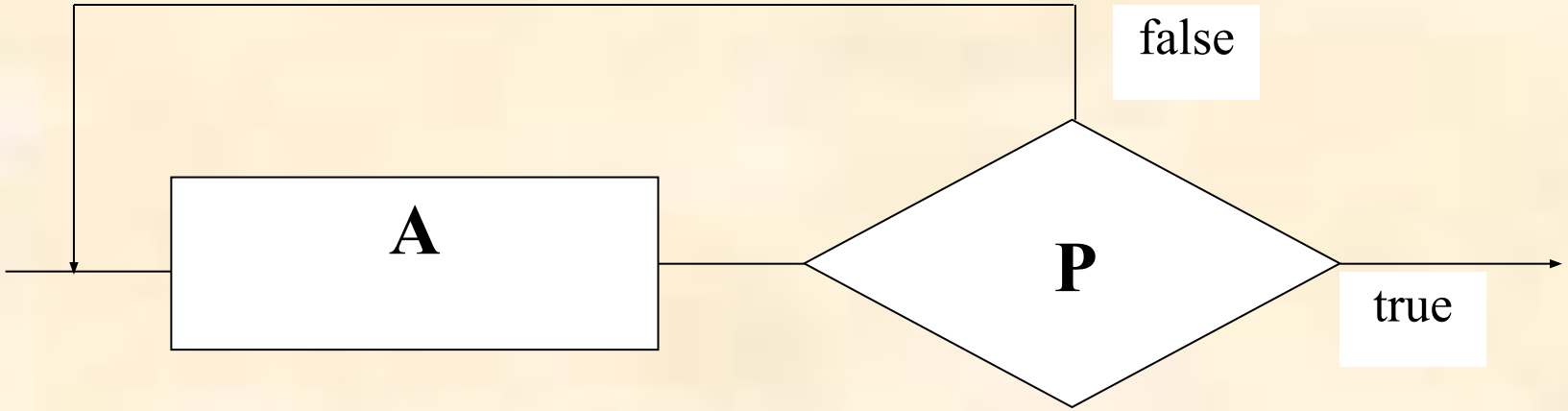
цикл – әзірше



While P do A ;

Мұнда А әрекеті Р шарт мәні ақиқат болып тұрса, қайталана береді. Сондықтан А әрекеті орындалуы кезінде Р-ға әсер ететін айнымалылар мәні өзгеруі тиіс. Бұлай болмаған жағдайда шексіз цикл орын алады. Шарт мәні А әрекетіне дейін анықталады, сол себепті кейде А әрекеті бір де бір рет орындалмауы да мүмкін.

цикл – дейін



цикл – дейін түріндегі қайталау кем дегенде бір рет орындалады, өйткені шарт А әрекетінен кейін тексеріледі. А әрекеті Р мәні ақиқат болған кезде орындалмайтын болады.

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.

Практикада белгілі бір айнымалының сандық мәніне байланысты орындалатын арифметикалық циклдер жиі кездеседі. Мұнда арифметикалық прогрессияға ұқсас болып келетін циклдер ең қарапайым арифметикалық цикл болып табылады. Оны басқару қайталану кезеңінде прогрессияның заңына сәйкес тұрақты шамаға өзгеріп отыратын цикл параметрінің сандық мәнімен байланысты болуы тиіс.

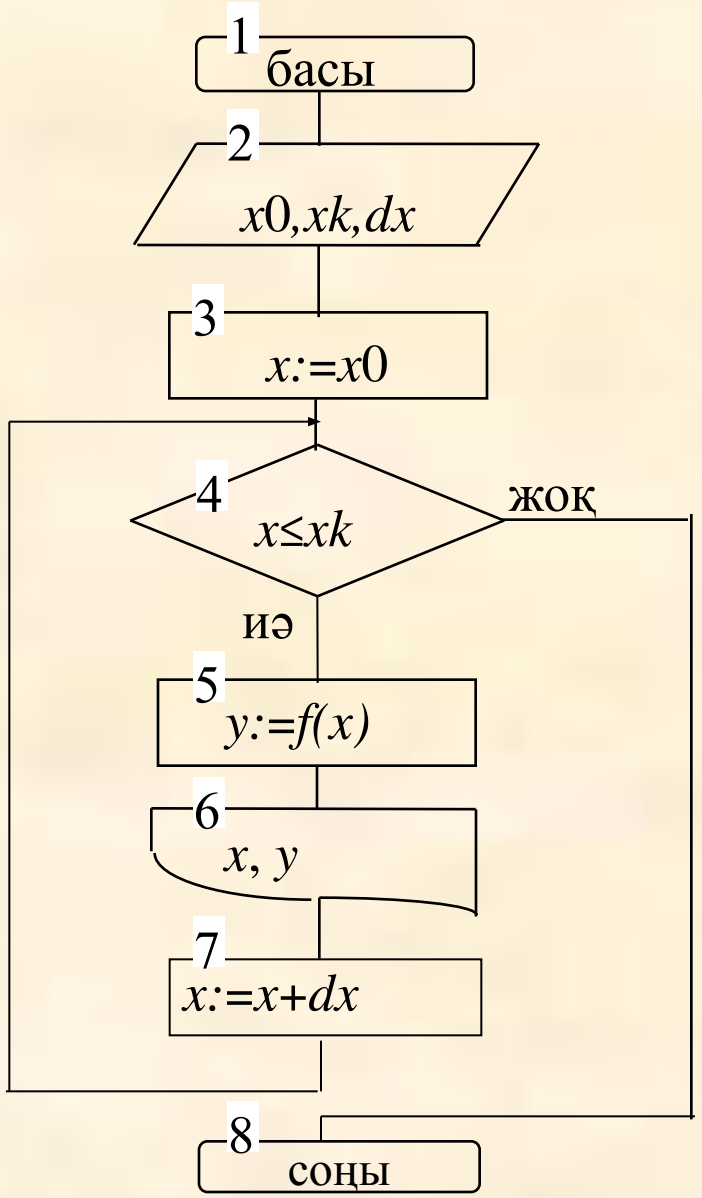
Цикл орындалуы алдында оның айнымалы аргументі – параметрі алғашқы мәнге ие болуы керек, сонан соң қайталау кезінде цикл параметрі белгілі бір шамаға (қадамға) өзгере отырып, ол алдын ала берілген ең соңғы мәнге дейін жетуі қажет.

Алгоритмнің орындалу барысында цикл параметрі, мысалы, x өзінің ең алғашқы $x0$ мәнінен ең соңғы xk мәніне дейін тұрақты шамаға (dx) өзгеріп отырады. Осының нәтижесінде x мынадай мәндерді қабылдайды: $x0, x0+dx, x0+2dx, ..., x0+(n-1)dx, xk,$ мұндағы n – циклдің қайталану саны, ол былай анықталады:

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

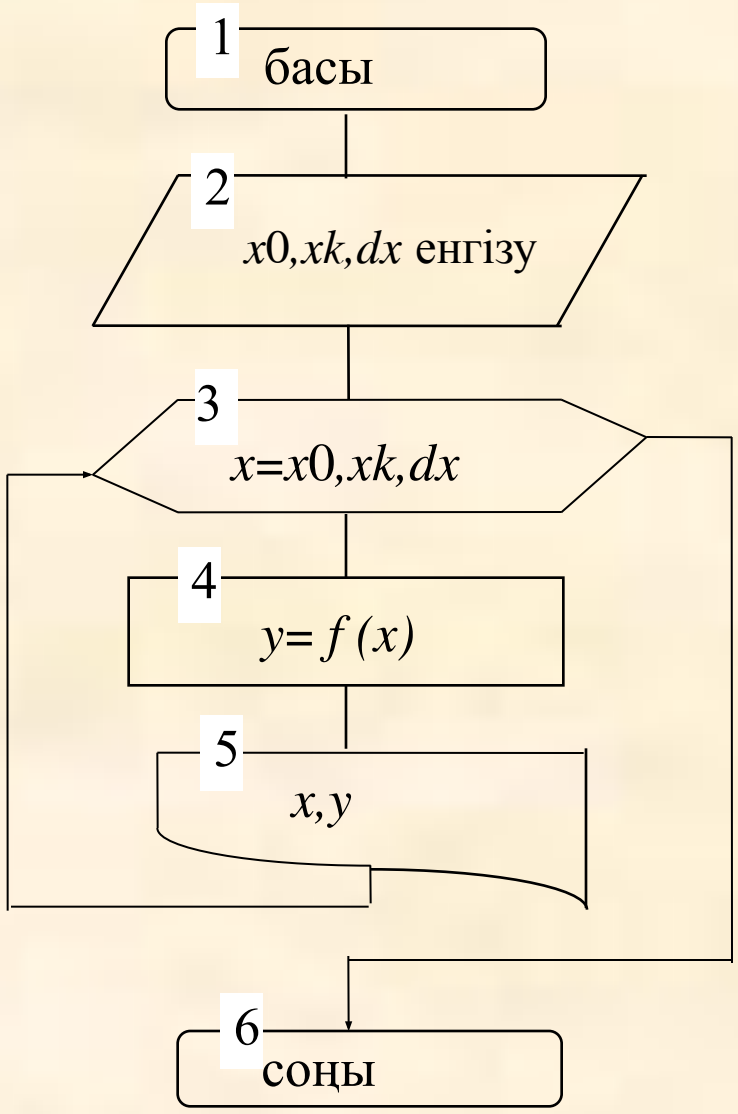
мұнда [...] – өрнектің бүтін бөлігі алынатынын көрсетеді. n әрқашанда бүтін сан болуы тиіс, егер ол аралас сан болса, онда оның бөлшегі алынып тасталады, өйткені циклдің қайталану саны бүтін натуралдық сан болуы тиіс.

Арифметикалық цикл үшін $y=f(x)$ функциясының есептелу жолы алгоритм ретінде 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.



Қарапайым циклдік алгоритм

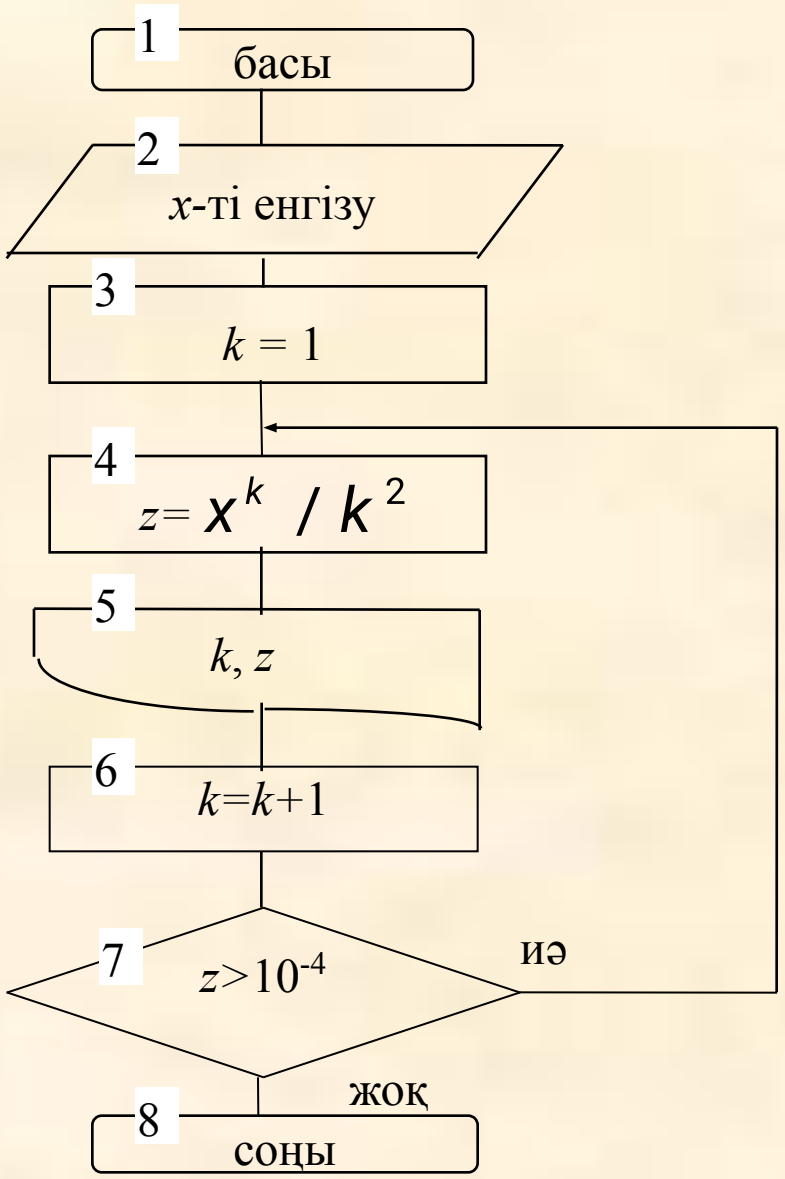
Алгоритм салуды және прог-рамманы жазуды жеңілдету үшін цикл алгоритмдерін "модификатор" немесе "цикл басы" блогын пайдалану арқылы жазылады. Онда алд-ыңғы көрсетілген 3-ші, 4-ші, 7-блоктардың орнына "цикл басы" блогы орналасады. Ол алтыбұрышты фигурадан тұрады және оның міндетті түрде екі кіру және екі шығу сызығы болуға тиіс.



Модификаторлы циклдік алгоритм

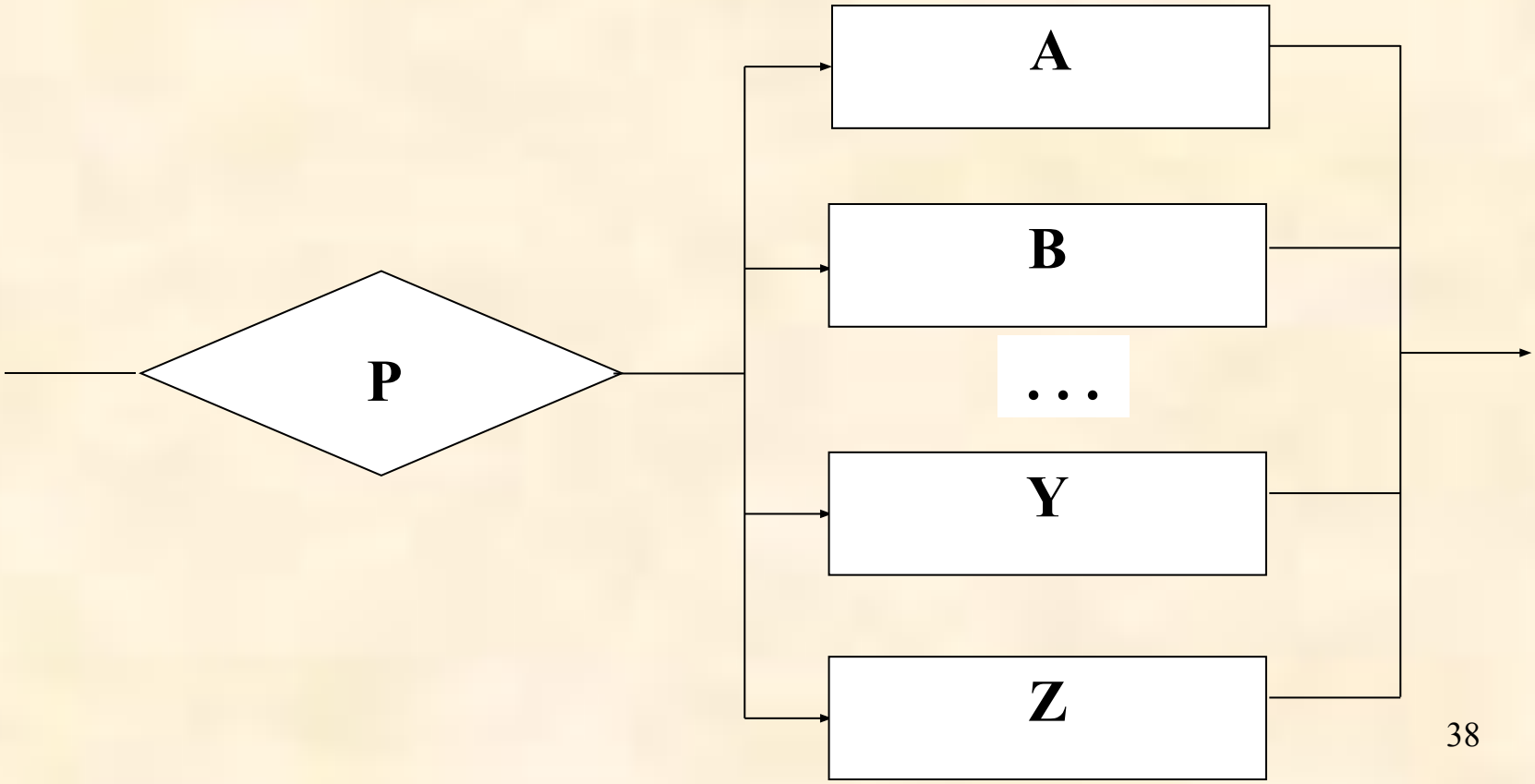
Қадамдық циклдер. Циклді орындаудың алдында, оның қайталану саны белгісіз болған жағдайда қадамдық циклдер пайдаланылады. Мұнда циклді жазу үшін тек қана "шартты тексеру" блогын қолдану қажет, ол циклді аяқтау үшін белгілі бір шартты тексереді. Қадамдық циклдердің схемасын сызғанда модификаторды (алтыбұрышты) 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 $k = 1, 2, 3, \ldots$ 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 функциясы-ның 0.0001-ден артық болуы циклді қайталау шарты болып есептеледі ($Z >$ 0.001). Суретте осы есептің алгоритм схемасы көрсетілген.



Қадамдық цикл алгоритмі

Таңдау – **case** ауыстырғыш (көп тармақты) құрылымы программалауды жеңілдететін мүмкіндік болып табылады. Таңдау құрылымы бірнеше мүмкіндіктердің біреуін ғана орындау кезінде өте қолайлы.



P мәніне байланысты A, B, …, Z әрекеттерінің бірі орындалады да, сонан кейін келесі құрылымдар атқарылады.

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

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

болып бөлінеді.

Жалпы түсініктер

С тілі өткен ғасырдың 70-жылдары басында АҚШ-та Bell Telephon Laboratories компания-сының қызметкері Дэннис Ритчидің бастауы-мен дүниеге келді. Бұл тілдің негізі Алголдан басталып, С және ПЛ/1 тілдерімен қатар пайда болды.

С тілінің шығуы UNIX операциялық жүйесінде программалаумен тығыз байланысты, өйткені бұл жүйе ассемлерде және осы С тілінде жазылып шыққан болатын.

Бұл тілде жазылған программаны компьютерде орындау кезінде ол алдымен *трансляция* сатысынан өтіп (машина тіліне аударылып), объектілік программа түріне ауысады да, сонан кейін барып орындалады. Осы сәтте компьютерде программаның екі нұсқасы болады, оның біріншісі – С тіліндегі *алғашқы нұсқасы*, ал екіншісі – *объектілік кодтағы машина тілінде жазылған программа*. Есептің нәтижесін тек машиналық кодта жазылған программа арқылы аламыз, ал программаны түзету қажет болғанда оның алғашқы нұсқасы өңделіп, оны қайта түрлендіру сатысы жүзеге асырылады.

Жалпы Си тілінің даму жолына қарасақ:

Алгол-60 – 1960-ж. халықаралық комитет жасап шығарды

CPL – (Combined Programming Language) Кембриджде және Лондон университетінде 1963 ж. қатарласа жасалды

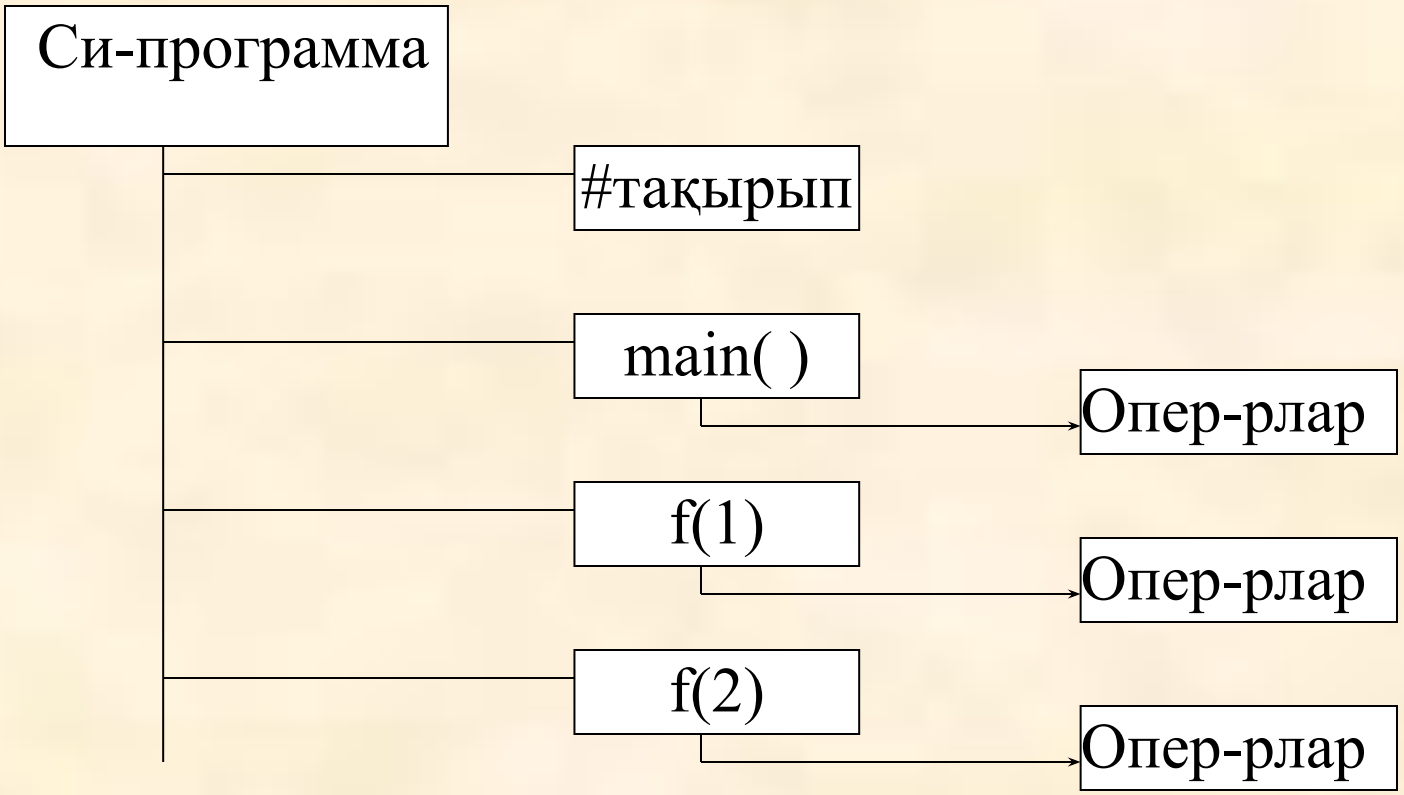
BCPL – (Basic Combined Programming Language) Кембриджде Мартин Ричардс 1967 ж. жасап шығарды

B – Bell Labs қызметкері Кен Томпсон 1970 ж. жасады

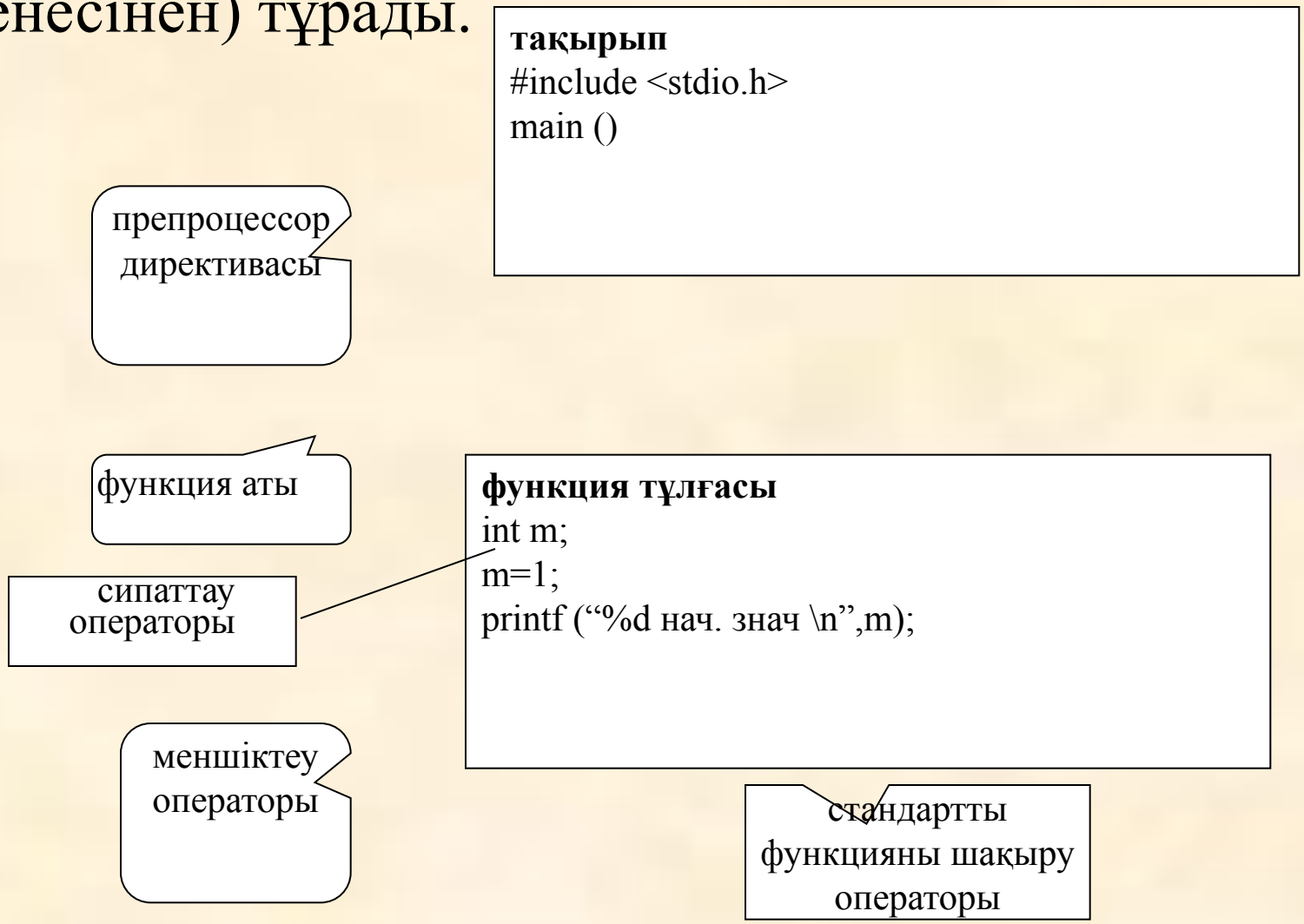
C – Bell Labs қызметкері Дэннис Ритчи 1972 ж. жасады

Сонымен, 1983 ж. Си тілі стандартын жасау үшін (ANSI C) Америка ұлттық стандарттар институты (ANSI) құрылды.

Кез келген С-программа бір немесе бірнеше функция-лардан тұрады. Олар программа құруға керекті негізгі модульдер болып табылады. Келесі суретте С прог-раммасының жалпы құрылымы көрсетілген.



Си программасы бірнеше функциялардан құралады, олардың ішінде міндетті түрде main() болуы қажет. Функция мәтіні оның тақырыбы мен тұлғасынан (денесінен) тұрады.



Си тіліндегі программа жеке-жеке жолдардан тұрады. Оларды теру, түзету арнайы мәтіндік редакторлар арқылы атқарылады. Программа қатарларының алдындағы азат жол немесе бос орындар саны өз қалауымызша алынады.

Бір қатарға бірнеше командалар немесе операторлар орналаса алады, олар бір бірінен нүктелі үтір (;) арқылы ажыратылып жазылады, бірақ бір жолда бір ғана оператор тұрғаны дұрыс, әрі түзетуге жеңіл, әрі ыңғайлы болып саналады.

Программа мысалы:

```
/* x-тің оң және теріс мәндері үшін y функциясын есептеу
        y = sqrt(x*x+1)+abs(x), егер x<0;
        y = 3*x+4,                    егер x>=0  */
#include <conio.h>  /* экранмен жұмыс істеу директивасы*/
#include <stdio.h>  /* енгізу-шығару директивасы */
#include <math.h>  /* матем. функциялар директивасы*/
main()
{
  int x; float y;
  textcolor(GREEN);              /* жасыл мәтін*/
  textbackground(BLACK);    /* қара фон*/
  clrscr();                          /* экранды тазалау*/
  printf("\nx,y енгіз: ");
  scanf("%d",&x);
  if (x<0)        y=sqrt(x*x+1)+abs(x);
     else          y=3*x+4;
  printf("Нәтиже=%f\n",y);
  printf("Аяқтау үшін Enter басу керек");
  getch();    /* нәтиже экранын көрсету */
}
```

$$y = \begin{cases} \sqrt{x^2+1}+|x|, & \text{егер } x < 0; \\ 3x+4, & \text{егер } x \geq 0 \end{cases}$$

2. Си тілінің алфавиті

Тілдің алфавиті программаның элементтерін құруда қолдануға болатын символдар жиынынан тұрады. Оған әріптер, цифрлар және арнайы белгілер кіреді.

Тіл ерекшеліктеріне қарай символдар тобын шартты түрде төмендегі топтарға жіктеуге болады, олар

- атау (идентификатор) ретінде қолданылатын символдар (a,b,c,…,z және цифрлар);
-цифрлар (0,1,2,…,9) ;
-айыру белгілері (, . : ; “ _);
-арнайы символдар (@, $, #, &, *…).

а) Әріп ретінде латын алфавитінің бас және кіші әріптері қолданылады, олар:

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

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

және астын сызу таңбасы (_) әріпке саналады;

Бас әріптер мен кіші әріптер бірдей болып саналмайды, мысалы, *X* пен *x* екі айнымалы атауы, дәл сол сияқты *ALFA1, aLfA1* және *alfa1* де әр түрлі атаулар түрлері болып саналады. Атауларда әріптер цифрлармен араласып жазыла береді, бірақ атаудың алғашқы символы міндетті түрде әріп болуы тиіс, мысалы, *VES1, SALMAK2, Baga5, cena7, T7S25, ART25var8*,т.с.с.

Ұлттық әріптер (қазақ, орыс, араб т.с.с.) атау ретінде қолданылмайды, олар тек тырнақшаға (“) алынған тұрақты сөз тіркестері немесе / және */ таңбаларымен қоршалған түсініктеме ретінде ғана кездеседі* /* бұл түсініктеме */.

ә) Ондық цифрлар: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 тәрізді сан таңбалары.

Он алтылық цифрлар ондық цифрлардан және A-дан F-қа (немесе a-дан f-қа) дейінгі латын әріптерінен тұрады.

Арнайы символдарға пунктуациялау және операциялар (амалдар) белгілері жатады.

б) *Арифметикалық амалдардың белгілері:*

\+ - қосу; * - көбейту;

– - алу; / - бөлу; % - қалдық табу;

10 % 3 – нәтижесі 1;

в)Логикалық амалдардың белгілері:

and - және - && (екі шарт қатар орындалады);

or - немесе - || (екі шарттың бірі орындалады);

not - емес - ! (шартқа кері - терістеу амалы);

г) *Айыру белгілеріне* бос орын, ENTER (келесі жолға көшіру) пернесін басу белгісі және үтір, нүктелі үтір таңбалары жатады. Айыру белгілері атауларды, сандарды, түйінді сөздерді бір-бірінен бөліп тұрады.

Түсініктеме // таңбасынан кейін жол соңына немесе /* және */ белгілерімен қоршалып, солардың ішіне жазылады, соңғысы – бір немесе бірнеше жолдардан тұруы мүмкін.

Сонымен, ***айыру белгілері***: _ (бос орын), , (үтір), . (нүкте), : (қос нүкте), ; (нүктелі үтір), ' (апостроф), “ (қостырнақша), (,), [,] , { , } таңбалары.

д) *Қатынас таңбалары немесе салыстыру белгілері:*

= (тең), != немесе <>(тең емес), < (кіші), > (үлкен), <= (үлкен емес таңбасының орнына), >= (кіші емес таңбасының орнына)

Әрбір символдың өзінің реттік нөміріне сәйкес белгіленген коды болады, ол стандарт түрінде бекітілген. Әр елдің стандарттары негізінде американдық кодтар стандарты жатады (American Standart Code for Information Interchange - ASCII), компьютерде жұмыс істеу кезеңінде оларды да білген абзал.

0	32	64 - @	96 - `	128 - А	160 - а	192 - ?	224 -р
1 - ☺	33 - !	65 – A	97 - a	129 - Б	161 - б	193 - ?	225 -с
2 - ?	34 - "	66 – B	98 - Ь	130 - В	162 - в	194 - ?	226 -т
3 - ♠	35 - #	67 – C	99 - c	131 - Г	163 - г	195 - ?	227 - у
4 - ♥	36 - $	68 – D	100 - d	132 - Д	164 - д	196 - ?	228 - ф
5 - ♦	37 - %	69 – E	101 - e	133 - Е	165 - е	197 - ?	229 - х
6 - ↔	38 - &	70 – F	102 - f	134 - Ж	166 - ж	198 - ?	230 - ц
7 - •	39 - '	71 - G	103 - g	135 - З	167 - з	199 - ?	231 - ч
8 - ?	40 - (	72 - H	104 - h	136 - И	168 - и	200 - ?	232 - ш
9 - o	41 -)	73 - I	105 - i	137 - Й	169 - й	201- ?	233 - щ
10 - ?	42 -*	74 - J	106 - j	138 - К	170 - к	202 - ?	234 - ъ
11 -?	43 - +	75 - K	107 - k	139 - Л	171 - л	203- ?	235 - ы
12 - ?	44 - ,	76 - L	108 - 1	140 - М	172 - м	204 - ?	236 - ь
13 - ?	45 - -	77 - M	109 - m	141 - Н	173 - н	205 - ?	237 - э
14 - ?	46 - .	78 - N	110 - n	142 - О	174 - о	206 - ?	238 - ю
15 - ?	47 - /	79 - 0	111 - o	143 - П	175 - п	207 - ?	239 - я
16 - ?	48 - 0	80 - P	112 - p	144 - Р	176 - ?	208 - ?	240 - Ё
17 - ?	49 - 1	81 - Q	113 - q	145 - С	177 - ?	209 - ?	241 - ё
18 - ↕	50 - 2	82 - R	114 - r	146 - Т	178 - ?	210- ?	242 - ?
19 - ?	51 - 3	83 - S	115 - s	147 - У	179 - ?	211 - ?	243 - ?
20 - ¶	52 - 4	84 - T	116 - t	148 - Ф	180 - ?	212 - ?	244 - ?
21 - §	53 - 5	85 - U	117 - u	149 - Х	181 - ?	213 -?	245 - ?
22 - ■	54 - 6	86 - V	118 - v	150 - Ц	182 - ?	214- ?	246 - ?
23 – ?	55 - 7	87 - W	119 - w	151 - Ч	183- ?	215 - ?	247 - ?
24 - ?	56 - 8	88 - X	120 - x	152 - Ш	184 - ?	216 - ?	248 - °
25 - ?	57 - 9	89 - Y	121 - v	153 - Щ	185 - ?	217 - ?	249 - •
26 - ?	58 - :	90 - Z	122 - z	154 - Ъ	186 - ?	218 - ?	250 - ?
27 - ?	59 - ;	91 - [	123 - {	155 - Ы	187 - ?	219 - ?	251 - ?
28 - ?	60 - <	92 - \	124 - |	156 - Ь	188 - ?	220 - ?	252 - n
29 - ?	61 - =	93 -]	125 - }	157 - Э	189 - ?	221 - ?	253 - ²
30 - ?	62 - >	94 - ^	126 - ~	158 - Ю	190 - ?	222 -?	254 - ?
31 - ?	63 - ?	95 - _	127 - ?	159 - Я	191 - ?	223 - ?	255 - □

0 - Alt + 48	A - Alt + 65	a - Alt + 97
1 - Alt + 49	B - Alt + 66	b - Alt + 98
...	...	...
9 - Alt + 57	Z - Alt + 90	z - Alt + 122

С тілінің түйінді сөздері – программада алдын ала анықталған белгілі бір мағынасы бар сөз тіркестері.

С тілінің түйінді сөздері (служебные или ключевые слова) мәліметтер типтері, операторлар мен стандартты функциялар атаулары, жады кластары, модификаторлар (толықтырғыштар), т.с.с., олардың тілдің әр түрлі нұсқаларында аздап айырмашылықтары болуы мүмкін.

Қордағы (резервтегі) сөздер:

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, т.б.

Бұл келтірілген түйінді сөздерді айнымалы аттары немесе тұтынушы қойған бейстандарт атаулар ретінде қолдануға болмайды.

3.Тілдің қарапайым объектілері

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

Тілдің ары қарай бөлінбейтін ең қарапайым бірлігі лексем (token деп те айтады) деп аталады.

Лексемнің 5 типі бар: операциялар (operator), айыру таңбалары (separator), идентификаторлар (identifier), түйінді сөздер (keyword) және константалар (constant).

Біз лексемдерді дұрыс жазуға мүмкіндік беретін объектілерді қарастырып, программаның жалпы құрылымынан мәліметтер берейік.

Константалар және кез келген айнымалылар бүтін саннан, нақты саннан, символдан немесе сөз тіркестерінен құралады.

1. **Мәліметтер типтері. Мәліметтер –** белгілі бір процесс көмегімен тасымалдап, өңдеуге болатын, формальды түрде бейнеленген фактілер мен идеялар. Мәліметтердің бірнеше негізгі типтері қолданылады. Олар:

· **char** – символдық, яғни таңбалық тип,

· **short** – қысқа бүтін сан,

· **int** – бүтін сан типі,

· **long** – екі еселенген бүтін сан,

· **float** – нақты (жылжымалы нүктелі) сан типі,

· **double** – екі еселенген нақты сан типі.

· **unsigned** – таңбасыз бүтін сан,

Алғашқы төрт тип бүтін сандарды сипаттау үшін қолданылады. Төмендегі кестеде әр түрлі типтердің IBM PC-ге арналған ұзындықтары көрсетілген.

Мәлімет типі	Ұзындығы (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	$3.4*10^{-38}$ ... $3.4*10^{38}$
double	64 бит – 8 байт	$1.7*10^{-308}$ ... $1.7*10^{308}$
long double	80 бит – 10 байт	$3.4*10^{-4932}$ ... $3.4*10^{4932}$

Көрсетілген типтердің ең үлкен ұзындықтары әр түрлі компиляторларда бірдей болуы қажет емес. Олар компьютерлердің аппараттық мүмкіндіктеріне қарай тағайындалады. Бірақ оларға мынадай талаптар қойылады:

int > =short >= 16 бит;

Int типін стандарт бекітпеген, ол компьютерге немесе компиляторға байланысты өзгеріп отырады. 16-разрядты процессорде ол 2 байт (**32768**), ал 32-разрядтысында – 4 байт (**2 147 483 647**). ТурбоСи-де **int = short** (**32768**)

long > = int; long типінің ұзындығы >= 32 бит;

(ТурбоС-дегі ең үлкен long типіндегі сан: **2 147 483 647**)

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.

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

`Char` типін 0–255 аралығындағы таңбасыз бүтін сандарды сипаттауға қолдануға болады, компьютер жадында бұларға бір байт орын бөлінген. Мысалы:

```
char  c1;
char  ck='k'; // бірден осылай мән беруге де болады
```

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

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

Программалау практикасында көбінесе жылжымалы нүктелі (нақты немесе 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.

С тілінде объектілердің мәндерін байт арқылы анықтау үшін `sizeof` стандартты операторы қолданылады. Мысалы:

```
printf("Данные типа double занимают %d байт\n", sizeof(double));
```

Сандар. Сандар мен айнымалылар бүтін және нақты болып екіге бөлінеді. *Бүтін сандар*: +4, -100, 15743, 0 , т.с.с.

Дербес ЭЕМ-дер үшін қолданылатын бүтін сандар (int) бұрын -32768 бен +32767 аралығында ғана жазылған еді, қазір компьютер типіне байланысты ол басқаша да бола береді.

Мұнда ондық, сегіздік және он алтылық бүтін сандар да пайдаланылады. Он алтылық сандардың алдына 0x белгісі қойылады. Мысалы, 0xA12 немесе 0x8B2.

Сегіздік сандар алдына 0 қойылады: 0556, 07012, т.с.с.

Нақты сандар (float) 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±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:

$1{,}25 \cdot 10^{6}$	1.25E+6	0,0005	0.5E-3
$-5{,}2 \cdot 10^{-12}$	-5.2E-12	-180000	-1.8E+5

3. Атау – идентификатор (identifier – объектіге қойылған ат) программаны және программадағы тұрақтыларды, айнымалыларды, функцияларды, файлдарды және тағы басқаларды белгілеп жазу үшін қажет.

Идентификатор – міндетті түрде әріптен басталатын сандар мен әріптердің тізбегі.

Оның ұзындығын өте үлкен етудің қажеті жоқ, өйткені атауларды теру және кейіннен сақтау біраз уақыт керек етеді. Бірақ оларды өте қысқартпай, мағынасына сәкес атау беру қалыптасқан. Мысалы: *X, x1, cymka, bec, p23ps6, dt54as, ALFA, baga2, salmak, Omega2,* т.с.с.

Бас әріп пен кіші әріп бірдей болып саналмайды.

4. Тұрақты немесе **константа** деп программаның орындалу барысында мәндері өзгеріссіз қалатын шамаларды айтады.

Тұрақтыға программа басындағы директива арқылы мән берсек те немесе оны программаның сипаттау бөлімінде идентификатор түрінде белгілеп алып мән берсек те болады. Олар сандық, символдық және тіркестік (int, float, char, string) мәндерді қабылдай алады.

Символдық және тіркестік (строковый – string) мәндер үшін орыс, қазақ әріптерін және кез келген символдарды пайдалануға болады. Олар тырнақша ішіндегі таңбалармен жазылады, мысалы: “S=“ , “сумма” , “функцияның мәні”, “y=“ және т.б.

Тұрақтыларды анықтау мысалдары:

а) Директива арқылы

#define *константа мәні*

```
#define  PI  3.14159   // pi=3.14159 болады
#define  EULER   2.718282  // EULER=2.718282
```

Атау арқылы

- const *типі константа = мәні*

```
const  float  m=99999999;
const  F=765;  // типі int болады
const  char s='B'; // символдық тип
const Mening_atjm = 'Бақыт Бөрібаев' ;
const  C='Turbo C';
```

5. Айнымалылар деп программаның орындалу барысында әр түрлі мәндерді қабылдай алатын шамаларды айтады. Айнымалы – компьютер жадының ат қойылған аймағы. Оған мән берілгенде, сол аймаққа мәннің екілік коды жазылады. Айнымалы мәнін қолдану үшін оның атын – идентификаторын және мән орналасқан аймақтың адресін білу керек.

Олар идентификаторлармен белгіленіп, әр уақытта әр түрлі мәнге ие бола алады. Айнымалылардың белгіленулері: alfa, y, x3, summa, baga, a1b8, т.с.с. Айнымалы атауы оның орындайтын міндетіне сәйкес, түсінікті және қарапайым болғаны жөн. Айнымалыларды сипаттау оларды пайдалану алдында кез келген жерде орналасады да, алдында олардың типі көрсетіледі.

Айнымалы қасиеттері:

1. айнымалы белгілі бір мәнге ие болмағанша, анықталмаған болып саналады. Оған мән беру мынадай тәсілдермен орындалады:

- сырттан енгізу арқылы;
- константантаны меншіктеу арқылы;
- бұрын анықталған айнымалының мәнін беру арқылы;

2. кез келген сәтте айнымалының белгілі бір мәні болады немесе ол анықталмаған болып есептеледі;

3. айнымалыға соңғы берілген мән оның алдыңғы мәнін жойып (өшіріп) жібереді. Айнымалыны таңдау (оқу) және оны пайдалану айнымалының мәнін өзгертпейді.

Айнымалының жазылу форматы:

<тип> <идентификатор1>, ...,<идентификаторN> ;

Мысалы:

```
int    a, b=5, d, D;
float  c, alfa=2.15, b4=1.336e2;
char   symbol, cc;
string  coz, coilem;
```

Сипаттау кезінде бірден бастапқы мән меншіктеуге болады, оны айнымалыны инициалдау дейді. С тілінде символдық тіркестерді сипаттау үшін арнайы тип жоқ, олар көбінесе char типтегі элементтерден тұратын массив (жиым) ретінде қарастырылады. Жолдық немесе тіркестік символдар ЭЕМ жадында көршілес ұяшықтарда сақталады да, олардың соңында ‘\0’ символы тұрады. Символдар қатарының ұзындығын анықтау үшін `strlen` сөзі қолданылады.

Арифметикалық немесе логикалық амалдар таңбасымен біріктірілген айнымалылар, атаулар, функциялар, жиымдар (массивтер) тізбегі *өрнек* деп аталады.

Математикадағы формулалар, арифметикалық өрнектер, алгебрадағы көпмүшеліктер программалау тілінде тек осы өрнек ұғымы арқылы беріледі.

Программалау тілінің белгілі бір іс-әрекетті орындай алатын тиянақты мағынасы бар ең қарапайым сөйлемі *оператор* деп аталады.

4. Стандартты функциялар

Си тілінде математикадағы тәрізді стандартты функциялар бар. 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: abs(x), tg(x), sqr(x), sin(x), cos(x), exp(x), ln(x), sqrt(x), arctan(x), frac(x) – санның бөлшегі, int(x) – санның бүтіні, pi (3.14159) т.с.с. Функцияны есептеу барысында аргумент пен функция типтерінің әр уақытта сәйкес келе бермейті-нін есте сақтаған жөн.

Енді программаларда жиі пайдаланылатын өрнектерді мысал ретінде қарастырайық.

$\|x\|$	*fabs(x)*	Аргументтік абсолюттік шамасы	нақты
tgx	*tan(x)*	Аргументтің тангенсы	нақты
arctgx	*atan(x)*	Аргументтің арктангенсы	нақты
cosx	*cos(x)*	Аргументтің косинусы	нақты
sinx	*sin(x)*	Аргументтің синусы	нақты
e^x	*exp(x)*	*e*-нің (2,71828) *x* дәрежесі	нақты
	frac(x)	*x*-санының бөлшек бөлігі	нақты
[x]	*int(x)*	*x*-санының бүтін бөлігі	нақты
lnx	*log(x)*	*x*-санының натурал логарифмі	нақты
π	*pi*	(π-дің мәні pi=3.14159265)	
x^n	*pow(x,n)*	*x*-тің *n* дәрежесі	*x*-тің типіндей
$\sqrt{x}$	*sqrt(x)*	*x*-тің квадрат түбірі	нақты

lgx	*log10(x)*	Аргументтің 10-дық логарифмі *x*-тей	
10^x	*pow10(x)*	Аргументтің тангенсы	нақты
arctgx	*atan(x)*	Аргументтің арктангенсы	нақты
arccosx	*acos(x)*	Аргументтің арккосинусы	нақты
arcsinx	*asin(x)*	Аргументтің арксинусы	нақты
	ceil(x)	үлкен бүтінге қарай дөңгелектеу	нақты
	floor(x)	кіші бүтінге қарай дөңгелектеу	нақты
	frac(x)	*x*-санының бөлшек бөлігі	нақты
[x]	*int(x)*	*x*-санының бүтін бөлігі	нақты
RAND_MAX ең үлкен int саны 32767			бүтін
π	*M_PI*	(π-дің мәні pi=3.14159265)	нақты
	rand()	кездейсоқ бүтін сан (0-32767)	бүтін

1. $y = 5\sin^2 x + \sqrt[3]{3x^5 + 5}$

2. $y = \cos x^2 + \ln x$

3. $y = tgx + \sqrt{5x^2 + 7}$

4. $y = e^{2x} + (x + \cos^2 x)$

5. $y = e^{5x} + (2x + 3)^2$

6. $y = \sin^2 x + \left|x^3 + \cos x\right|$

7. $y = \ln x^2 + \sin \pi\ x$

8. $y = tg3x + \cos^2 \pi\ x$

9. $y = 9x^5 + \sqrt[5]{3x^3 - 7}$

10. $y = (x + 3)\ln 5x^2 + \sin 4x$

11. $y = (x - 5)\sin 2x + \sqrt[3]{x^2 + 6}$

12. $y = e^{3x} + \ln 3x$

13. $y = \sin 5x + \left|x^3 + 7x\right|$

14. $y = tg4x + \sin(3x + 8)$

15. $y = 9x^2 + \ln(5x + 3)$

16. $y = \sqrt[3]{11x^3 + 17} + \cos 4x$

17. $y = \cos 5x + e^{3x}$

18. $y = tg2x + \sqrt[4]{13x^2 - 17}$

19. $y = e^{5x} + \ln 5x$

20. $y = 3\sin^2 x + \sqrt{3x^2 + 6}$

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, аргумент-тердің 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. <Аргументтер тізімі> 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 e орнына E ([-]d.ddddE±dd) шығарылады,

%c – бір символ, яғни таңба (char) 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.

2) **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.

%e және %f спецификациялары енгізу кезінде бірдей болып табылады;

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.

Жол енгізуден бір мысал келтірейік.

```
/* Жол енгізу мысалы */
main()
{
char name [15];
clrscr();
printf("\n,Ввод имени \n");
scanf("%s",name);
printf("Автором программы является
%s\n",name);
}
```

Тест сұрақтары

1. Алгоритм дегеніміз ...

A) бастапқы айнымалы түрде берілген мәліметтерден қажетті нәтижеге қол жеткізу жолында атқарылатын есептеу процесін анықтайтын дәлме-дәл нұсқаулар жиыны;

B) есепті шығару жолын құрастыру процесі;

C) есепті шығару жолының формальды (жасанды) түрде жазылуы;

D) есеп шығаруға арналған символдар мен сол символдардан тұратын конструкцияларды құрастыру және түсіндіру ережелерінің жиыны;

E) белгілі бір процесс көмегімен тасымалдап, өңдеуге болатын, формальды түрде бейнеленген фактілер мен идеялардың дәлме-дәл жиыны.

2. Алгоритмдік тіл …

A) алгоритмдерді жазуға арналған символдар мен сол символдардан тұратын конструкцияларды құрастыру және түсіндіру ережелерінің жиыны;

B) бастапқы айнымалы түрде берілген мәліметтерден қажетті нәтижеге қол жеткізу жолында атқарылатын есептеу процесін анықтайтын дәлме-дәл нұсқаулар жиыны;

C) есепті шығару жолын құрастыру процесі;

D) есепті шығару жолының формальды (жасанды) түрде жазылуы;

E) белгілі бір процесс көмегімен тасымалдап, өңдеуге болатын, формальды түрде бейнеленген фактілер мен идеялардың дәлме-дәл жиыны.

ыңдағандарыңызға

рахмет!