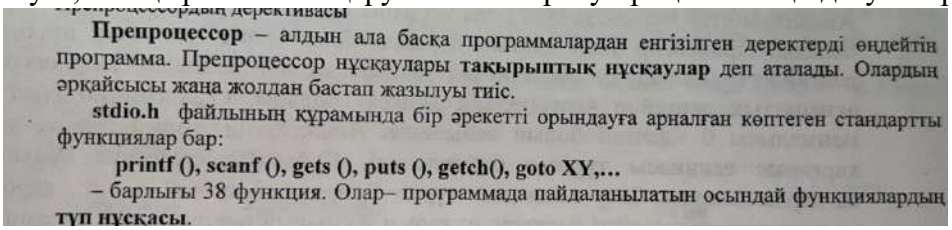
1-деңгей

1. **Препроцессор дегеніміз не және қолдану жолдарын түсіндір**

Бағдарламалаудағы препроцессор - бұл бастапқы кодты құрастыру немесе орындау алдында өңдеу және өзгерту үшін қолданылатын құрал. Ол мәтінді ауыстыру, белгілі бір код фрагменттерін енгізу, шартты компиляцияны анықтау және басқа манипуляциялар сияқты код бойынша белгілі әрекеттерді орындайды.

Препроцессорлар C, C++, Java, Python және т.б. сияқты бағдарламалау тілдерінде жиі пайдаланылады және бағдарламалық жасақтаманы әзірлеуді жеңілдету үшін әртүрлі мүмкіндіктерді ұсынады. Препроцессорды пайдалану бағдарламашыларға кодты икемді етуге, оның әрекетін басқаруға және әзірлеу процесін жеңілдетуге мүмкіндік береді.



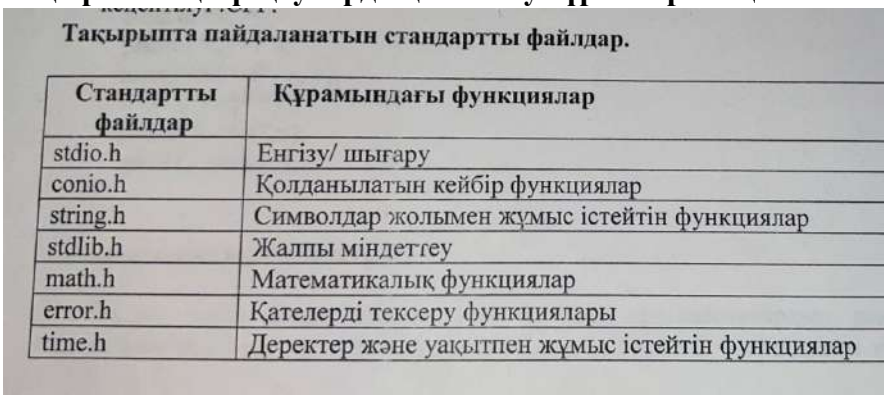
Препроцессор – алдын ала басқа программалардан енгізілген деректерді өңдейтін программа. Препроцессор нұсқаулары **тақырыптық нұсқаулар** деп аталады. Олардың әрқайсысы жаңа жолдан бастап жазылуы тиіс.

stdio.h файлының құрамында бір әрекетті орындауға арналған көптеген стандартты функциялар бар:

printf (), scanf (), gets (), puts (), getch(), goto XY,...

– барлығы 38 функция. Олар– программада пайдаланылатын осындай функциялардың **түп нұсқасы**.

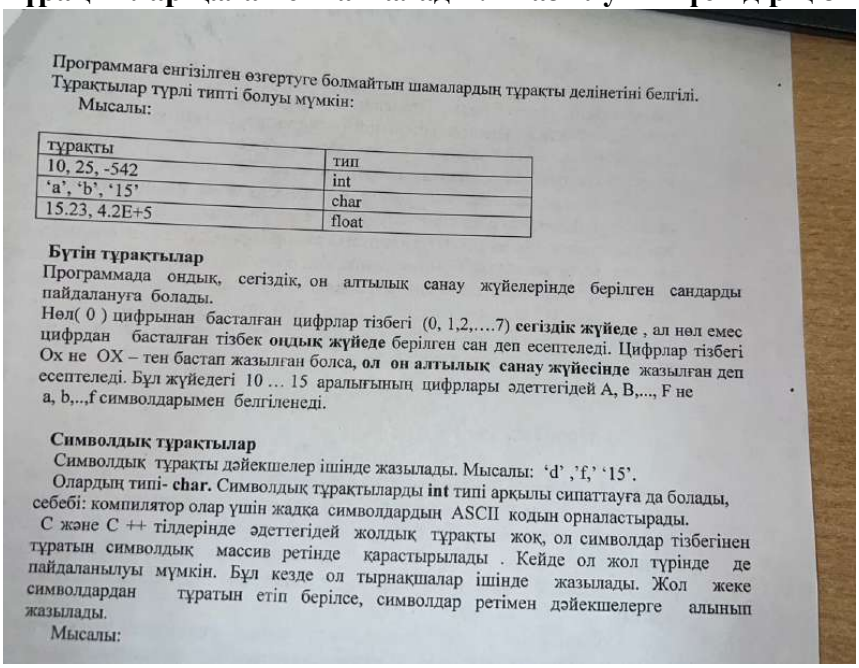
2. **Тақырыптық нұсқаулардың жазылу түрін көрсетіңіз**



Тақырыпта пайдаланатын стандартты файлдар.

Стандартты файлдар	Құрамындағы функциялар
stdio.h	Енгізу/ шығару
conio.h	Қолданылатын кейбір функциялар
string.h	Символдар жолымен жұмыс істейтін функциялар
stdlib.h	Жалпы міндеттеу
math.h	Математикалық функциялар
error.h	Қателерді тексеру функциялары
time.h	Деректер және уақытпен жұмыс істейтін функциялар

3. **Тұрақтылар қалай сипатталады? Жазылуын түсіндіріңіз**



Программаға енгізілген өзгертуге болмайтын шамалардың тұрақты делінетіні белгілі. Тұрақтылар түрлі типті болуы мүмкін:

Мысалы:

тұрақты	тип
10, 25, -542	int
'a', 'b', '15'	char
15.23, 4.2E+5	float

Бүтін тұрақтылар

Программада ондық, сегіздік, он алтылық санау жүйелерінде берілген сандарды пайдалануға болады.

Нөл(0) цифрынан басталған цифрлар тізбегі (0, 1,2,….7) **сегіздік жүйеде** , ал нөл емес цифрдан басталған тізбек **оңдық жүйеде** берілген сан деп есептеледі. Цифрлар тізбегі Ох не ОХ – тен бастап жазылған болса, **ол он алтылық санау жүйесінде** жазылған деп есептеледі. Бұл жүйедегі 10 … 15 аралығының цифрлары әдеттегідей A, B,..., F не a, b,..,f символдарымен белгіленеді.

Символдық тұрақтылар

Символдық тұрақты дәйекшелер ішінде жазылады. Мысалы: 'd' ,'f,' '15'.

Олардың типі- **char.** Символдық тұрақтыларды **int** типі арқылы сипаттауға да болады, себебі: компилятор олар үшін жадқа символдардың ASCII кодын орналастырады.

С және С ++ тілдерінде әдеттегідей жолдық тұрақты жоқ, ол символдар тізбегінен тұратын символдық массив ретінде қарастырылады . Кейде ол жол түрінде де пайдаланылуы мүмкін. Бұл кезде ол тырнақшалар ішінде жазылады. Жол жеке символдардан тұратын етіп берілсе, символдар ретімен дәйекшелерге алынып жазылады.

Мысалы:

4. **Алфавит, идентификатор, кілттік сөздер. Мысалдар келтіріңіз**

Алфавит - программалау тілі программа жазу үшін пайдаланатын символдар жиынтығы.

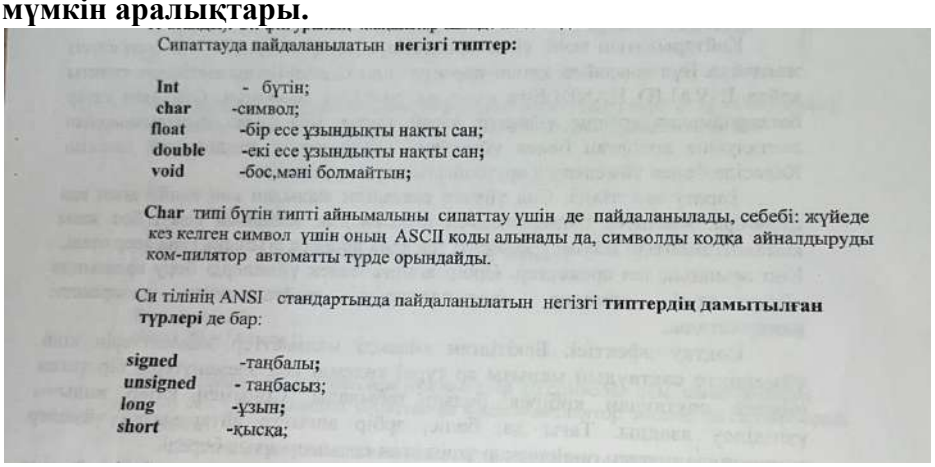
Идентификаторлар - Бұл айнымалыларды, функцияларды, сыныптарды және басқа бағдарлама элементтерін анықтау үшін бағдарламашылар берген атаулар. Олар белгілі бір ережелерді сақтауы керек және тілдік кілт сөздермен бірдей бола алмайды.

Идентификаторлар: main, number, pi.

Ключевое слово: int, double, return.

Алфавит: C++ кодын жазу үшін бүкіл бағдарламада қолданылады.

5. Пайдаланылатын негізгі типтер. Олардың биттік өлшемдері мен пайдалануы мүмкін аралықтары.



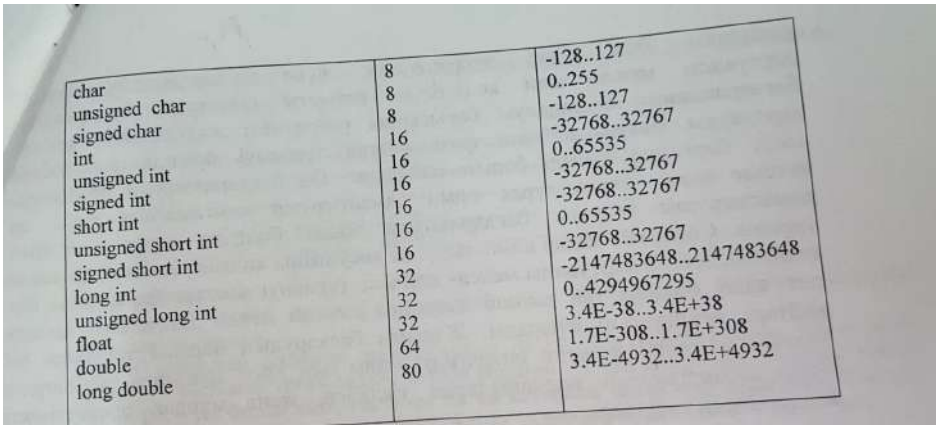
Сипаттауда пайдаланылатын **негізгі типтер:**

Int - бүтін;
char -символ;
float -бір есе ұзындықты нақты сан;
double -екі есе ұзындықты нақты сан;
void -бос,мәні болмайтын;

Char типі бүтін типті айнымалыны сипаттау үшін де пайдаланылады, себебі: жүйеде кез келген символ үшін оның ASCII коды алынады да, символды кодқа айналдыруды ком-пилятор автоматты түрде орындайды.

Си тілінің ANSI стандартында пайдаланылатын негізгі **типтердің дамытылған түрлері** де бар:

signed -таңбалы;
unsigned - таңбасыз;
long -ұзын;
short -қысқа;

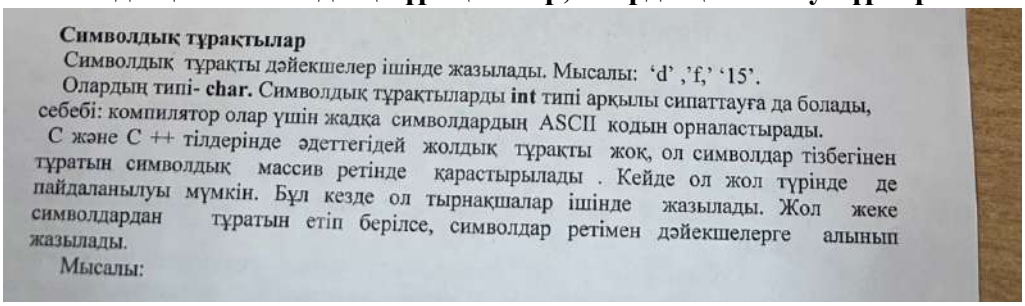


char	8	-128..127
unsigned char	8	0..255
signed char	8	-128..127
int	16	-32768..32767
unsigned int	16	0..65535
signed int	16	-32768..32767
short int	16	-32768..32767
unsigned short int	16	0..65535
signed short int	16	-32768..32767
long int	32	-2147483648..2147483648
unsigned long int	32	0..4294967295
float	32	3.4E-38..3.4E+38
double	64	1.7E-308..1.7E+308
long double	80	3.4E-4932..3.4E+4932

6. Бүтін, нақты және символдық типті айнымалыларды сипаттау тәсілдері.

Тип	Мәндері	операциялары	Ішкі көрсеттілімі
Бүтін	Кейбір диапазондағы бүтін оң және теріс сан. Мысалы, 23,-12, 387.	Бүтін сандармен арифметикалық операциялар: +,- ,/,*, mod, div. Қатынас операциялары (, =, және т.б.)	Тиянақты үтірлі формат
Нақты	Кейбір диапазондағы кез келген (бүтін және бөлшек) сандар. Мысалы, 2.5, -0.01, 45.0, 3.6-109	Арифметкалық операциялар: +,-,*,/. Салыстыру операциялары	Жылжымалы үтірлі формат
Символдық	True (ақиқат) False (жалған)	Логикалық операциялар: ЖӘНЕ (and), НЕМЕСЕ (OR), ЕМЕС (not) қатынас операциялары	1 бит: 1- True 0- False

7. Символдық және жолдық тұрақтылар, олардың жазылу түрлері.



Символдық тұрақтылар

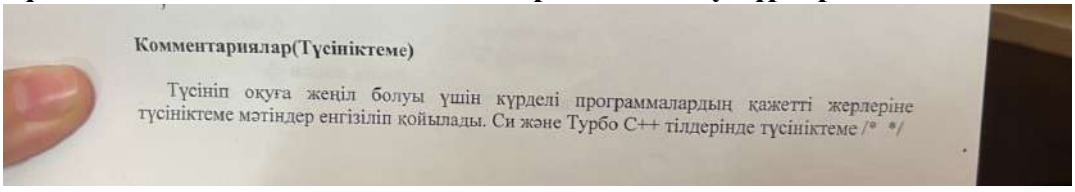
Символдық тұрақты дәйекшелер ішінде жазылады. Мысалы: ‘d’ ,’f,’ ‘15’.

Олардың типі- **char.** Символдық тұрақтыларды **int** типі арқылы сипаттауға да болады, себебі: компилятор олар үшін жадқа символдардың ASCII кодын орналастырады.

C және C ++ тілдерінде әдеттегідей жолдық тұрақты жоқ, ол символдар тізбегінен тұратын символдық массив ретінде қарастырылады . Кейде ол жол түрінде де пайдаланылуы мүмкін. Бұл кезде ол тырнақшалар ішінде жазылады. Жол жеке символдардан тұратын етіп берілсе, символдар ретімен дәйекшелерге алынып жазылады.

Мысалы:

8. Түсініктеме. Оның С және С++ тілдерінде жазылу түрлері.



Комментариялар(Түсініктеме)

Түсініп оқуға жеңіл болуы үшін күрделі программалардың қажетті жерлеріне түсініктеме мәтіндер енгізіліп қойылады. Си және Турбо С++ тілдерінде түсініктеме /* */

символдарының арасына жазылады. Түсініктеме бірнеше қатарлық болуы да мүмкін, ол үшін түсініктеме енгізетін бос орындар болса болғаны.

Мысалдар.

1) /* квадрат теңдеу */
2) /* программаға енгізелген
   /* көпжылдық түсініктеменің
   /*жазылу түрі */

Borland C++ тілінде түсініктеме тек бір қатардан тұрады және ол көлбеу // символдарынан соң енгізіледі, мысалы: // **квадрат теңдеу**

Ал, бірден бірнеше түсініктеме енгізу қажет болса, оларды әр қатарда осы түрде жазып қойса болғаны.

9. Программа дегеніміз не? Қарапайым программа құру тәсілі.

Программа дегеніміз машина тіліне аударылған сіздің есеп-бұйрығыңыз, өйткені әзірше машина кәдімгі адам тілін түсіне алмайды.

Программа клавишті пульт перфокарта, магнитті дискілер арқылы компьютерге енгізеді. Компьютер программада көрсетілген іс-әрекеттерді орындайды. Программа қатесіз жасалу керек, белгілі ережелер сақталмаса, машина ондай програм маны орындамайды.

Программа- арнайы текст арқылы комспьютерге тапсырманың алгоритмін хабарлайды. Алгоритм –арнайы іс-әрекеттердің белгілі кезекте орналасқан тәртібі, алогритм арқылы машина тапсырманы орындайды.

ПҚ- компьютерге арналған программалар жиыны

Ол 3 топқа бөлінеді:

1. жүйелік(базалық) ПҚ
2. қолданбалы ПҚ
3. аспаптық ПҚ

Сонымен, С++ программасы бірнеше функциялардан (**main, f1, f2**…) құралады және олардың біреуі міндетті түрде **main**() болуы қажет.Қарапайым программа функциялардан тұрады.Функция тұлғасы операторлардан тұрады, олар жүйелі жақшалармен шектеледі. Әрбір оператордан кейін **;** таңбасы қойылады.

Енді бір программа мысалын келтірейік:

10. main() функциясы дегеніміз не? Оның құрылымын жазып беріңіз.

main() *бас функция атауы (* main–негізгі). Оны *функция тақырыбы* деп те атайды. Кез келген программа осы функцияны шақырудан (жазудан) басталады.

{– ашу,} –жабу фигуралық жақшалары. Олар *функция денесінің (блоктың) басы мен соңын білдіретін символдар.* Символдар Турбо Паскальдағы программа мен құрама оператор үшін пайдаланатын **begin-end** операторлары сияқты. Әдетте олар жаңа жолдан бастап жазылады және фигуралық {жақшасының алдында бос орын қалдырылмайды.

main() функциясының құрылымы:

```
main ()                    /*функция тақырыбы */
{                          /*функция денесінің басы */
    функция денесі
}                          /* функция денесінің соңы */
```

11. Тақырыпта пайдаланылатын кітапханалық файлдар.

Тақырыпта пайдаланатын стандартты файлдар.

1.2.-кесте

Стандартты файлдар	Құрамындағы функциялар
stdio.h	Енгізу/ шығару
conio.h	Қолданылатын кейбір функциялар
string.h	Символдар жолымен жұмыс істейтін функциялар
stdlib.h	Жалпы міндеттеу
math.h	Математикалық функциялар
error.h	Қателерді тексеру функциялары
time.h	Деректер және уақытпен жұмыс істейтін функциялар

12. printf(), scanf(), clrscr(), getch(), return функцияларының орындайтын іс-әрекеттері. Олардың программада жазылу түрлері.

әркайсысы жаңа жолдан бастап жазылуы тиіс.

stdio.h файлының құрамында бір әрекетті орындауға арналған көптеген стандартты функциялар бар:

printf (), scanf (), gets (), puts (), getch(), goto XY,…

– барлығы 38 функция. Олар– программада пайдаланылатын осындай функциялардың **түп нұсқасы**.

Мұндай файлдардың құрамындағы түп нұсқалық функциялардың тізімін *экранға шығару* үшін **Турбо С++** -те құрылған программаның басында жазылған препроцессор файлы *атауын тышқанның оң түймесімен шерту* жеткілікті.

Бұл **функция** бір әрекетті орындайтын программаның **жеке бірлігі (құрылымдық блогы)** Блок ішінде математикалық функциялар сияқты бірнеше жеке функциялардың болуы да мүмкін. Олардың әркайсысы меншіктеу операторының құрамында жазылады.

Программада бірнеше блоктың болуы да ықтимал. Си, С++ тілдерінде оларды **жеке функциялар** деп атайды. Жергілікті айнымалылар блок басында сипатталады (олар блок ішіндегі сәйкес айнымалылардың– *түп нұсқалары*).

clrscr() – программа іске қосылған кезде *жаңа терезелік бетті ашып, оны тазалау және курсорды оның жоғарғы сол жақ бетіне орналастыру операторы*. Ыңғайлы болуы үшін Си тілінде программада *бұл оператор айнымалылар сипатталған соң бірден енгізіледі.*

Формат % символынан басталады да, оның соңына тағайындалған **форматтық символ** жазылады. Мысалы, басқару жолына енгізілген **%d** – формат спецификаторы (**спецификатор** – форматты түрлендіру командасы.) ол сәйкес аргументке бүтін ондық санды пайдалану керектігін нұсқайтын символдық тұрақтыдан тұрады. Форматтық символдар 1.3–кестеде берілген.

Тырнақшалар ішінде соңғы құрастырылып жазылған **\ n** символдық тұрақтысы– *курсорды келесі жолдың басынан бастап орналастыру командасы.*Мұндай **басқарушы символ жаңа жол символы** деп аталады. Басқарушы символдар 1.4–кестеде енгізілген. Кестеде **бос орын символы** да бар. Оны пайдалану үшін формат бойынша шығару (printf()) операторының *басқару жолында бос орындар тастап кетсе болғаны.*

Программада оған екі аргумент енгізілген: біріншісі–формат спецификаторы, екіншісі–жадқа енгізілетін айнымалы. Оның алдында жазылған **&(амперсанд) символы– адрес (ол адресті алу символы** деп аталады**).** Бұл символ оның соңына жазылған айнымалы адресін тез тауып, scanf() функциясының дұрыс жұмыс істеуін қамтамасыз етеді, яғни айнымалы жазылатын жерге айнымалының атауы емес, **адресі енгізіледі.**

Return 0–ағымдық функциядан(ішкі блоктан) шығып, басқаруды шақырылған программаға (негізгі жүйеге) қайтарып беру операторы. Си, Турбо Си және Borland C++ тілдерінде басқаруды **main()** функциясына қайтарып беру операторы да бар – ол **return**; түрінде жазылады.

Ескерту. Программаны құру кезінде

13. Формат символдары, басқарушы символдар. %d, %s, \n символдарының тағайындамалары мен программада жазылу түрлері.

Формат символдары

1.3.-кесте

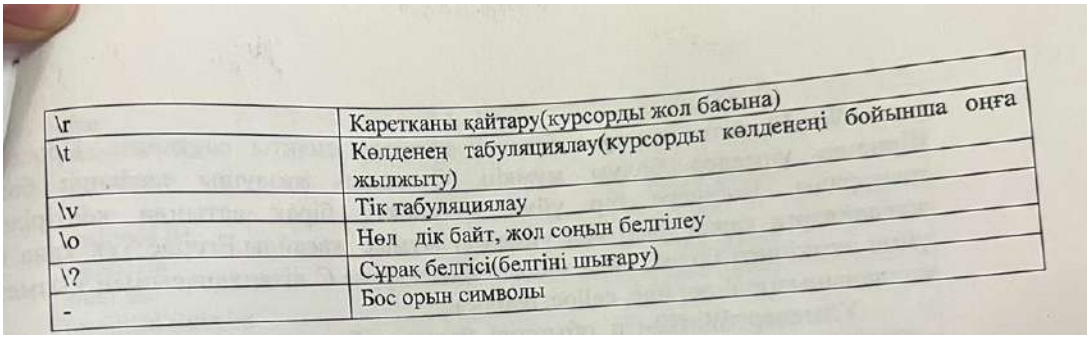
Символдар форматы	Тағайындама	Тип
%d	Бүтін ондық сан (енгізу / шығару)	int
%c	Символ (енгізу / шығару)	char
%s	Символдар жолы (енгізу / шығару)	char[]
%f	Ондық нақты сан (шығару)	float/ double
%e	Ондық нақты сан (енгізу)	float
%p	Көрсеткіш (енгізу / шығару)	түрлі

Ескерту. енгізу–**scanf** функциясында, шығару–**printf** функциясында пайдаланатынын білдіру жазулары.

Басқарушы символдар

1.4.-кесте

Басқарушы символдар	Тағайындама
\ n	Жаңа жол (курсорды жаңа жол басына жылжыту)
\f	Жаңа бет, бетті аудару



\r	Каретканы қайтару(курсорды жол басына)
\t	Көлденең табуляциялау(курсорды көлденеңі бойынша оңға жылжыту)
\v	Тік табуляциялау
\o	Нөл дік байт, жол соңын белгілеу
\?	Сұрақ белгісі(белгіні шығару)
-	Бос орын символы

14. Арифметикалық операцияларға мысалдар келтір

Арифметикалық операциялар	Орындалуы
+	Қосу
–	Азайту
*	Көбейту
/	Бөлу, бүтін бөлігін алу
%	Бөлгендегі қалдығын алу

Мысалы: егер b=5, c=2, бүтін типті айнымалылар болса, онда

a=b%c; d=b/c;

операциялардың орындалуынан кейін a айнымалысы 1 мәнін қабылдайды, d айнымалысы 2 мәнін қабылдайды.

Арифметикалық операциясына сәйкес меншіктеу операторын басқаша пайдалануға болады.

Мысалы, x=x+n; жазуын x+=n; деп жазуға болады. Сол секілді

x=x-n; x-=n;

x=x/n; x/=n;

x=x*n; x*=n;

x=x%n; x%=n;

Си тілінде операнд мәнін бірге арттыру (++) және бірге кеміту (--) операциялары пайдаланады.

x++; ++x; операторлардың нәтижесі бір, бірақ пайдалану барысында айырмашылықтары бар. Мысалы:

```
main()
{
int x,y;
x=5; y=60;
x++;  ++y;
printf("x=%d  y=%d\n",x,y);
printf("x=%d  y=%d\n",x++,++y);
}
```

Нәтижесі:

x=5 y=61

x=5 y=62

x++ операторын пайдаланғанда x айнымалының мәні алдымен өрнекте пайдаланады, содан соң бірге артады. ++y пайдаланғанда алдымен y айнымалысы бірге артады, содан соң өрнекте пайдаланады.

15. Қатыс және логикалық операциялар түрлері мен жазылуын көрсетіңіз.

Қатыс операциялары (салыстырулар)

<	Кіші
>	үлкен
<=	Кіші немесе тең
>=	Үлкен немесе тең
= =	Тең
!=	Тең емес

16. Меншіктеу операциясы, түрлері және жазылуын көрсет

Меншіктеу операциялары

=	*Жай меншіктеу:* өрнек мәнін меншіктеу –оң жақтағы операндаға сол жақтағы операнданы меншітеу.Мысал: P= 10 3-2*x;
*=	*Көбейтуден кейінгі меншіктеу* сол жақтағы операнданы оң жақтағы операндаға көбейткен мәнді сол жақтағы операндаға меншіктеу. P *= 2 эквивалентті P = P * 2;
/=	*Бөлуден кейінгі меншіктеу:* сол жақтағы операнданы оң жақтағы операндаға бөлген мәнді сол жақтағы операндаға меншіктеу. P /= 2.2 - d эквивалентті P=P/(2.2-d);
%=	*Модуль бойынша бөлуді меншіктеу:* сол жақтағы операнданы оң жақтағы операндаға бүтін санды бөлген кездегі қалдық мәнді сол жақтағы операндаға меншіктеу. N %= 3 эквивалентті N = N % 3;
+=	*Қосудан кейінгі меншіктеу:* сол жақтағы операнданы оң жақтағы операндаға қосқан мәнді сол жақтағы операндаға A +− B эквивалентті A − A +B;
-=	*Азайтудан кейінгі меншіктеу:* сол жақтағы операндадан оң жақтағы операнданы азайтқан мәнді сол жақтағы операндаға меншіктеу. X -= 4.3 - Z эквивалентті X = X - (4.3 - Z);

17. Инкремент (++) және декремент (--) операторларыны қызметі?

Инкремент – айнымалыны көбейтетін көптеген бағдарламалау тілдеріндегі операция. Көбінесе өсу айнымалының 1 бірлікке ұлғаюын білдіреді.

Кері операция азайту деп аталады. Декремент – айнымалының азаюы. Көбінесе бір.

C++ тілінде арттыру және азайту кез келген сандық айнымалыға қолданылатын бірінші кезекте біртұтас операторлар болып табылады.
Өсу: екі қосу белгісімен «++» белгіленеді және айнымалының мәнін 1-ге арттырады.
Декремент: «--» екі минус белгісімен белгіленеді және айнымалының мәнін 1-ге азайтады.

18. For циклінің жазылу түрі? Мысал келтір

C++ тілінде for циклі өте кең іске асуы және қолданбалы болуы мүмкін. For циклі параметрі бар цикл деп те аталады.
for циклінің операторының жалпы формасы:

```
for (инициализация; өрнек; арттыру)
{
   // операторлар тізбегі
   // ...
}
```

инициализация – циклдік айнымалының бастапқы мәнін орнататын тағайындау операциясы. Бұл айнымалы цикл жұмысын басқаратын есептегіш болып табылады. for циклін басқаратын айнымалылар саны екі немесе одан да көп болуы мүмкін;
өрнек – цикл айнымалысының мәнін тексеретін шартты өрнек. Бұл кезеңде циклдің одан әрі орындалуы анықталады;
ұлғайту – цикл айнымалысының мәні әрбір итерациядан кейін қалай өзгеретінін анықтайды.
For циклі өрнек шын мәніне бағаланғанша орындалады. Өрнектің мәні жалған болған кезде, цикл орындауды тоқтатады және for циклінен кейінгі оператор орындалады.

19. While, do-while циклдерінің жазылу түрі?

Цикл қанша итерация жасау керектігін білмеген кезде, бізге while немесе do...while циклі қажет. C++ тіліндегі while циклінің синтаксисі келесідей.

```
while (шарт) {
   Цикл денесі;
}
```

Бұл цикл жақшада көрсетілген шарт ақиқат болғанша орындалады. Сол мәселені while циклінің көмегімен шешейік. Бұл жерде біз циклдің қанша итерацияны орындау керектігін нақты білсек те, бұл мән белгісіз болатын жағдайлар жиі кездеседі.

do while циклі while цикліне өте ұқсас. Жалғыз айырмашылығы, do while циклін орындау кезінде циклдің бір өтуі шартқа қарамастан орындалады. do while циклі арқылы 1-ден 1000-ға дейінгі сандардың қосындысын табу есебін шешу.

```
#include <iostream>
using namespace std;
int main ()
{
   setlocale(0, "");
   int i = 0; // инициализируем счетчик цикла.
   int sum = 0; // инициализируем счетчик суммы.
   do {// выполняем цикл.
      i++;
      sum += i;
   } while (i < 1000); // пока выполняется условие.
   cout << "Сумма чисел от 1 до 1000 = " << sum << endl;
   return 0;
}
```

20. Енгізу функциясының берілу түрі және қолданылу аймағы?+

21. Шығару функциясының берілу түрі және қолданылу аймағы?+

Си++ енгізу-шығару ағымы. Си++ тілінде программалау кезінде Си тілінің stdio.h тақырыптық файлымен қосылатын стандартты енгізу-шығару жабдықтарын пайдалануға болады. Бірақ Си++ өзінің ерекше енгізу-шығару жабдықтары бар. Бұл iostream.h файлының көмегімен қосылатын кластар кітапханасы. Бұл кітапханада объектілер ретінде стандартты символдық ағымдар анықталған:
cin — пернетақтадан стандартты енгізу ағымы;
cout — экранға стандартты шығару ағымы.
Мысалы, x айнымалыға мәнді енгізу, мына оператормен іске асырылады.

```
cin>>x;
```

cout ағымында тырнақшаға алынған мәтіндер, өрнектер мәні шығарылады. Ағымға орналасудың операция белгісі <<. Ағымды шығарудың мысалдары:

```
cout<<a+b;
cout<<"\nНәтиже="<<Y;
cout<<"x="<<x<<" y="<<y<<" z="<<z<<endl;
```

егізу-шығару ағымын пайдаланып, C++ тілінде ушбұрыштың ауданын табудың программасын құру.

22. CONTINUE операторының орындалу түрі

continue операторы – циклдік ағынды интерациясы аяқталу үшін және сол циклден келесі итерациясына көшу үшін қолданылады. Бірақ ол циклдан шығу тәсілі болып табылмайды.

1 ден 100 дейінгі аралықта жұп сандардың бәрін шығарудың программа фрагментін қарастырайық:

```
for(i=l;i<=100;i++)
{if(i%2) continue; cout<<"\t"<<i;}
```

Жұп емес айнымалылар үшін, i айнымалысын 2-ге бөлгендегі қалдығы 1-ге тең болады да continue операторы орындалады.

23. Жады кластары қанша түрге бөлінеді

Жады класы – жадыдағы объектілердің орналасу тәртібін анықтайды.

Программадағы барлық айнымалылар тек тип бойынша емес, сонымен қатар жады класы бойынша да жіктеледі. Си программалау тілінде төрт жады класы бар:

1. Автоматты (automatic);
2. Регистрлі (register);
3. Статистикалық (static);
4. Сыртқы (external);

24. Автоматты айнымалы (auto) дегеніміз не?

C++ бағдарламалау тілінде «auto» кілт сөзі компилятор арқылы айнымалының деректер түрін автоматты түрде шығару үшін қолданылады. Бұл айнымалы мәндерді жариялауды жеңілдетеді, әсіресе итераторлар немесе ұзын типтер сияқты күрделі деректер түрлерінің контекстінде.

Мысалы:

```
#include <iostream>

int main() {
    auto x = 42;  // x айнымалысының деректер типі int ретінде автоматты түрде шығарылады

    std::cout << "X айнымалысының мәні: " << x << std::endl;

    return 0;
}
```

Бұл мысалда auto компиляторға тағайындалған мән негізінде x айнымалысының түрін дербес анықтауға мүмкіндік береді. Бұл кодыңызды икемді етеді және әсіресе күрделі деректер құрылымдарымен немесе үлгілермен жұмыс істегенде, деректер түрлерін нақты көрсету қажеттілігін азайтады.

25. Регистрлік айнымалы (register) дегеніміз не?

C++ тіліндегі register — компиляторға айнымалы мәнді сақтау үшін процессор регистрін пайдалануды айтатын кілт сөз. Дегенмен, қазіргі заманғы компиляторлар әдетте ресурстарды оңтайландыруда тиімді және «register» пайдалануды елемеуге болады.

```
#include <iostream>
int main() {
    register int x = 10;

    std::cout << "Значение переменной x: " << x << std::endl;

    return 0;
}
```

26. Сыртқы айнымалылар (extern) дегеніміз не?

C++ бағдарламалау тілінде сыртқы айнымалыларды жариялау үшін extern түйінді сөзі қолданылады. Сыртқы айнымалылар әдетте бір бағдарлама файлында жарияланады және басқа файлдарда қолданылады. Олар ағымдағы файлдан тыс айнымалы мәндерге қол жетімді ету жолын қамтамасыз етеді.

'extern' пайдалану мысалы:

```
// «example.cpp» файлында сыртқы айнымалыны жариялаймыз
extern int globalVariable;

int main() {
    // Сыртқы айнымалыны пайдаланыңыз
```

```
    globalVariable = 42;

    return 0;
}

// Басқа "otherFile.cpp" файлында біз сыртқы айнымалыны қолданамыз
extern int globalVariable;

void someFunction() {
    // Сыртқы айнымалыны пайдаланыңыз
    int x = globalVariable + 10;
}
```

Мұнда екі файлдағы 'extern int globalVariable;' компиляторға 'globalVariable' айнымалысы басқа жерде (мүмкін басқа файлда) жарияланғанын және компилятор бағдарламаны құру кезінде оның басқа жерде анықталуын күтуі керек екенін айтады.

extern көмегімен айнымалы мәнді жариялау айнымалы мәндерді бірнеше файлдарда ортақ пайдалануға мүмкіндік береді, бұл үлкен бағдарламалық жобаларды әзірлеу кезінде пайдалы.

27. Статикалық айнымалы (static) дегеніміз не?

Бағдарламалаудағы статикалық айнымалы – функцияға шақырулар арасында немесе программаның барлық орындалу ұзақтығы ішінде белгілі бір код блогында өз мәнін сақтайтын айнымалы. Ол тек бір рет инициализацияланады және оның мәні функция шақырулары арасында сақталады.

C++ бағдарламалау тілінде static кілт сөзі функциялар немесе код блоктары ішіндегі статикалық айнымалыларды анықтау үшін қолданылады. Мысалы:

```
#include <iostream>

void exampleFunction() {
    // Статикалық айнымалы x, тек функцияға бірінші шақыруда инициализацияланған
    static int x = 0;

    // x айнымалысының ағымдағы мәнін шығарыңыз
    std::cout << "x: " << x << std::endl;

    // x айнымалысының мәнін ұлғайту
    x++;
}

int main() {
    // Функцияны бірнеше рет шақырыңыз
    exampleFunction();
    exampleFunction();
    exampleFunction();

    return 0;
}
```

Мұнда x айнымалысы статикалық және оның мәні exampleFunction() шақырулары арасында сақталады.

28. if операторы жазылу түрі, мысал келтір

Си программалау тілінде тармақталу үрдісін программалау үшін әртүрлі жабдықтар бар. Оларға шартты операция ?:, шартты оператор if және switch таңдау операторы жатады.

Тармақталу операторының жазылу үлгісі:

1) if (өрнек) оператор1; else оператор2;

2) if (өрнек) оператор1;

1) мұндағы өрнек немесе қандайда бір шарт орындалғанда оператор1 денесі орындалады, кері жағдайда оператор2 орындалады.

Басқаша if - else операторын шартты операциямен былай жазамыз: Өрнек1?: өрнек2: өрнек3;

a,b берілген. Табу керек: max - ?

```
{
  int a,b,max;
  scanf (“%d%d’’,&a,&b);
  if (a>b) max=a;       else max=b;
  printf(“ең үлкені= %d \n”,max);
}
```

29. switch операторы жазылу түрі, мысал келтір

Си тілінде көп жақты таңдау жасайтын орнатылған оператор бар. Оның негізгі формасы мына түрде болады:

switch (өрнек)

```
{ case constant1:
операторлардың тізбегі
                    break;
   case constant2:
операторлардың тізбегі
                    break;
                    ...
   case constantN:
операторлардың тізбегі
                    break;
                default
операторлардың тізбегі
          }
```

Алдымен switch негізгі сөзінен кейін, жақшаның ішіндегі тұрған өрнектер есептелінеді. Содан кейін есептелінетін өрнектің мәніне сәйкес метка табылмағанша, белгілер тізімі (case constant және т.б.) қарастырылады. қос нүктеден кейінгі операторлардың тізбегі жалғастырылып орындалады. Егер де өрнектің мәні switch операторының ешбір белгісіне сәйкес келмесе, онда default негізгі сөзінен кейінгі операторлар тізбегі орындалады.

```
#include<stdio.h>
main()
{   char ch;
    printf(«Қазақ алфавитінің бас әріптерін енгіз:»);
    ch = getchar ();
    if (ch>=’A’&&’ß’)
    switch (ch)
    {        case ‘A’:
    printf(«Ахметов\n »);
case ‘Б’:
    printf(«Бокеев\n »);
case ‘В’:
    printf(«Валашов\n »);
case ‘Г’:
    printf(«Газизов\n »);
default:
  printf («Дастанов, Зейінқұлов және басқалар\n »);
}
   else printf («Қазақ алфавитінің бас әріптерін енгізуіңіз керек еді\n»);
```

30. Шартты операторлардың жазылу түрі

C++ программалау тілінде шартты операторлар if, else if (қосымша) және else түйінді сөздерінің көмегімен жазылады. Мұнда негізгі шартты мәлімдеменің мысалы келтірілген:

```
cpp
егер (шарт) {
   // шарт шын болса орындалатын код
} басқа {
   // шарт жалған болса орындалатын код
}
```

'else if' арқылы мысал:

```
cpp
егер (1-шарт) {
   // 1-шарт шын болса орындалатын код
} else if (2-шарт) {
   // егер 1-шарт жалған болса және 2-шарт ақиқат болса, орындалатын код
} басқа {
   // егер екі шарт та жалған болса, орындалатын код
}
```

Мұнда 'шарт', 'шарт1' және 'шарт2' нәтижесі логикалық мән (шын немесе жалған) ретінде түсіндірілетін өрнектер болып табылады. Шарттың нәтижесіне байланысты кодтың сәйкес блогы орындалады.

31. Циклдық есептеу процесі дегеніміз не?

Циклдік есептеу процесі – белгілі бір аяқтау шартына жеткенше бір немесе бірнеше қадамдарды қайталайтын циклде орындалатын процесс. Бұл процесс әдетте белгілі бір әрекеттерді қайталап орындауды талап ететін мәселелерді шешу үшін қолданылады.

Мысал ретінде бағдарламалаудағы циклды келтіруге болады, мысалы, «for» немесе «while», онда цикл шарты ақиқат болған кезде код блогы қайта-қайта орындалады. Бұл массив арқылы өту, мәндерді қосу немесе басқа қайталанатын әрекеттер сияқты қайталанатын тапсырмаларды тиімді өңдеуге мүмкіндік береді.

Циклдік есептеу процестері тапсырмаларды автоматтандыру және қайталау үшін бағдарламалауда, математикада және басқа салаларда кеңінен қолданылады.

32. for циклының мүмкіндіктерін ата.

СИ тілінде санауыш бойынша циклді ұйымдастыру үшін, арнайы оператор пайдаланады:

for (1-өрнек; 2-өрнек; 3-өрнек;)
оператор;

мұндағы “1-өрнек“ - цикл параметрінің бастапқы мәнін орнатады, “2-өрнек“ - операцияның орындалуының шартының жалғасын анықтайды, “3-өрнек“ - параметрдің модификациясының ережесін береді.

Цикл операторының мысалы ретінде бүтін оң санның N! факториалын есептеуді қарастырайық

```
F=l;
for(i=l;i<=N;i++) F=F*i;
```

Ескерте кететін жай, әрбір өрнек үтір арқылы бөлінген, бірнешеуден құралуы мүмкін. Мұндай өрнектер топтық деп аталынады.

Мысалы:

```
#include<stdio.h>
main()
{   char *text;
    int vverh, vnis;
    text = «қадам »;
    ind =1;
    for (vverh =1, vnis=5; vverh<=5; vverh++, vnis --)
 printf(« %s: %2d\t %2d\n», text, vverh, vnis);
}
```

33. while циклының мүмкіндіктерін ата.

Циклді ұйымдастыру операторлары. Қайталау есептеу процестері - while..., for... немесе do... while... операторлары бойынша орындалады.

while (өрнек)
оператор;

Біршама әмбебап оператор болып табылады. әзір өрнек нольге тең болмаса, оператордың (жай немесе құрама) әрекетін қайталап орындайды.

Цикл операторының мысалы ретінде бүтін оң санның N! факториалын есептеуді қарастырайық\

```
// Программа вычисле-
ния факториала
  #include <iostream.h>
  void main()
  { long int F;
    int i,N;
    cout<<"N="; cin>>N;
    F=i=1;
    while(i<=N) F=F*i++;
   cout<<"\n"<<N<<"!="<<F;
  }
```

34. Do while циклының мүмкіндіктерін ата?

Циклді do оператор while (өрнек);

әзір өрнекте берілген шарт орындалғанша, “операторомен“ сипатталған әрекет қайталанады.

Цикл операторының мысалы ретінде бүтін оң санның N! факториалын есептеуді қарастырайық

Мысалы:

```
#include<stdio.h>
#include<conio.h>
main()
{   float a, b, res;
    do {
        puts («\n Екі санды енгізіңіз:»);
    scanf («%f%f »,&a,&b);
```

```
    if (b= =0)
    puts («Бөлуге болмайды \n »);
    else
            {res = a/b;
printf(«Нәтижесі: %f\n»,res);
};
    puts («q пернесін басу жұмысты аяқтайды»);}
while (getch() ! =’q’);}
```

while және do...while циклдерінің айырмашылығы, do...while циклін пайдаланған жағдайда, оның құрамына кіретін операторлар міндетті түрде бір рет орындалады.

35. Көрсеткіштер дегеніміз не және оны қолдану жолдарын айтып бер

Функцияға күрделі құрылымды құрылым берілетін болса, онда оның толығымен көшірмесін берудің орынына, оған көрсеткішті қолданған тиімді. Құрылымға көрсеткіштер қарапайым айнымалыларға көрсеткіштер секілді.

Сипаттау мысалдары:

Struct strtype *p;

Мұндағы p айнымалысы strtype типті құрылымға көрсеткіш деп аталады. Ал, *p – құрылымның өзі, оның элементтері (*p).dom, (*p).god түрінде қолданылады.

Көрсеткішті пайдалану жолдары:

Struct strtype sotrudn, *p;

P=&sotrudn;

Printf(“sotrudn info: %s \t %i \t %6.2f \n”, (*p).fam, (*p).god, (*p).zarp);

Мұнда (*p).fam бейнесінде жақша міндетті түрде болуы тиіс, себебі нүкте (.) операторының приоритеті * операторының приоритетінен жоғары. Құрылымдарға көрсеткіштер жиі қолданылады. Сондықтан оның элементтерін қолдану барысында ыңғайлы болатын қысқаша жазу формасы қарастырылған.

- Дерек түрінің көрсеткіштері:* Айнымалының деректер түрін көрсететін int, double, char сияқты кілт сөздерді қосыңыз.
- Қолжетімділік көрсеткіштері:* public, private, protected кілт сөздері сынып мүшелеріне қолжетімділік деңгейін анықтау үшін қолданылады.
- Күй көрсеткіштері:* Айнымалы мәннің өзгермейтінін көрсететін const сияқты.
- Сілтегіш индикаторлары:* Көрсеткіштер мен айнымалы мекенжайларды өңдеу үшін пайдаланылатын * және & сияқты арнайы таңбалар.

36. Массивтерді сипаттау түрі, мысал келтір+

37. Массивтерді инициалдау дегеніміз не?+

Массив дегеніміз — бір атпен берілген айнымалылар тізбегі. Массивтің аты және өлшемі болады. Си тілінде массивті былай сипаттаймыз:

Айнымалылыр типі массив_аты [элементтер саны]

Мысалы: char irr [10], float git [5]

Си программалау тіліндегі қабылдайтын мән 0-ден бастап n-1 аралығына дейінгі мәндерді қабылдайды.

Массив элементтерін енгізуді ұйымдастыру.

Мыс1: A[10] массив элементін енгіз.

Перне тақтаның көмегімен енгізу.

```
#include <stdio.h>
#include <stlib.h>
#include <conio.h>
void  main()
{ const int N=10;
  int A[N];I;
  for (i=1;i<N;i++)
  {
   printf  ( “\nA[%d]→”,i);
   scanf (“%d”,&A[i]);
}
```

38. Екі өлшемді массив дегеніміз не? +

39. Екі өлшемді массивтерді сипаттау түрі?+

Екі өлшемді массив - матрица – векторлар массиві түрінде бейнеленеді және тік жақшадағы екі мәнмен беріледі:

Элементтер типі массив аты [өлшем1,өлшем2....]

Мыс: int Maz [10][20], float [M][N]

```
        const  int M=10;
```

```
const int N=20;
```
Екі өлшемді және оданда көп өлшемді массивтермен орындағанда цикл ішіндегі циклді пайдаланамыз.

Массив элементтерін клавиатурадан енгізгенде мына әрекеттер орындалады:

```
void  main()
{ const int M=5;
 const int  N=5;
 int mas [M][N];i,j;
 for (i=0;i<M;i++)
 { for (j=0;j<N;j++);
  printf  ( “\nmas [%d][%d]=”,I,j);
  scanf (“%d”,&mas[i][j]);
 }
```

40. Графикалық функцияларға мысал келтіріңіз

- initgraph(): графикалық режимді инициализациялау.
- closegraph(): графиканы жабады.
- line(): екі нүктенің арасына сызық сызыңыз.
- circle(): радиусы мен центрі берілген шеңберді салады.
- rectangle(): тіктөртбұрыш салады.
- getch(): перне басылғанша күтіңіз.
- putpixel(): экрандағы пикселдің түсін орнатады.
- setcolor(): сурет салу үшін түсті орнату.

Мысалы

```
#include <graphics.h>

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");

    // Координаты вершин квадрата
    int x1 = 100, y1 = 100, x2 = 200, y2 = 200;

    rectangle(x1, y1, x2, y2);

    getch();
    closegraph();
    return 0;
}
```