«Тұран-Астана» университеті

Таукенова Л.Ж., Абдибекова Л.М., Жумабаев Е.Н.

«C++ бағдараламалау негіздері»

Оқу құралы

Нұр-Сұлтан, 2022

УДК 004.42

Пікір беруші:
Жукабаева Т.К. PhD; қауымдастырылған Профессор Л.Н.Гумилев атындығы Еуразиялық Ұлттық Университеті.
Толегенова А.С. т.ғ.к. С.Сейфуллин атындағы Қазақ Агротехникалық Университеті.
Нуспеков Е.Л. т.ғ.к. профессор “Тұран-Астана” университеті.

Таукенова Л.Ж., Абдибекова Л.М., Жумабаев Е.Н.
С++ бағдараламалау негіздері : Оқу құралы/ Таукенова Л.Ж., Абдибекова Л.М., Жумабаев Е.Н.
Нұр-Сұлтан: «Тұран-Астана» университеті, 2022-104бет

Оқу құралы барлық мамандықтар бойынша бакалаврларды оқыту бағдарламасының оқу жоспарында қарастырылған жұмыс бағдарламасына сәйкес әзірленген.

Оқу құралының құрылымы оқу басылымдарына қойылатын талаптарға сәйкес келетін нақты, логикалық, дәйекті және құрылымдалған мазмұндамамен ерекшеленеді.

Оқулықта бағдарламалау тілінің басты операторларына тоқталып, әрқайсысына мысалдар келтіре отырып, оператордың қолданылуына түсінік берілген. Нақты шығарылған есептердің беретін нәтижесі әдістемелік оқу құралды қолданушыға С++ бағдарламасында оператордың нақты қолданылуын түсінуге мүмкіндік береді.

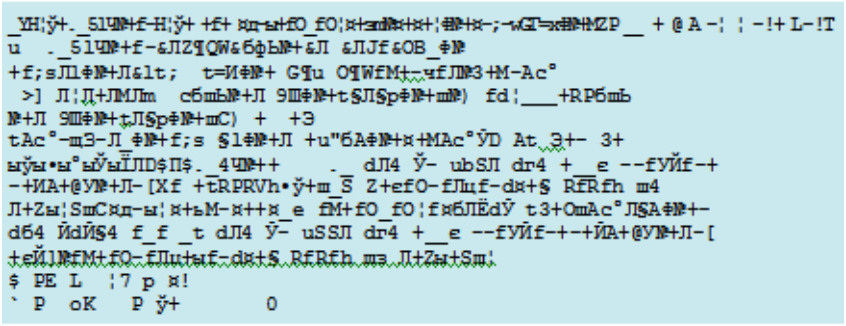
Оқу құралы ақпараттық технологиялар саласында мамандарды дайындауда өзектілікке ие, себебі бітірушінің құзыреттілік моделін қалыптастырудағы жаңашылдық тәсілдері қуантады.

УДК 004.42

Мазмұны

1. Бірінші жоба.

Сіз бірінші бөлімнен С бағдарламалау тілі екенін білесіз. Бізге компьютерден не қалайтынымызды түсінікті түрде түсіндіруге мүмкіндік беретін тіл. Шынын айтқанда, компьютер бір ғана тілді - машиналық кодтар тілін түсінеді. Мысалы, «Hello, world!» тіркесін көрсететін бағдарлама компьютердің «ана тілінде» елесідей көрінеді:



Сурет 1. Бағдарламалық тілдің жазылуы.

Сіз мұндай кодты жазу мүмкін емес деп айтасыз және сіз өте дұрыс боласыз !! Бірақ, біз жазбаймыз. Ол үшін бағдарламаларды жазуды жеңілдету үшін бағдарламалау тілі қажет. Бағдарламалау тілдері екі негізгі топқа бөлінеді, ИНТЕРПРЕТИРУЕМЫЕ және КОМПИЛИРУЕМЫЕ. Бұл бөлім мамандандырылған бағдарлама командаларды программалау тілінен машина тіліне аударады - КОМПИЛЯТОР немесе ИНТЕРПРЕТАТОР. Олардың арасындағы айырмашылық неде екенін білейік. Бізде командалар жиынтығы бар файл бар деп елестетейік.

Бірінші жағдай. Командалар аударылған тілде жазылған.

Бағдарлама іске қосылған сайын интерпретатор код жолын жол бойынша тексереді. Егер синтаксистік қателер болмаса, командалар машина кодына (процессорға арналған нұсқаулықтар жиынтығы) түрлендіріледі. Бағдарлама орындала бастайды. Егер қате болса, аудармашы тоқтайды және сізге оны түзетіп, бағдарламаны қайта іске қосу ұсынылады. Бірақ, егер қателер болмаса және бағдарламаны сіз аяқтасаңыз да, аудармашы іске қосылған сайын іске қосылады және кодты қайта тексереді. Осылайша, кодтың машиналық нұсқасы еш жерде сақталмайды деп қорытынды жасауға болады. Бұл әдістің кемшіліктері - бағдарламаның іске қосылу жылдамдығы баяулайды, бірақ қарап шығуды өшіру мүмкін емес.

Екінші жағдай. Командалар құрастырылған тілде жазылған.

Компилятор интерпретатор сияқты әрекет етеді, яғни кодтық жолды жолмен тексереді. Бірақ егер ол қатеге тап болса, ол тоқтамайды, бірақ кодты соңына дейін зерттейді, барлық кейінгі қателерді анықтайды және олар туралы хабарламаларды көрсетеді. Сонымен қатар, компилятор .OBJ

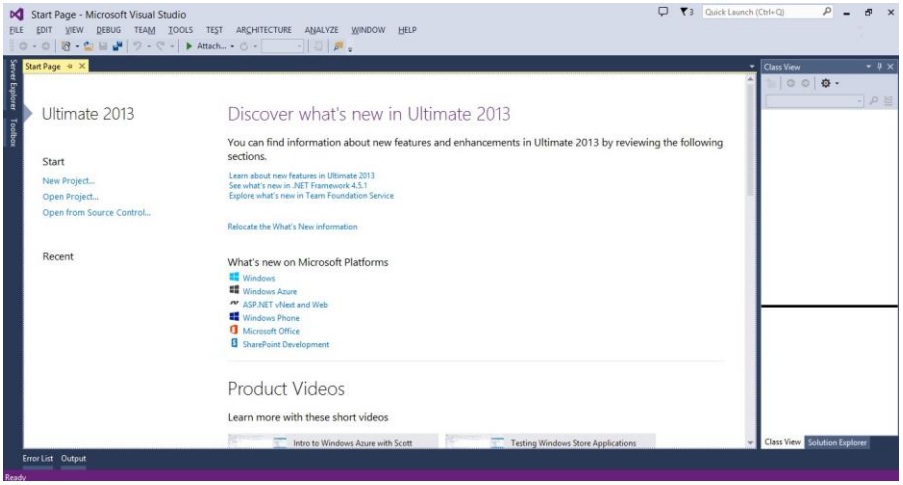
кеңейтімі бар арнайы объектілік файл жасайды. Бұл файлда машина тіліне аударылған бағдарламаның мәтіні бар.

Бірақ компьютер бұл файлмен тікелей жұмыс істемейді. Орналастыру немесе байланыстыру сияқты нәрсе бар. Байланыстырушы - бұл машина кодын (.OBJ кеңейтімі бар файлдан) және әр түрлі көмекші деректерді .EXE кеңейтімі бар бір орындалатын файлға жинайтын тағы бір арнайы бағдарлама. Мұндай файлды бөлек, тәуелсіз бағдарлама ретінде іске қосуға болады, ал компилятор оны іске қосуға қатыспайды.

Сіз үйрене бастаған C тілі - құрастырылған тіл. Біздің жағдайда, Microsoft Visual Studio қабығымен жұмыс жасағанда, компилятор автоматты түрде шақырылады және C тілінің командаларын «қолдың сәл қозғалысы» деп аталатын машина кодына аударуға мүмкіндік береді.

Қалам сынағы

C негізін қалаушы Брайан Керниган: «Жаңа бағдарламалау тілін үйренудің жалғыз жолы-оған бағдарламалар жазу»,-деді. Біз енді сізден бастайтын боламыз. Бағдарламалау әлемінде дәл солай болды, жаңа тілдегі бірінші бағдарлама - «Hello world!» бағдарламасы.



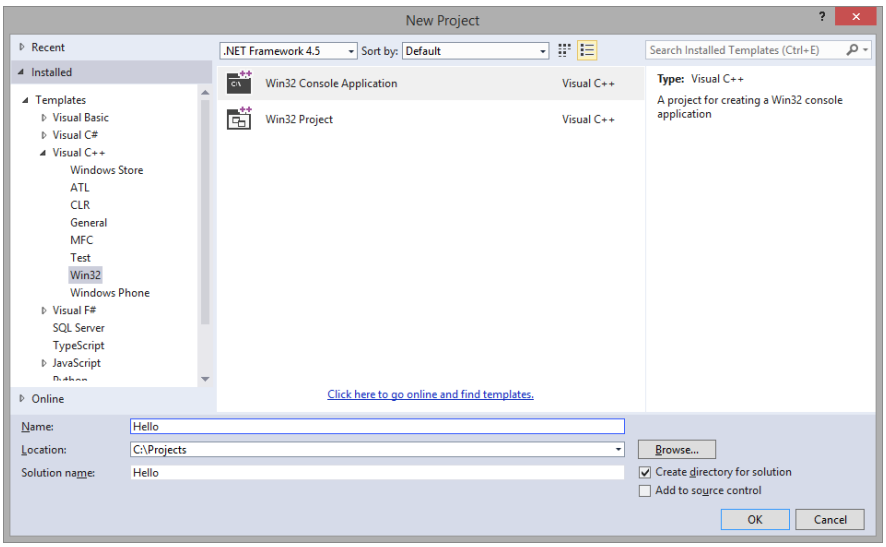
Сурет 2. Жаңа бағдарлама.

Бірінші бағдарламаны жазу үшін алдымен Microsoft Visual Studio 2013 бағдарламасының таңбашасын «Бастау» -> «Барлық бағдарламалар» -> «Microsoft Visual Studio 2013» мәзірінен немесе сізге таныс басқа жолмен іске қосу қажет. Іске қосылғаннан кейін біз суретте көрсетілгенге ұқсас суретті көреміз:

Енді, сайып келгенде, біздің бағдарламаны көрсететін жоба құруға тырысайық. Жобаның не екенін толығырақ кейінірек, үлкен бағдарламалар жазғанда түсінеміз. Әзірге біз жобаны бірнеше файлдарды біріктіру деп ойлайтын боламыз. Енді біртіндеп қадам жасайық:

- Қабықты жүктегеннен кейін Файл -> Жаңа -> Жоба мәзір элементін таңдаңыз. Сіздің алдыңызда диалогтық терезе пайда болады.

- Ашылған Жаңа жоба тілқатысу терезесінде Орнатылған үлгілер тізімінде Visual C ++ жобасының түрін -> Win32 -> Win32 консоль қосымшасын таңдаңыз.
- Орналасу өрісінде біз жобамыз қай дискіде және қай қалтада орналасатынын көрсетеміз. Мұны істеу үшін осы өріске C: \ Projects енгізіңіз (немесе сізге ыңғайлы барлық қалта жобалары сақталатын қалтаның атауы)
- Біздің ЖОБАға атау беріңіз - бұл үшін Атау өрісіне Hello жобасының атын енгізіңіз.
- Енді OK түймесін басуға болады.

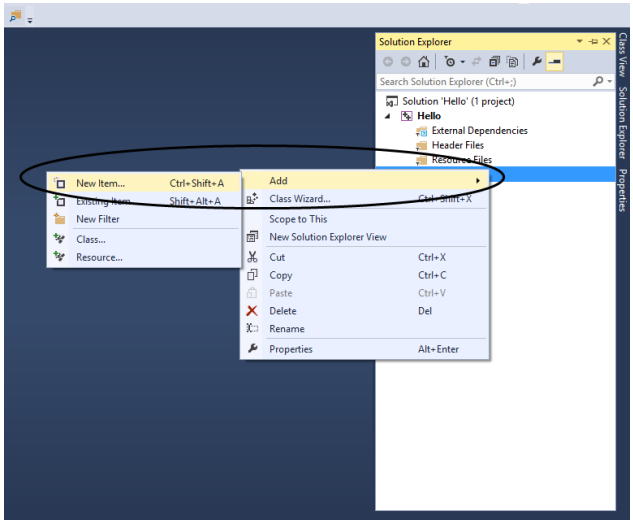


Сурет 3. Бағдарламаны орнату.

Сіздің алдыңызда жоба сипаттарын орнатуға арналған терезе ашылды - Қолданба параметрлері қойындысын таңдаңыз. Бос жоба құсбелгісін қойыңыз - бұл біз бос жоба құрып жатқанымызды білдіреді. Сондай -ақ, қауіпсіздікті дамытудың өмірлік циклінің (SDL) тексерулерінің құсбелгісін алып тастаңыз. Енді Аяқтау түймесін басыңыз.

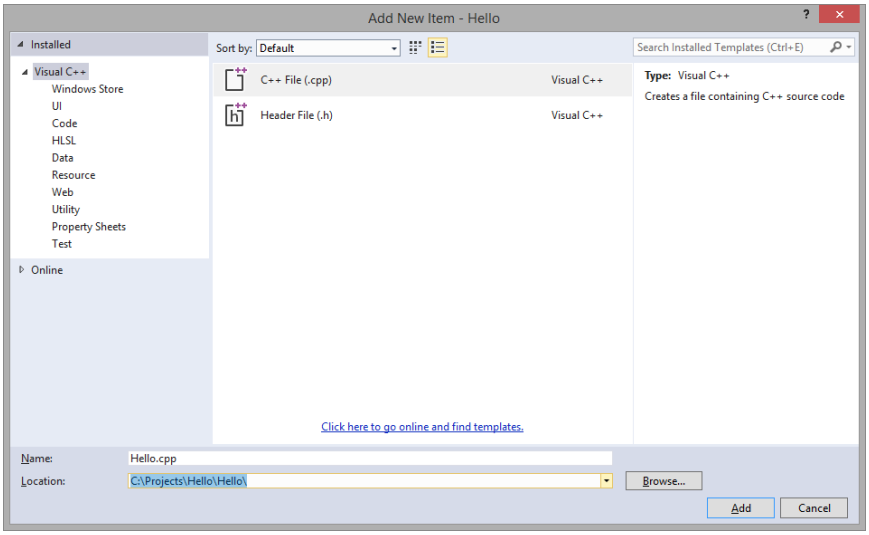
Сонымен, біз өз бағдарламамызды өткізетін орын дайындадық. Осымен тоқтамай, жобаға таза файл қосайық. Онда біз бағдарламамыздың мәтінін теремін. Ол үшін мына қадамдарды орындаңыз:

- Оң жақта Solution Explorer деп аталатын терезе пайда болды. Бұл терезеге қоңырау шалу үшін Ctrl + Alt + L пернелер тіркесімін пайдалануға болады. Бұл терезеде бастапқы файлдар деп аталатын қалтаны тінтуірдің оң жақ түймесімен басу керек.
- Ашылмалы мәзірде Add->Add New Item... тармағын таңдаңыз.



Сурет 4. Бағдарламаның интерфейсі.

- Файлды таңдау терезесі ашылады. Тағы да, сізде үлкен таңдау бар. Енді C ++ File (.cpp) белгісін таңдауды ұсынамыз (C / C ++ бағдарламасы бар файл).
- Атау мәтін жолағына Hello атауын енгізіңіз. Add түймесін басыңыз.



Сурет 5. Мәтін жазу аймағы.

Add түймесін басқаннан кейін сізде бірінші бағдарламаны теруге болатын мәтіндік аймақ болады.

Бірінші C бағдарламасының мысалы.

Бағдарламаны жазуды бастамас бұрын, ыңғайлы болу үшін түсініктеме түсінігін енгізейік. Түсініктемелер - бұл тек бағдарламашыға арналған бағдарлама жазбалары. Компилятор оларды елемейді. Мысалы, түсініктеме көмегімен бағдарламаның белгілі бір жолы не үшін қолданылатынын

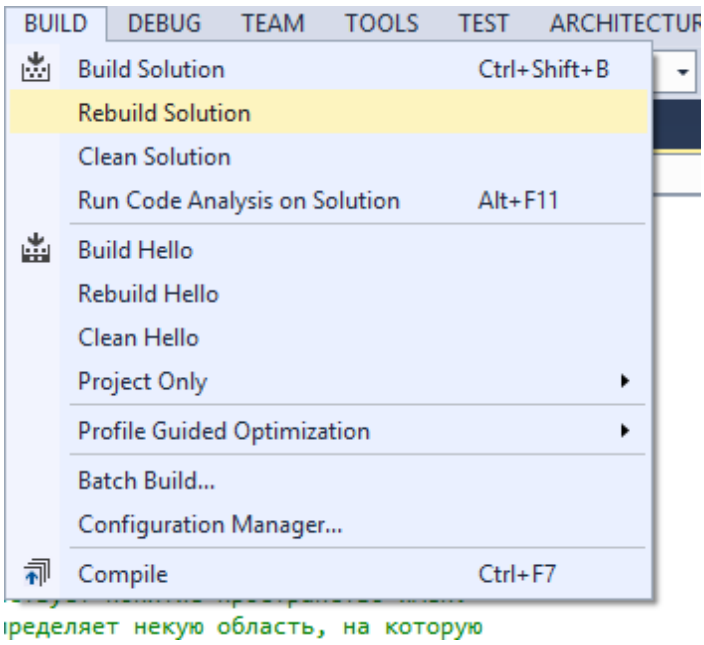
көрсетуге болады. Төмендегі мысал түсініктемелермен қалай дұрыс жұмыс жасау керектігін көрсетеді.

Енді пайда болатын мәтін жолағына келесі кодты енгізіңіз:

```
// Это комментарии к программе
// Они выделяются зеленым цветом
// Начинаются комментарии двумя черточками //
/* если необходимо создать многострочный
комментарий, используется конструкция
/* комментарий */
/* Данная строчка подключает в программу библиотеку под названием
 iostream. Библиотека - файл, в котором содержатся описания
 различных функций, реализованных другими программистами.
 Данная программа получила возможность использовать функции
 находящиеся в библиотеке iostream */
#include <iostream>
/* В языке C++ существует понятие пространство имен.
 Это пространство определяет некую область, на которую
 приходятся действия оператора или функции. Для того, чтобы использовать
 оператор, находящийся в определенном пространстве,
 необходимо подключить это пространство в свою программу.
 Ниже подключается пространство под названием std */
using namespace std;
void main() // Начало программы, отсюда программа начнет своё выполнение
// Весь текст программы располагается между фигурными скобками
{          //Это фигурная скобка
        // Следующая строка выводит на экран приветствие Hello, World!
       // Данное действие осуществляется с помощью cout<< Именно для его
  // работы подключалась библиотека и пространство имен, в которых он располагается
        cout<<"Hello, World!\n";
        // В конце команды стоит точка с запятой. Этим знаком ДОЛЖНА заканчиваться
       // каждая команда в языке С.
}
```

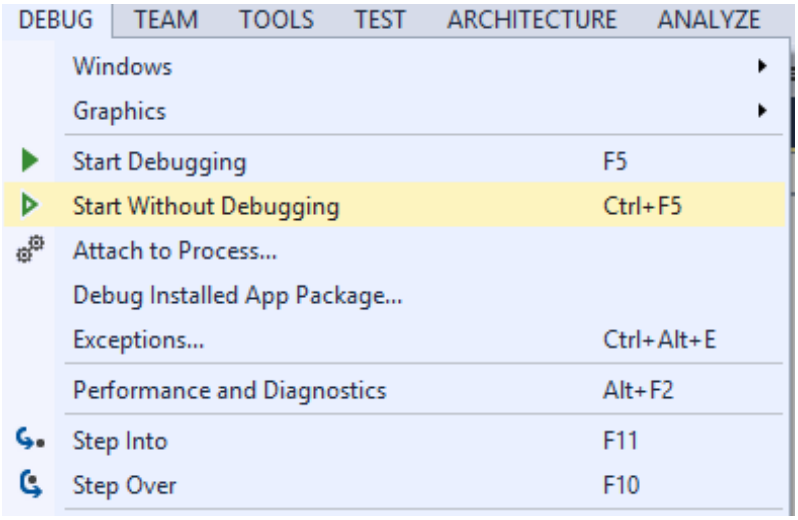
Сурет 6. Мәтін жолының коды.

Өздеріңіз білетіндей, компьютер тек машиналық кодтардың тілін түсінеді. Ал программаны компьютерде орындаудан бұрын оны машиналық кодтар тіліне аудару қажет. Сіз мұны біз емес, КОМПИЛЯТОР жасайтынын есіңізде сақтаңыз. С жобаларына көптеген файлдар кіреді. Жобаға енгізілген барлық файлдарды бірден құрастырайық. Мұны істеу үшін мәзір жолағында Build (құрастыру) мәзірінің тармағын таңдаңыз, содан кейін Rebuild Solution (бәрін қайта құру).



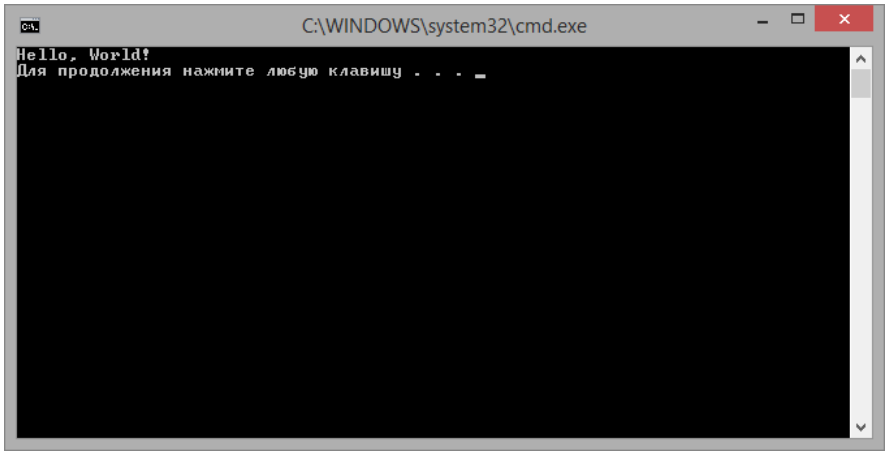
Сурет 7. Бағдарлама мәзірі.

Мәтінді дұрыс тердіңіз деп үміттенеміз. Біздің бағдарлама машина кодтарының тіліне сәтті аударылды және оны орындауға болады. Бағдарламаны орындау үшін оны іске қосу оңай. Debug мәзірінен Start Without Debugging таңдаңыз.



Сурет 8. Debug мәзірі.

Бағдарлама іске қосылады және сіз келесі терезені көресіз:

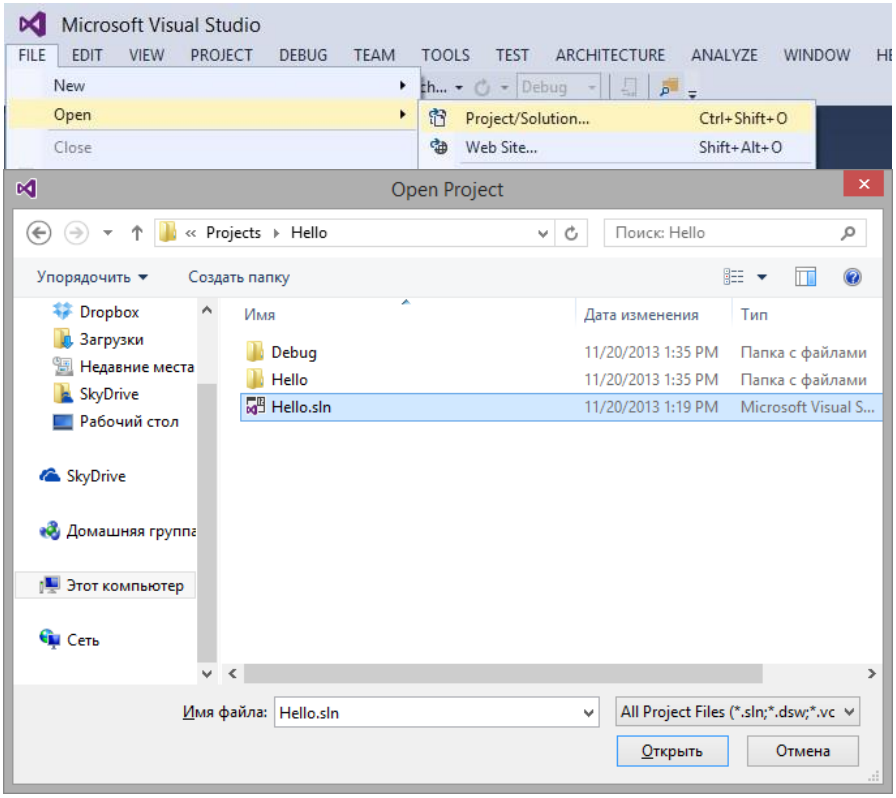


Сурет 9. Бағдарламаның алғашқы беті.

Құттықтаймын! Бұл өте оңай болды. Шын емес па? Бағдарламаны тоқтату үшін пернетақтадағы кез келген пернені басыңыз (Press any key to continue).

Сақталған жобаны ашу

Дискіде бұрын сақталған жобаны қалпына келтіру үшін Visual Studio бағдарламасын іске қосыңыз (егер сіз оны әлі жүктемеген болсаңыз). Файл мәзірінен Ашу -> Жоба / Шешім пәрменін таңдап, жобаңыздың атын көрсетіңіз.



Сурет 10. Бағдарламаның көмекшісі.

2. Деректерді шығару.

Сіз cout << командасының көмегімен экранда әр түрлі мәтіндік жолдарды көрсетуге болатынын білесіз. Алайда, компилятор мұндай команданы түсінуі үшін келесі үш негізгі ойды есте сақтау қажет:

Бағдарлама тақырыбында #include <iostream> жолы болуы керек

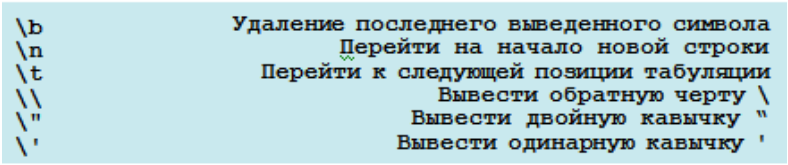
Команданы қолданар алдында cout командасы тиесілі аттар кеңістігін қосу керек.

using namespace std;

Біз cout << арқылы көрсеткіміз келетін жолды оны тырнақшаға алуымыз керек. Мысалға:

cout<<”здесь пишем то, что хотим”;

Cout << пәрмені жолдарды көрсетіп қана қоймайды, сонымен қатар оларды форматтауға мүмкіндік береді. Жолдың шығуын форматтау үшін \ таңбасы мен жолда орындалатын әрекетті анықтайтын символдың комбинациясынан тұратын арнайы басқару таңбалары қолданылады. Бұл басқару таңбалары Escape реттілігі деп аталады. Олардың кейбіреулері төменде келтірілген:



Сурет 11. Escape реттілігі

Соңғыүш Escape тізбегінің болуы әрқашан бастапқыда біраз таң қалдырады. Неліктен «немесе \ немесе '» жаза алатын кезде басқару таңбаларын қолдану керек? Жауап бетте жатыр, бұл үш символдың барлығы операторлар, егер олар «жазуға оңай» болса, онда компилятор оларды оператор ретінде қабылдайды. Мысалы, сөз бейнелі түрде қолданылғанда, ол тырнақшаға алынады. Сіз келесі мәтінді экранда көрсеткіңіз келеді делік:

The Man in red was "old friend" of John...

Егер сіз Escape реттілігін пайдаланбасаңыз, онда сіздің командаңыз келесідей болатыны анық:

cout<<”The Man in red was "old friend" of John...”;

Және бұл жақын арада қатеге әкеледі. Компилятор жолдың бір бөлігін ғана қабылдайды, атап айтқанда cout<<"The Man in red was". Ол қос тырнақшаны жабылған баға ретінде қарастырады және қалғандарын қате тіл синтаксисі ретінде қабылдайды. Әрине, мұндай бағдарлама іске қосылмайды. Дұрыс нұсқа келесідей:

cout<<”The Man in red was \"old friend\" of John...”;

Енді, cout << қайда Escape реттілігін қолдануға болатыны туралы сөйлесейік. Сіз білуіңіз керек ең маңызды нәрсе - бұл Escape дәйек сөзі әрқашан тырнақшаның ішінде болуы керек, себебі бұл мәтін, содан кейін сіздің мүмкіндіктеріңіз іс жүзінде шексіз. Мысалы, мына сияқты:

cout<<" My name is"<<" - Ira\n "; cout<<"I'm from Odessa\n ";
cout<<"My eyes are blue"<<"\n "<<"That`s all!!!";
Бағдарламаның нәтежесінде экранда келесі кодты көреміз:

```
My name is - Ira
I'm from Odessa
My eyes are blue
That`s all!!!
```

Сурет 12. Escape дәйек сөзі

cout<< қолданудың практикалық мысалы:

Біз зерттеген Escape тізбектерінің қысқаша мазмұнын көрсететін бағдарлама жазайық. Міне, экранда көргіміз келетін нәрсе:

```
\b                                          Backspace
\n                                          New line
\t                                      Horizontal tab
\\                                        Backslash \
\"                                Double quotation mark "
\'                                Single quotation mark '
```

Сурет 13. Escape тізбектерінің қысқаша мазмұны.

```
// Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main ()
{
/* Следующая команда через 4 табуляции выводит текст
   Escape Sequences
   и переводит вывод на следующую строку */
                                        cout<<"\t\t\t\tEscape Sequences\n";
            // Выводит пустую строчку
                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \b,
   и еще через 1 табуляцию Backspace
   Затем \n переводит вывод на следующую строку */
                                                cout<<"\t\t\\b"<<"\tBackspace\n";
            // Выводит пустую строку
                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \n,
   и еще через 1 табуляцию New line
   Затем \n переводит вывод на следующую строку */
                                                cout<<"\t\t\\n"<<"\tNew line\n";
           // Выводит пустую строку
                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \t,
   и еще через 1 табуляцию Horizontal tab
   Затем \n переводит вывод на следующую строку */
                                              cout<<"\t\t\\t"<<"\tHorizontal tab\n";
            // Выводит пустую строку
                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \\,
   и еще через 1 табуляцию Backslash \
   Затем \n переводит вывод на следующую строку */
                                             cout<<"\t\t\\\\"<<"\tBackslash \\\n";
            // Выводит пустую строку
                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \t,
   и еще через 1 табуляцию Double quotation mark "
   Затем \n переводит вывод на следующую строку */
                                        cout<<"\t\t\""<<"\tDouble quotation mark \"\n";
            // Выводит пустую строку
                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \',
   и еще через 1 табуляцию Single quotation mark '

 Затем \n переводит вывод на следующую строку */
                                        cout<<"\t\t\'"<<"\tSingle quotation mark \'\n";
            // Выводит пустую строку
                                                        cout<<"\n";
```

Сурет 14. Бардарлама коды.

Бағдарламаны құрастырайық (Build -> Rebuild Solution). Егер қателер көп болса, келесі ережелерді есте сақтаңыз:

Егер бағдарлама экранда хабарларды көрсететін болса, онда бағдарламаның басына #include <iostream> жолы жазылады және cout пәрмені тиесілі аттар кеңістігі қосылады (using namespace std;). Әр бағдарламада main () деп аталатын функция болуы керек. Бағдарлама осы функцияны орындаудан басталады.

Main () функциясының командалары бұйра жақша ішінде {}
Барлық командалар нүктелі үтірмен аяқталуы керек.
Және оны іске қосыңыз (Debug -> Start Without Debugging).
P. S.

Деректерді экранда көрсету кезінде біз тек латын әріптерін қолданатынымызды байқаған боларсыз. Бұл бағдарламаны Windows ОЖ -де жазғанымызға байланысты және оның орындалуы MS DOS жүйесінде жүзеге асады. Шындығында, кез келген операциялық жүйеде әр таңбаның өзіндік сандық коды болады. Жүйе оны дәл осы кодпен анықтайды. MS DOS пен Windows -те кириллицаның символдық кодтары сәйкес келмейді, сондықтан кириллицаны қолданатын бағдарлама дұрыс жұмыс істемейді. Мысалы, біз Windows -те жаздық:

cout<<"Утро";

Және экранда мыналар көрсетіледі:

µЄЁю

Бұл Windows -та, мысалы, о әрпінің 238 екендігімен оңай түсіндіріледі, ал DOS -та бұл код ю әрпіне сәйкес келеді. Латын коды екі операциялық жүйеде бірдей. Кейінірек біз бұл жағдайды қалай түзетуге болатынын білеміз.

C ++ 11 тілдік стандартында бағдарламашыларға арнайы таңбаларды экранда көрсетуге жаңа мүмкіндік берілді. Ол үшін «raw» жолдарды пайдалану қажет. Жолды «raw» деп жариялау компиляторға оны символ бойынша сипаттауды ұсынады. Жолды осындай жол деп жариялау үшін келесі форматты қолданыңыз.

R"(текст_строки)"

R бұл «raw» жол екенін көрсетеді. Жолдың мазмұны жақшаға алынуы керек. Міне, бірнеше мысалдар:

cout<<R"(hello\nworld)"; // на экране hello\nworld cout<<R("Test 'string'\t")"; // на экране "Test 'string'\t" cout<<R"((Such brackets))"; // на экране (Such brackets)

3. Деректер түрлері.

Сабақтың алдыңғы бөлімдерін оқығаннан кейін, экранда ештеңе көрсетпейтін бағдарлама жазу үшін сізге ешқандай шығын болмайды.

```
// Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
                                // вывод фразы "Поздравляем с хорошим началом!!! :)"
                                                // фраза выводится через три табуляции,
                                                 // затем добавляются две пустых строки
                                cout<<"\t\t\tCongratulation with good beginning!!! :)\n\n";
}
```

Сурет 15. Бағдарлама бастамасы.

Сіз жасай алатынның бәрі осы (әзірге).

Адам ешқашан қол жеткізген нәрсеге қанағаттанбауы керек. Шын мәнінде, сіз деректерді экранда қалай көрсету керектігін ғана емес, сонымен бірге бұл деректермен жұмыс істеуді де қызықтыруыңыз керек. Мысалы, кейбір есептеулерді орындаңыз. Қолсыз жонглер жоқ, доптардың бір бөлігі ауада жүргенде цирк әртісі қалғанын қолында ұстайды. Бір нәрсені (атап айтқанда деректерді) сақтау үшін сізде сақтау орны болуы керек. Біздің бағдарлама үшін бұл сақтау жедел жады болады. Бір жерге бірдеңе қоймас бұрын, дұрыс қаптаманы таңдау керек. Сіз сіріңке қорабына сүт құйып алуыңыз екіталай делік. Бағдарламалау кезінде ақпаратты жедел жадқа салмас бұрын, сіз бұл ақпараттың сипатын міндетті түрде анықтауыңыз керек. Сонымен, мәліметтер түрлері.

Деректер түрі - бұл максималды өлшемді (байтпен) және бағдарламада қолданылатын ақпарат түрін анықтайтын ұғым.

Бағдарламалау сыртқы әлемнің объектілерін көрсетеді, оларды айтарлықтай жеңілдетеді. Зерттеудің басында біз шынымен де сіз талай рет кездескен нәрселерге кезігеміз. Барлық деректер түрлерін шартты түрде келесі топтарға бөлейік:

1. Сандық.
2. Символдық.
3. Логикалық.

Әрі қарай, біз С тілінде деректер түрлерін белгілеу үшін қолданылатын бірнеше кілт сөздерді қарастырамыз.

Сандық түрлері.

Өздеріңіз білетіндей, сандар бүтін және нақты болуы мүмкін. Біз нақты сандарды өзгермелі сандар деп атаймыз. Бүтін бөлікті бөлшек бөліктен бөлетін үтір нүктеге ауыстырылғанын ерекше атап өтеміз. Мысалы, С тілінде 7,8 7.8 деп жазылған.

Біз нақты сандардың мәндерін сақтайтын айнымалылар float немесе double түріндегі жарияланады. Бұл түрлердің айырмашылығы неде? Қалқымалы түрі бір дәлдіктегі өзгермелі нүктелі сандарды сипаттайды, ал қосарлылық-екі дәлдіктегі сандар. Түсіндіру үшін, математикада дәлдік санды білдіретін цифрлар санымен анықталады. Қос дәлдік - бұл цифрлар санынан кәдімгіге қарағанда екі есе көп болатын сандарды көрсету әдісі. Міне, өзгермелі нүктелердің типтік сипаттамалары:

Түсіндіру	Типі	Өлшемі байтпен
нақты дәл сандарды сипаттайды	float	4
нақты дәл сандарды қос дәлдікпен сипаттайды	double	8

Нақты мәліметтерден басқа, С бүтін мәліметтерді жариялаудың үш түрін ұсынады. Кестеде бұл түрлердің негізгі сипаттамалары көрсетілген:

Түсіндіру	Типі	Өлшемі байтпен	Мәндер ауқымы
бүтін сандарды сипаттайды	int	4	от -2147483648 до 2147483647
қысқа бүтін сандарды сипаттайды	short	2	от -32768 до 32767
ұзын бүтін сандарды сипаттайды	long	4	от -2147483648 до 2147483647
ұзын бүтін сандарды сипаттайды	long long	8	от –9,223,372,036, 854,775,808 до 9,223,372, 036,854,775,807

Таңба түрі.

Түр тек бір таңбаны сақтауға арналған. Сізге бірден ескертейік - С -де жолдарды сақтаудың түрі жоқ.

Түсіндіру	Типі	Өлшемі байтпен
символдарды сипаттайды	char	1

Логикалық түрі.

Түр логикалық мәліметтерді сақтауға арналған. Біз онымен толығырақ кейінірек танысамыз. Логикалық деректер екі мәннің бірін қабылдай алады: ақиқат (true) немесе жалған (false).

Түсіндіру	Типі	Өлшемі байтпен	Мәні
логикалық мәндерді сипаттайды	bool	1	true false

Ескерту: Егер сіз теріс мәндерді деректер түрінің диапазонынан алып тастағыңыз келсе, түрінің атауын unsigned сөзбен жалғаңыз. Мысалы, unsigned int. Бұл түр 0 -ден 4294967294 -ке дейінгі оң мәндерді ғана қамтиды.

Сонымен, біз мәліметтердің қандай түрлерін және оларды белгілеу үшін С кілт сөздерінің қолданылатынын білдік. Қорытындылай келе, С тілінің регистрді ескеретінін атап өту керек (яғни бас әріптер мен төменгі әріптер онда бірдей емес). Назар аударыңыз, жоғарыда сипатталған барлық деректер түрлері кіші әріппен жазылады. Бұған назар аударыңыз, себебі int - бұл деректер түрі, ал INT - қате.

Келесі тақырыпта біз деректер түрлерін практикада қолдануды қарастырамыз.

4. Айнымалылар және тұрақтылар

Сонымен, біз өзгеретін деректерді АЙНЫМАЛЫЛАР, ал тұрақты деректерді - ТҰРАҚТЫЛАР деп атауға келісеміз.

Айнымалы - өз аты бар және өзгертілуі мүмкін мәліметтерді сақтауға арналған жедел жадының аймағы.

Тұрақты - бұл өз аты бар және тұрақты деректерді сақтауға арналған кездейсоқ қол жетімді жад аймағы.

Тұрақтылардың мысалы мынада: әркім аптадағы күндер мен жылдағы айлардың санын біледі ... Ол ешбір жағдайда өзгермейді, сондықтан бұл мәндер тұрақтылар.

Бірақ біздің жасымыз - өзгермелі сан. Бүгін біреу 26 жаста, ал бір жылдан кейін 27 болады.

Анықтамалардан жадыдағы деректерді іздеу үшін оларға аттар берілетіні белгілі болады (багаждағы вагондағы заттар тегтермен қамтамасыз етілгеніне ұқсастығы бойынша). Бағдарламалау ортасында оларды идентификаторлар деп атайды. Жаңа туған нәресте үшін ата -аналардың шешетін бірінші мәселесі - оған есім таңдау. Есім адамның мінезінде із қалдыра ма, оның тағдырында күрделі және даулы мәселе. Сіз бұл мәселе бойынша мүлдем қарама -қайшы пікірлерді ести аласыз. Бірақ жаңа айнымалыға (тұрақтыға) берілген атаудың (идентификатордың) интуитивті болуы және айнымалының мақсатын түсіндіруі керек екендігіне тәжірибелі бағдарламашы таласа алмайды.

Атау ережелері.

Атауында тек келесі таңбалар рұқсат етілген:

Латын әліпбиінің бас және кіші әріптері. Сонымен қатар, тілдің регистрге сезімталдығы туралы ұмытпаңыз. Мысалы, Age пен age - бұл екі түрлі атау.

Фигуралар. Бірақ цифрды бірінші таңба ретінде қолдануға болмайды. Яғни Name1 жарамды, 1Name жоқ.

«_» асты сызығы. Мәселе мынада, сіз бос орын символ екенін есте ұстауыңыз керек және айнымалы атауында бұл таңбаға жол берілмейді. Оның орнына астын сызу қойылады, бұл есімдердің мәнерлілігін жақсартады. Мысалы, ageofman мен Age_Of_Man салыстырыңыз.

Айнымалы атауды анықтау кезінде мыналарды есте сақтаңыз:

1. Бағдарламалау тілінің кілт сөздерімен айнымалыны шақыруға болмайды. Кілт сөз - программалау тілінің синтаксисі үшін сақталған сөз (int, float, double, т.б.). Visual Studio -да кілт сөздер көк түспен ерекшеленеді, бұл кем дегенде шатасуға әкеледі.

2. Бір аттас екі идентификатордың болуы жағымсыз.

3. Сіз жарамдылардан басқа басқа таңбаларды қолдана алмайсыз. (жоғарыдан қараңыз).

Айнымалылар мен тұрақтыларды жариялау және қолдану.

Енді бізде айнымалыны құру (жариялау) туралы барлық ақпарат бар. Жалпы синтаксистің не екенін анықтау ғана қалады:

тип_данных имя_переменной; — бұл жағдайда ЖЖҚ -да өлшемі көрсетілген түрге сәйкес келетін ұяшық бөлінеді. Және бұл ұяшыққа сіз таңдаған атау беріледі. Онда не болады? Сіз жасаған айнымалыға операциялық жүйе анықтайтын кездейсоқ сан толтырылады. Арнайы тағайындау операторының көмегімен айнымалы мәнді басқа мәнмен толтырғанша, бұл сан жадыда сақталады

тип_данных имя_переменной=значение; — сондай -ақ мұндай мүмкіндік бар - айнымалы мәнді тікелей құру кезінде толтыру. Біз бұл процесті инициализация деп атаймыз.

const тип_данных имя_переменной=значение; — бұл тұрақты мәлімдеме. Ең бастысы, деректер түріне қарамастан, const кілт сөзінің префиксі болады. Сонымен қатар, константа құру кезінде инициализациялануы керек. Кейін оның құнын өзгерту мүмкін болмайды.

Экранда айнымалының мәнін көрсету.

Экранда айнымалының мәнін көрсету cout << көмегімен жүзеге асады.

```
cout<<имя_переменной; // кавычки в данном случае не указывают
```

<< арқылы бірнеше айнымалылардың мазмұнын көрсетуге болады.

```
cout<<имя_переменной1<<имя_переменной 2;
// кавычки в данном случае не указывают
```

Сіз айнымалылардың мазмұнын мәтіндік хабарлармен және << арқылы Escape- реттіліктерін көрсете аласыз.

```
cout<<"Текст"<<имя_переменной1
<<"Текст"<<имя_переменной2<<"\n";
```

Тұрақтылардың мазмұнын көрсету айнымалылармен толық ұқсастықта жүзеге асады.

Практикалық мысалдар

Мұнда әр түрлі деректер типтері үшін айнымалылар мен тұрақтыларды құру мен инициализациялаудың бірнеше мысалдары келтірілген.

Бүтін айнымалылар мен тұрақтылар.

Біз барлық жерде бүтін сандарды көреміз: жасы, орындықтар саны, бөлмелер саны, аптаның күндер саны және т.б.

Бүтін сандарды сақтайтын айнымалылар келесідей ЖАРИЯЛАНАДЫ:

int Age;

Бұл желі не туралы айтады? Age деп аталатын айнымалы бүтін мәнді сақтайды. Int сөзі Age деп аталатын айнымалы мәннің TYPE мәлімдейді.

Енді, мысалы, Age айнымалысына 34 қосқымыз келеді, мұны қалай істейміз?

Age =34;

Эта строчка читается так: «Переменной Age присвоить значение 34».

Бұл жол келесідей: «Age айнымалы мәнін 34 -ке орнатыңыз».

Тағы да тағайындау операторына қарайық: Age = 34;

Теңдік белгісінің сол жағында мән берілген айнымалының атауы орналасқан. Ал оң жақта - берілген мән.

Бүтін сан сақталатын тұрақты мән келесі түрде жарияланады:

const int Count_Days_in_Week=7;

Бұл желі не туралы айтады? Const сөзі тұрақты мән жарияланып тұрғанын көрсетеді. int тұрақтысының бүтін сан болатынын айтады. Осыдан кейін Count_Days_in_Week тұрақтысының атауы және оның мәні 7 келеді.

Енді айнымалының мәнін қалай есептеу керектігін қарастырайық. Ол не үшін? Қарапайым мысал: 2000 жылы қанша сағатты қалай есептеу керек? Сіз бұл санды өзіңіз есептегіңіз келе ме?

Компьютерді өз бетімен жасау өте оңай. Бізге бұл есептің формуласын ғана жазу қажет.

2000 жылы 366 күн, тәулік бойы. Сонымен 2000 жылғы сағат санын есептеу формуласы: 366 24 -ке көбейтілді. Си тілінде * көбейту белгісі ретінде қолданылады (жұлдызша, Shift + 8 комбинациясы).

Біз 2000 жылы қанша сағатты есептейтін бағдарламаны жасаймыз.

Бағдарламаны жасамас бұрын оның алгоритміне қысқаша эскиз жасау ұсынылады.

Алгоритм - бұл берілген мәселені шешуге бағытталған әрекеттер тізбегі.

Берілген: бір күндегі күндер саны - 366. Бұл мән өзгермейді, сондықтан оны DayIn_2000Year деп аталатын бүтін сан тұрақтысы деп жариялайық. тәуліктегі сағат саны 24. Ол да өзгермейді. Оны HourInDay деп аталатын бүтін сан тұрақтысы деп жариялайық. Біздің бағдарламада жалғыз айнымалы болады, біз оған есептің нәтижесін жазамыз. Бұл айнымалыны HourIn_Year2000 деп атайық. Ол барлық типте болады. (int)

Алгоритм келесідей болады:

1. Айнымалылар мен тұрақтыларды декларациялау және инициализациялау.
2. Нәтижені есептеу.
3. Нәтижені көрсету.

Айнымалылардың атауларын өзіңіз ойлап табуға болады (айнымалы атауларды құру ережелерін ғана ұмытпаңыз).

Енді, әдеттегідей, жаңа жоба құрып, келесі кодты енгізейік:

```
// Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
        // вывод пустой строки
        cout<<"\n";
        //Объявляем целочисленные константы
        int DayIn_2000Year=366;
        int HourInDay=24;
       //объявляем целочисленную переменную
      int  HourIn_Year2000;
      // вычисляем искомое значение и
      // помещаем его в переменную HourIn_Year2000
      HourIn_Year2000=DayIn_2000Year*HourInDay;
      // выводим значение переменной HourIn_Year2000 на экран
      cout<<"\t\t In 2000 year "<< HourIn_Year2000;
      cout<<" hours\n ";
```

Сурет 16. Айнымалы атауларды құру

Нақты айнымалылар мен тұрақтылар.

Декларация мен инициализацияның мысалы.

```
float Weight; Weight=12.3452;
double weight_atom;
weight_atom= 2.5194e+017;
```

2.5194e + 017 саны нені білдіреді?

Бұл нақты сандардың қысқаша жазбасы. Ол сандарды жазудың экспоненциалды түрі деп аталады. Біз сізге жазылғанды декодтаудың құпиясын айтамыз. Бұл таңбалар жиыны 251940000000000000 немесе 2.1594 × 1017 санын сипаттайды.

–3.4E008 3.4 × 10-8 дегенді білдіреді, бұл 3.4: 108-ге ұқсас.

–1.5E + 003 –1.5 × 103 дегенді білдіреді.

Қабаттар -3,4 × 1038 -ден 3,4 × 1038 -ге дейін болуы мүмкін.

–3,4 × -38 -тен 3,4 × 10–38 -ге дейінгі мәндер нөлге тең деп саналады.

Енді практикада нақты сандармен жұмыс жасайық:

Сатып алу бағасын есептейтін бағдарлама жазайық. Бағдарлама өнімнің бағасын (Cost), сатып алынған өнімнің санын (Count), ал дисконтты (Discount) ескере отырып, сатып алу құнын (Price) есептеуге рұқсат етсін. Pokupka жаңа жобасын құрып, келесі бағдарламаның мәтінін енгізейік

```
// Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main ()
{
        //Объявляем переменную Discount
        float Discount=0.05;
        //Объявляем переменную Cost
        float Cost=10.50;
        //Объявляем переменную Count
        int Count=5;
        //Объявляем переменную Price
        float Price;
         //Вычисляем значение переменной Price
          Price=Count*Cost-Count*Cost*Discount;
         // Выводим итоговую стоимость товара со скидкой
         cout<<"Please, pay:"<<Price<<"\n";
}
```

Сурет 17. Бағдарлама өнімнің бағасы

Бағдарламаны құрастырып, оны орындауға жіберіңіз. Экранда не көру керек, төменде көрсетілген.

Сурет 18. Бағдарламаны құрастыру

Символдық және логикалық айнымалылар мен тұрақтылар.

Біз тек хабарландыру мен инициализацияны қамтамасыз етеміз.

```
// Логическая переменная bool Flag;
Flag=true;
// Один символ всегда указывается в одинарных кавычках char Symbol='A';
```

/* Escape - последовательность рассматривается компилятором, как один символ и соответственно может быть записана в переменную или константу типа char*/
const char NewLine='\n'; cout<<NewLine// показывает пустую строку

5. Мәліметтерді енгізу.

Сіз компьютер экранында ақпаратты көрсету операциясымен таныссыз - cout, бірақ көптеген бағдарламалар тек кейбір ақпаратты экранда шығаруды ғана емес, сонымен қатар компьютерге пернетақтадан кез келген мәліметтерді енгізуді талап етеді. Алдыңғы бөлімде жеңілдіктерді есептеу бағдарламасы берілген. Әрине, бағдарламаның орындалу кезеңінде тауарлардың бағасы мен саны сияқты параметрлерді пернетақтадан енгізу жақсы болар еді. Мұны қалай жасауға болатынын қарастырайық. Егер компьютерге деректерді енгізу қажет болса, cin командасын қолданамыз. Мен оны қалай қолданамын? Енгізу мәлімдемесінің синтаксисі:

cin>>имя_переменной;

имя_переменной пернетақтадан енгізілген деректерді орналастыратын айнымалыны көрсетеді:

Мысалы:

cin>>Age;

Бұл команда пернетақтадан енгізілген санды Age деп аталатын айнымалыға орналастырады. Number айнымалысына санды енгізу үшін келесі пәрменді енгізу қажет:

cin>>Number;

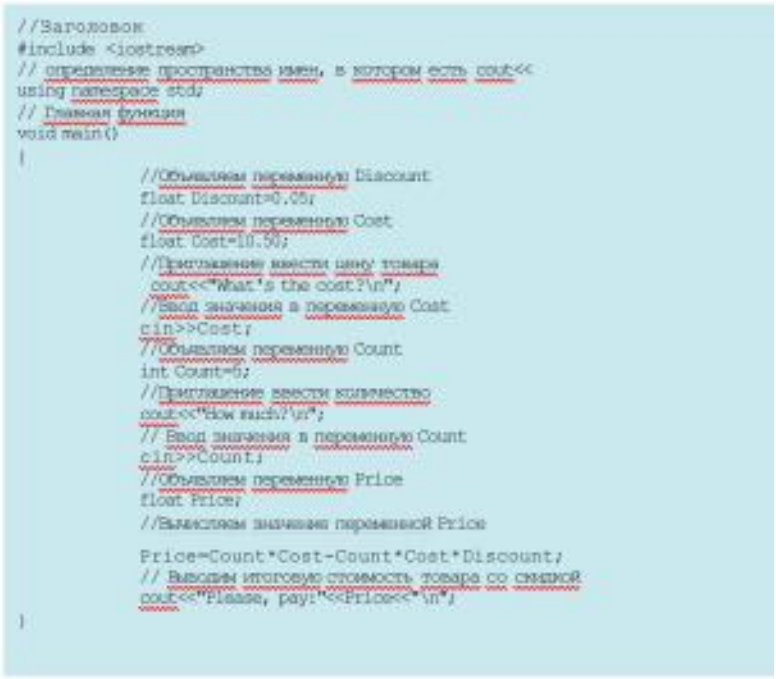
Бірнеше айнымалыны бірден енгізу келесі түрде жазылады:

cin>>имя_переменной1>>имя_переменной2>>...>>имя_переменной N;

Айнымалы атаулар тізімінде деректерді пернетақтадан енгізгіңіз келетін барлық айнымалылардың атаулары болуы керек. Атаулар тізімі >> таңбалар комбинациясымен бөлінген айнымалы атаулардың кез келген санынан тұруы мүмкін.

cin>>Quantity>>Price>>Discount;

Pokupka бағдарламасына пернетақтамен мәліметтерді енгізуін қосайық:



```
//Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
        //Объявляем переменную Discount
        float Discount=0.05;
        //Объявляем переменную Cost
        float Cost=10.50;
        //Приглашение ввести цену товара
         cout<<"What's the cost?\n";
        //Ввод значения в переменную Cost
        cin>>Cost;
        //Объявляем переменную Count
        int Count=0;
        //Приглашение ввести количество
        cout<<"How much?\n";
        // Ввод значения в переменную Count
        cin>>Count;
        //Объявляем переменную Price
        float Price;
        //Вычисляем значение переменной Price

        Price=Count*Cost-Count*Cost*Discount;
        // Выводим итоговую стоимость товара со скидкой
        cout<<"Please, pay!"<<Price<<'\n';
}
```

Сурет 19. Мәліметтерді енгізу

Бағдарламаны құрастырыңыз. Біздің бағдарламаны қолдану оңай. Тек кез келген нөмірді енгізіңіз, ол үнемі компьютердің нөмірі жоғары болып шығады және ол жеңеді. Бағдарламаны іске қосқан кезде экранда мынаны көресіз, егер сіз "Enter a number:" жолына 67 санын енгізсеңіз:

Let's play!
Enter a number: 67 I have 68
I'm winner!
Press any key to continue...

Неліктен ол үнемі жеңіске жетеді? Жолға қарайық:

cout<<"I have "<<i+1<<"\n";

Ол пернетақтадан енгізілген мәні і айнымалы мәнін көрсетеді, яғни компьютер әрқашан пернетақтадан енгізгеннен 1 -ге артық санды көрсетеді.

Егер сіз осы команданың і + 1 өрнегін і-1 өрнегімен алмастырсаңыз, онда сіз әрқашан ұтасыз, өйткені компьютер көрсететін сан әрқашан пернетақтадан енгізілген саннан кем болады.

Қорытындылай келе, біз сіздің назарыңызды + (плюс) және - (минус) операторларына аударғымыз келеді. Олар қосу мен азайту үшін қолданылады. С -де бөлуге арналған оператор бар

- /. Бұл ақпарат үй тапсырмасын орындауға көмектеседі және келесі сабақтарда операторлар туралы толығырақ айтатын боламыз.

Әдебиеттер.

Әдебиеттер - бұл бағдарлама өзгерте алмайтын тұрақты мәндер. Әдебиеттер C тілінің әр түрі үшін бар, оның ішінде символдар мен логикалық типтер, бүтін сандар және өзгермелі нүкте сандары. Парадоксальды түрде, C -де жолдарды сақтауға арналған деректер түрі жоқ, бірақ string литералдары жасайды.

Кейбір мысалдар:

5	бүтін сан-int
5l	l немесе L ұзақ дегенді білдіреді
true	логикалық сөзбе -bool
5.0	өзгермелі нүкте, double деп түсініледі
5.0f	f немесе F - өзгермелі нүкте, float деп түсініледі
0.3e–2	өзгермелі нүкте әріптік double, e немесе E экспоненциалды бөлікті ажыратады
'd'	сөзбе -сөз
"Visual"	жолдың литералы - тырнақшаға алынған ерікті таңбалар жиыны. Компилятор оны символдар жиынтығы ретінде қабылдайды және тырнақшаларда кейбір кілт сөздер мен амалдар болса да, оны ешқандай жолмен өңдемейді.

Оператор туралы түсінік

Өткен сабақта сіз айнымалы ұғымы мен мәліметтер типі туралы білдіңіз. Сонымен қатар, сабақ мысалдарында, сондай -ақ үй тапсырмасында біз айнымалылар бойынша белгілі бір әрекеттерді орындадық, яғни мәліметтерге операция жасадық. Оператор мен операция сөздерінің шығу тегі бір екені анық, сондықтан қарапайым логикаға сәйкес: Өткен сабақта сіз айнымалы ұғымы мен мәліметтер типі туралы білдіңіз. Сонымен қатар, сабақ мысалдарында, сондай -ақ үй тапсырмасында біз айнымалылар бойынша белгілі бір әрекеттерді орындадық, яғни мәліметтерге операция жасадық. Оператор мен операция сөздерінің шығу тегі бір екені анық, сондықтан қарапайым логикаға сәйкес:

Оператор - бұл белгілі бір нәтижеге әкелетін мәліметтер бойынша әр түрлі әрекеттерді орындауға мүмкіндік беретін тілдік құрылым.

Барлық операторлар әдетте олардың әрекеттеріне қарай топтарға бөлінеді. Мысалы, арифметикалық амалдар - бұл мәліметтерге арифметикалық амалдарды орындауға мүмкіндік беретін операциялар (қосу, азайту және т.б.). Біз болашақта C тілінде ұсынылатын барлық осындай топтар туралы сөйлесетін боламыз. Әзірге айнымалылардың мазмұнына әсеріне қарамастан, барлық операторлардың кеңірек жіктелуін талқылау қажет. Осылайша, барлық операторлар бөлінеді:

1. Біртұтас — бір ғана операнд қажет операторлар (әрекет етілетін деректер). Сіз мектептегі математика курсының біртұтас операторының мысалын білесіз- бірлік минус, ол санды теріс (3 және –3) немесе оңға (- (- 3))

айналдыруға мүмкіндік береді. Яғни, біртұтас оператордың жалпы синтаксисі:

оператор операнд; или операнд оператор;

2. Екілік — оператордың сол және оң жағында екі операнд қажет болатын операторлар. Сіз осындай көптеген операторларды білесіз - олар +, - , *, т.б. және олардың жалпы синтаксисін келесі түрде ұсынуға болады:

операнд оператор операнд;

3. Үштік — үш операнд қажет операторлар. Си программалау тілінде осындай бір ғана оператор бар, біз оның синтаксисімен сәл кейінірек танысамыз.

Басымдық

Барлық операторларға басымдық беріледі. Төменде операторлар басымдыққа сәйкес келеді. Біз кейбір сабақтармен бүгінгі сабақта тереңірек танысамыз, ал басқалары қосымша дайындық барысында үйренетін болады. Әрине, бұл кесте тілдің барлық операторларын білдірмейді, бірақ әзірге біз үшін ең маңызды.

Операция белгісі	
Ең жоғары басымдық	Ең жоғары басымдық
() [] . ->	^
! *(ун) – (ун) ~ ++ —	|
% * /	&&
+ –	||
<< >>	?:
< > <= >=	= += –= *= /= %= &= |= ^= >>= <<=
!= ==	
&	Ең төменгі басымдық

Енді операторлар саласындағы білімнің іргетасы қаланды, сіз соңғысын толығырақ зерттеуге, яғни сабақтың келесі бөліміне көшуге болады.

Сандармен арифметикалық амалдар

Жақсы ұмытылған ескі …

Ендеше, бастайық. Жоғарыда айтылғандай, арифметикалық операциялар - бұл мәліметтерге арифметикалық амалдарды орындауға мүмкіндік беретін операциялар. Олардың көпшілігі сізге бала кезінен таныс, дегенмен төмендегі кестені қолдана отырып, өз білімімізді жүйелейік.

Операция атауы	Си тілінде белгілеу үшін қолданылатын белгі.	Қысқаша сипаттамасы. Мысал.
Қосу	+	Екі мәнді бірге қосады, нәтиже - операндтардың қосындысы: 5 + 18 нәтижесі 23 -те.
Азайту	–	Мәнді оператордың мәнінен оңға қарай азайтады. Нәтиже- операндтардың айырмашылығы: 20-15 нәтиже 5.
Көбейту	*	Екі мәнді көбейтеді, нәтиже - операндтардың туындысы: 5 * 10 нәтижесі 50.
Бөлім	/	Сол жақтағы мәнді оператордың оң жағындағы мәнге бөледі. Мысалы: 20/4 нәтижесі - 5.
Модульдік бөлу	%	Бұл операцияның нәтижесі бүтін санның қалған бөлігі болып табылады, мысалы, егер біз 11 -ді 3 -ке бөлсек, онда біз 3 бүтін бөлікті аламыз (3 * 3 = 9 болғандықтан), қалғаны 2 болады, бұл сан нәтиже болады модульдік бөлінудің: 11/3 3 бүтін 2 қалдықта 11% 3 = 2 (қалдығы)

Ескерту:

1. Бөлу модулінің жұмысы бүтін мәліметтерге ғана қолданыла алады. Бұл ережені бұзу әрекеттері компиляция уақытында қатеге әкеледі.

2. Егер кіші сан%көмегімен үлкен санға бөлінсе, нәтиже ең кіші санның өзі болады. 3% 10 = 3

3. Модульді нөлге бөлу мүмкін емес, бұл бағдарламаның орындалу сатысында дұрыс жұмыс істемеуіне әкеледі.

Инкремент және декремент.

Жоғарыда аталған барлық амалдар екілік болды, бірақ біртұтас арифметикалық амалдар да бар, мектеп курсында мұндай операциялар жоқ, дегенмен олар өте қарапайым:

Инкремент - ++ белгісімен белгіленеді. Бұл оператор кез келген айнымалының мазмұнын көбейтеді және айнымалының мәнін қайта жазады. Мысалға:

```
int a=8;
cout<<a; // на экране число 8 a++;
cout<<a; // на экране число 9
```

Декремент — құрылысымен белгіленеді -. Бұл оператор кез келген айнымалының мазмұнын азайтады және айнымалының мәнін қайта жазады. Мысалға:

```
int a=8;
cout<<a; // на экране число 8 a--;
cout<<a; // на экране число 7
```

Қарапайым, солай емес пе?! Мұндай өрнектерді келесі түрде де көрсетуге болады: a = a + 1 немесе a = a-1. Айта кету керек, литералдар үшін өсім де, кему де қолданылмайды, себебі 5 = 5 + 1 келесідей әрекет ету қисынды емес. Бұл анық қате. Алайда, бұл өсу мен кему бойынша танысуымыз аяқталған жоқ. Сабақтың соңғы бөлімінде біз біртұтас операторлық синтаксисі бұл ғана емес,

операнд оператор;

сонымен қатар осындай болуы мүмкін екенін білдік.

оператор операнд;

Мұндай белгілеу формалары постфикс, (оператор мәннен кейін орналасады) және префикс (оператор мәннен бұрын орналасқан) деп аталады. Қосудың да, кемудің де екі формасы бар. Пішіндердің қандай айырмашылықтары бар екенін және қандай жағдайда бұл айырмашылықтар маңызды екенін қарастырайық.

1 мысал.

```
int a=8;
cout<<a; // на экране число 8 a++;
cout<<a; // на экране число 9
++a;
cout<<a; // на экране число 10
```

Бұл мысалда префикс пен постфикстің формалары арасында ешқандай айырмашылық жоқ. Бірінші және екінші жағдайда a айнымалысының мәні жай ғана біреуіне артады. Операторлардың әр түрлі формаларын қолданудың мағынасы жолда операторлардың өзінен басқа қандай да бір команда болғанда ғана пайда болады.

2 мысал.

```
int a=8;
cout<<++a; // на экране число 9 cout<<a++; // на экране число 9 cout<<a;
// на экране число 10
```

Мысалға кіріспес бұрын, үш ережені құрайық:

1. Си тілінде команданы орындау принципі екіұшты. Сондықтан төменде кейбір операторлардың әрекет ету бағыттарының кестесі берілген:

2. Егер өсірудің немесе азайтудың постфикс түрінен басқа, жолда басқа команда болса, онда бұл команда алдымен, ал команданың жолдағы орналасуына қарамастан, тек содан кейін ғана өсу немесе кему орындалады.

3. Егер ұлғайту немесе азайту префиксінен басқа жолда басқа команда болса, онда жолдағы барлық командалар операторлардың басымдығына сәйкес оңнан солға қарай орындалады.

ОПЕРАТОР	БАҒЫТ
() [] . ->	СОЛДАН ОҢҒА
* / % + –	
<< >> & ^ |	
<< = >> = ==! =	
&& ||	
Унарный – унарный + ! ++ —	ОҢНАН СОЛҒА
?:	
+ = – = / = * = % = & = ^ = | =	

Енді мысал туралы толығырақ:

- Бастапқыда айнымалының мәні 8 болады.
- Command cout << ++ a; ұлғайту операторының префикс формасын қамтиды, сондықтан жоғарыда сипатталған үшінші ережені қолдана отырып, біз алдымен a айнымалысының мәнін бір -бірлеп арттырамыз, содан кейін оны экранда cout << пәрмені арқылы көрсетеміз.
- Command cout << a ++; ұлғайту операторының постфикстік формасы бар, сондықтан жоғарыда сипатталған екінші ережені қолдана отырып, біз алдымен айнымалы мәнін (әлі де 9) cout << пәрменін пайдаланып экранға шығарамыз, содан кейін a айнымалысының мәнін бірге көбейтеміз.
- Келесі команда орындалғанда cout << a; өзгерген (жоғарылаған) мән көрсетіледі, яғни 10 саны.
- Сабақтың осы бөлімінің алдыңғы тақырыптарына сүйене отырып, біз енді x = x +1 немесе x = x - 1 сияқты ыңғайсыз және «ұсқынсыз» жазуды қалай жеңілдету керектігін білеміз, оны x ++ немесе x— айналдырамыз. Бірақ осылайша біз айнымалының мәнін бір ғана көбейтіп, төмендете аламыз, бірақ басқа сан туралы не деуге болады? Мысалы, жазуды қалай жеңілдетуге болады:

X=X+12;

Бұл жағдайда қарапайым шешім де бар - аралас операторлар деп аталатын немесе стенографиялық арифметикалық формаларды қолдану. Олар осылай көрінеді:

Пішін атауы	Комбинация	Стандартты жазу	Қысқартылған жазу
Көбейту арқылы тапсырма	*=	A=A*N	A*=N
Бөлімге тапсырма	/=	A=A/N	A/=N
Модульдік тапсырма	%=	A=A%N	A%=N
Азайтуға тапсырма	– =	A=A–N	A– =N
Қосымшаға тапсырма	+=	A=A+N	A+=N

Сізге қысқартылған формаларды әрі қарай қолдануды ұсынамыз, себебі бұл бағдарламалаудың жақсы тәжірибесі ғана емес, сонымен қатар бағдарлама кодының оқылуын айтарлықтай арттырады. Сонымен қатар, кейбір дереккөздерде қысқартылған формалар компьютермен тез өңделіп, бағдарламаның орындалу жылдамдығын арттырады. Жоғарыда айтылғандардың барлығына іс жүзінде көз жеткізетін уақыт келді, өйткені, олар айтқандай, жүз рет естігенше, бір рет көрген артық. Сіз жобаларды қалай құруға және оларға файлдарды қосуға болатынын білесіз, іс жүзінде дәл қазір сізден дәл осылай талап етіледі. Төменде арифметикалық амалдардың практикада қолданылуын көру үшін теру қажет бірнеше бағдарламалар берілген. Game деп аталатын жобадан бастайық.

6. Арифметикалық амалдарды қолдану.

№1 мысал. Ойын.

```
// примитивная игра для малышей #include <iostream>
using namespace std; void main()
{
int buddies; // количество пиратов до битвы
int afterBattle; //количество пиратов после битвы
// Вы пират. Сколько человек в вашей команде, если не считать вас?
cout<<"You the pirate. How many the person in your command, without you?\n\n"; cin>>buddies;
//Внезапно на вас нападает 10 мушкетеров
cout<<"Suddenly you are attacked by 10 musketeers \n\n";
//10 мушкетеров и 10 пиратов погибают в схватке. cout<<"10 musketeers and 10 pirates perish in fight.\n\n";
```

```
//подсчет оставшихся в живых afterBattle=1+buddies-10;
// Осталось лишь ... пиратов
cout<<»Remains only "<<afterBattle<<» pirates\n\n";

//Состояние убитых насчитывает 107 золотых монет cout<<"The condition killed totals 107 gold coins \n\n";
//Это по ... монет на каждого
cout<<"It on "<<(107/afterBattle)<<"coins on everyone";
//Пираты устраивают большую драку из-за оставшихся cout<<"Pirates arrange greater fight because of remained\n\n";
//... монет cout<<(107%afterBattle)<<"coins \n\n";
}
```

Бұл мысалда біртұтастықты бүтінге бөлу ережесі қолданылады - бұл бөлу кезінде бөлшек бөлігі, керек болса да, қысқартылады. Бұл туралы толығырақ сабақтың «Түрді түрлендіру» бөлімінде талқыланатын болады. Өрнекте (107 / afterBattle) - егер біз оларды тең бөлсек, әр қарақшы қанша тиын алатынын білеміз. Сонымен қатар, модуль операторы бізге қанша монетаның бөлінбейтінін анықтауға көмектеседі, яғни біз қалған 107 -ді қарақшылардың аман қалған санына бөлеміз. Бұл мысалдың барлық ерекшеліктері.

№2 мысал. Шеңбер.

Бұл мысал математикалық есептеулер жүргізетін бағдарламаларда арифметикалық операторлардың қолданылуын көрсетеді. Жобаның аты Circle.

Біз арифметикалық операторларды білу қарапайым есептерді шешуге мүмкіндік беретініне көз жеткіземіз. Алайда, операторларды қолдана білу жеткіліксіз, оларды қолданудың нәтижесі қандай болатынын түсіну қажет. Бұл келесі бөлімде талқыланады.

```
// программа для выяснения параметров окружности #include <iostream>
using namespace std; void main()
{
const float PI=3.141592;//обозначение константы - числа пи
//объявление переменных для хранения параметров float radius, circumference, area;
// приглашение ввести радиус
cout<<"Welcome to program of work with rounds\n\n"; cout<<"Put the radius from rounds\n\n"; cin>>radius;
cout<<"\n\n";
area=PI*radius*radius; // подсчет площади круга circumference=PI*(radius*2); // подсчет длины окружности
// вывод результатов
```

```
cout<<"Square of round: "<<area<<"\n\n"; cout<<"length of round: "<<circumference<<"\n\n"; cout<<"THANKS!!! BYE!!!\n\n";
}
```

7. Типтерді түрлендіру

Біз бірдеңе жасаған кезде, оның нәтижесі қандай болатынын білу маңызды. Мысалы, харчо сорпасы дайындалатын ингредиенттерден кілегей қосылған торт жасау екіталай. Сондықтан нәтиже құрамдас бөліктерге тікелей байланысты. Дәл осындай нәрсе айнымалылармен де болады. Егер, айталық, int түрінің екі саны қосылса, нәтиже int түрінде болатыны анық. Бірақ егер деректер әр түрлі болса ше? Бұл туралы осы сабақтың ағымдағы бөлімінде айтатын боламыз.

Сонымен, ең алдымен, мәліметтер типтері бір -бірімен қалай әрекеттесетінін түсінейік. Түр иерархиясы деп аталатын бар, онда барлық типтер басымдылық ретімен орналасады. Түр түрлендіруді түсіну үшін сіз әрқашан осы иерархияның түрлерінің ретін есте сақтауыңыз керек.

bool, char, short-int-unsigned int-
long-unsigned long-float-double-long double

Кейбір түрлердің өлшемдері бірдей болғанына қарамастан, олар әр түрлі мәндер диапазонын қабылдай алады, мысалы, белгісіз int, int -ден айырмашылығы, екі есе көп оң мәндерді қамтуы мүмкін, сондықтан иерархияда ескі, ал екі түрі де көлемі 4 байт. Сонымен қатар, мұнда көрсетілген өте маңызды ерекшелікті, егер түрді түрлендіруге bool, char, short сияқты түрлер қатысса, олар автоматты түрде int түріне түрленеді.

Енді трансформацияның әр түрлі классификациясын қарастырайық.

Құрамындағы мәндердің диапазоны бойынша жіктелуі. Барлық түрлендірулерді түрлендіруге қатысатын түрлердің иерархиясында орналасуына байланысты екі топқа бөлуге болады.

Тарылту түрлендіруі — мұндай түрлендірумен - иерархиядағы үлкен деректер түрі кіші түрге айналады, әрине, бұл жағдайда деректердің жоғалуы орын алуы мүмкін, сондықтан конверсияның тарылуына абай болу керек. Мысалға:

```
int A=23.5;
cout<<A; // на экране 23
```

Кеңейту түрлендіруі. Түрлендірудің бұл түрі деректер түрін кіші мәндер диапазонынан үлкен диапазонға дейін кеңейтуге әкеледі. Мысал ретінде келесі жағдай ұсынылады.

```
unsigned int a=3000000000;
```

cout<<a; // на экране 3000000000

Бұл жағдайда 3 000 000 000 - бұл int түрінің сөзбе -сөзі, ол unsigned int - ге қауіпсіз түрде кеңейтіледі, бұл бізге басқа ештеңені емес, экранда дәл 3,000,000,000 көруге мүмкіндік береді. Содан кейін, қарапайым int сияқты, мұндай сан сәйкес келмейді.

Түрлендіру әдісі бойынша жіктелуі.

Трансформацияның бағытына қарамастан, оны екі жолдың бірімен жасауға болады.

Жасырын түрлендіру. Жоғарыда келтірілген мысалдардың барлығы түрлендірудің осы түріне арналған. Бұл түрлендіруді автоматты деп те атайды, себебі ол бағдарламашының араласуынсыз автоматты түрде жүреді, басқаша айтқанда, біз оны жүзеге асыру үшін ештеңе істемейміз.

float A=23,5; - double стал float без
каких-либо дополнительных действий

Айқын түрлендіру. Бұл жағдайда түрлендіруді қажет болған кезде бағдарламашы жасайды. Мұның қарапайым мысалын қарастырайық:

double z=37.4;
float y=(int) z;
cout<<z<<”*** ”<<y; // на экране 37.4 ***37

(int) z - екі еселенген түрден int түріне дейін айқын тарылту бар. Сол жолда алынған int түрінен float түрге кеңейтілген жасырын түрлендіру бар. Есте сақтау керек, кез келген түрлендіру уақытша және тек ағымдағы желіде ғана жарамды. Яғни, z айнымалысы double еселенгендей, бүкіл бағдарламада қалады және оны int – ге түрлендіру уақытша болды.

Өрнектегі түрлендіруді теріңіз.

Енді біз ақырында сабақтың осы бөлімінің басында айтылған нәрсеге келдік, өрнектің нәтижесі қандай болатынын қалай білуге болады. Алынған білімді пайдалана отырып, мұны есептеуге тырысайық. Бізде келесі айнымалылар бар делік:

int I=27;
short S=2;
float F=22.3;
bool B=false;

Осы айнымалыларды қолдана отырып, біз осындай өрнек құрамыз:

I-F+S*B

Нәтижені деректер түрінің қандай айнымалысына жазуымыз керек? Егер сіз өрнекті деректер түрлері түрінде көрсетсеңіз, шешім қарапайым:

int-float+short*bool

Еске сала кетейік, short және bool бірден int болады, сондықтан өрнек келесідей болады:

int-float+int*int, при этом false станет 0

Int -ді int -ке көбейту int нәтижесін беретіні сөзсіз. Бірақ int -ке float қосу шығуда float береді, өйткені мұнда жаңа ереже пайда болады:

Егер кез келген өрнекте әр түрлі деректер түрлері қолданылса, нәтиже осы типтердің үлкеніне дейін азаяды.

Ақырында, int -тен float алып тастау, жоғарыда аталған ережеге сәйкес, қайтадан float береді.

Осылайша, өрнектің нәтижесі float түріне айналады.

float res= I-F+S*B; // 27-22.3+2*0

cout<<res; // на экране число 4.7

Енді сіз ережемен таныс болсаңыз, өрнекті егжей -тегжейлі талдаудың қажеті жоқ, тек ең үлкен түрін табу қажет, ол нәтиже болады.

Ескерту: Айнымалыларды бірдей деректер түрлерімен біріктіру кезінде де өте абай болыңыз. Мысалы, есіңізде болсын, егер бүтін бүтінге бөлінетін болса, онда сіз бүтін аласыз. Яғни int A = 3; int B = 2; cout << A / B; // 1 экранда, себебі нәтиже int және бөлшек бөлігі жоғалады. cout << (float) A / B; // экранда 1.5, себебі нәтиже float.

Түрді түрлендірудің мысалы.

Енді алған білімді практикада бекітейік. Жоба құрып, келесі кодты жазайық.

#include <iostream> using namespace std; void main(){

// объявление переменных и запрос на ввод данных float digit;

cout<<"Enter digit:"; cin>>digit;

/* Даже, если пользователь ввел число с вещественной частью,

результат выражения запишется в int и вещественная часть будет утеряна, разделив число на 100 мы получим количество сотен в нем. */

int res=digit/100;

cout<<res<<"hundred in you digit!!!\n\n";

}

Енді сіз мысалды қарадыңыз, сіз, әрине, түрді түрлендірудің көмегімен бір түрден екіншісіне уақытша ауысуды ұйымдастырып қана қоймай, қарапайым логикалық есепті де шешуге болатынын көрдіңіз. Сондықтан сіз бұл тақырыпқа мұқият қарауыңыз керек. Болашақта трансформацияны түсіну

сізге қызықты мәселелерді шешіп қана қоймай, қажетсіз қателіктерден аулақ болуға көмектеседі.

Бірыңғай инициализация

С ++ 11 -де әр түрлі бағдарлама конструкцияларына (айнымалылар, массивтер, объектілер) біркелкі түрде мән қоюға мүмкіндік беретін бірыңғай инициализациялау механизмі қосылды.

Айнымалыларды инициализациялау мысалын қарастырыңыз.

```
int a = {11}; // В a записывается значение 11 int b{33};
// В b записывается значение 33
```

Айнымалыларға мәндерді тағайындау үшін біз {} қолданамыз. Мысалдан көріп отырғаныңыздай, мұны екі жолмен жасауға болады. Инициализацияның бұл түрі тізім инициализациясы деп те аталады.

Тарылту және тізімді инициализациялау

Что произойдет при выполнении кода?

```
int x = 2.88;
cout<<x; // на экране отобразится 2
```

Бұл мысалда зерттелген материалдан белгілі болғандай, біз бүтін сан түріндегі x айнымалысына екі есе мән беретіндіктен, тарылтудың жасырын түрленуі орын алады. Алайда, егер сіз тізім инициализациясын қолдансаңыз, компилятор компиляция уақытында қате тудырады, себебі инициализацияның бұл түрі тарылудан қорғайды. Ол үлкен мәнді осындай мәндер ауқымын қолдамайтын түрге жазуға жол бермейді.

```
int x = { 2.88 }; // ошибка на этапе компиляции. 2.88 — double,
а x переменная целого типа
char ch = { 777 }; // ошибка на этапе компиляции. 777 — int,
а ch переменная символьного типа
// 777 не попадает в диапазон значений char
```

Басқа жақтан:

```
char ch2 = { 23 }; // всё правильно. 23 попадает в диапазон char
double x = { 333 }; // всё правильно 333 – int и попадает в диапазон double
```

Егер сіз компиляция кезінде деректердің жоғалуы мүмкін мәселелерді анықтағыңыз келсе, тізім инициализациясын қолдана аласыз.

Логикалық операциялар

Бағдарламалау кезінде көбінесе кейбір есептеулерді жүргізіп қана қоймай, мәндерді бір -бірімен салыстыру қажет. Ол үшін логикалық операциялар қолданылады. Логикалық операциялар әрқашан true немесе false, яғни шындыққа немесе жалғандыққа әкеледі. Логикалық операциялар үш кіші топқа бөлінеді:

1.Салыстыру операторлары

2.Теңдік операторлары

3.Бірлік пен теріс инверсияның логикалық операторлары.

Енді операторлардың әр тобын толығырақ қарастырайық.

Салыстыру операторлары.

Екі шаманың бір -бірімен қалай байланысты екенін білу қажет болғанда қолданылады.

Оператор белгісі	Мәлімдеме
<	Сол жақ операнд оң жақтан азырақ
>	Сол жақ операнд оң жақ операндтан үлкен
<=	Сол жақ операнд оң жақтан кіші немесе оған тең
>=	Сол жақ операнд оң жақ операндтан үлкен немесе оған тең

Салыстыру операцияларының мағынасы (екінші атауы - қатынас операциялары), егер оператор көрсеткен оператор ақиқат болса, онда ол қатысатын өрнек true мәнімен, егер дұрыс болмаса - false мәнімен ауыстырылады. Мысалға:

cout<<(5>3); // на экране будет единица, так как утверждение (5>3) истина.

cout<<(3<2); //на экране будет 0, так как (3<2) ложь.

Ескерту: False мен ақиқаттың орнына 0 мен 1 көрсетіледі, себебі олар жалған мен ақиқатқа тең. С тілінде ақиқаттың рөлі оң және теріс 1 мен 0 -ден басқа кез келген басқа сан болуы мүмкін.

Теңдік операторлары.

Екі мән толық сәйкес келетінін немесе сәйкес келмейтінін тексеру үшін қолданылады.

Бұл операторлардың қолданылуы алдыңғы топтың принципімен бірдей, яғни мәлімдемеге байланысты шығыс шын немесе жалғанмен ауыстырылады.

Оператор белгісі	Мәлімдеме
==	Сол жақ операнд оңға тең
!=	Сол жақ операнд оңға тең емес

Бұл операторлардың қолданылуы алдыңғы топтың принципімен бірдей, яғни мәлімдемеге байланысты шығыс шын немесе жалғанмен ауыстырылады.

cout<<(5!=3); // на экране будет единица,

так как утверждение (5!=3) истина.

cout<<(3==2); //на экране будет 0, так как (3==2) ложь.

Логикалық операциялары және теріс инверсия.

Көп жағдайда бір ғана мәлімдемен шығу мүмкін емес. Көбінесе мәлімдемелерді бір жолмен біріктіру қажет. Мысалы, санның 1 -ден 10 -ға дейінгі диапазонда екенін тексеру үшін екі операторды тексеру қажет: сан бір мезгілде> = 1 және <= 10 болуы керек. Мұндай комбинацияны іске асыру үшін қосымша енгізу қажет. Операторлар

Операция	Аты
&&	ЖӘНЕ
||	НЕМЕСЕ
!=	ЕМЕС

Логикалық ЖӘНЕ (&&)

Логикалық ЖӘНЕ екі мәлімдемені біріктіреді және сол және оң мәлімдеме ақиқат болғанда ғана ақиқатты қайтарады. Егер мәлімдемелердің біреуі немесе екеуі де жалған болса, біріктірілген өрнек жалғанмен ауыстырылады. Логикалық ЖӘНЕ қысқартылған схема бойынша жұмыс істейді, яғни егер бірінші оператор жалған болса, енді екіншісі тексерілмейді.

Мәлімдеме 1	Мәлімдеме 2	Мәлімдеме 1&& Мәлімдеме 2
true	true	true
true	false	false
false	true	false
false	false	false

Енді бағдарламада санды алатын және бұл санның 1 -ден 10 -ға дейінгі диапазонға жататынын анықтайтын мысалды қарастырайық.

#include <iostream> using namespace std; void main()

{

int N;

cout<<"Enter digit:\n"; cin>>N; cout<<((N>=1)&&(N<=10));

cout<<"\n\nIf you see 1 your digit is in diapazone\n\n"; cout<<"\n\nIf you see 0 your digit is not in diapazone\n\n";

}

Бұл мысалда, егер екі тұжырым да рас болса, 1 өрнектің орнына қойылады, әйтпесе - 0. Тиісінше, қолданушы бағдарламалық нұсқауларды қолдана отырып жағдайды талдай алады.

Логикалық НЕМЕСЕ (||)

Логикалық НЕМЕСЕ екі мәлімдемені біріктіреді

және егер мәлімдемелердің кем дегенде біреуі дұрыс болса, ақиқатты қайтарады, ал егер екі мәлімдеме де дұрыс болмаса. Логикалық НЕМЕСЕ қысқартылған схема бойынша жұмыс істейді, яғни егер бірінші мәлімдеме рас болса, енді екіншісі тексерілмейді.

Мәлімдеме 1	Мәлімдеме 2	Мәлімдеме 1&& Мәлімдеме 2
true	true	true
true	false	true
false	true	true
false	false	false

Бағдарлама нөмірді алатын және бұл санның 1 -ден 10 -ға дейінгі диапазонға кіретінін анықтайтын мысалды қайта қарастырайық. Тек біз қазір НЕМЕСЕ қолданамыз.

```
#include <iostream> using namespace std; void main()
{
int N;
cout<<"Enter digit:\n"; cin>>N; cout<<((N<1)||(N>10));
cout<<"\n\nIf you see 0 your digit is in diapazone\n\n"; cout<<"\n\nIf you see 1 your digit is not in diapazone\n\n";
}
```

Бұл мысалда, егер екі мәлімдеме де жалған болса (яғни, сан 1 -ден кем емес және 10 -нан аспайды), 0 өрнектің орнына қойылады, әйтпесе - 1. Тиісінше, қолданушы, алдыңғы мысалдағыдай , жағдайды талдай алады және нәтиже шығара алады.

Логикалық ЕМЕС (!)

Логикалық ЕМЕС - бұл біртұтас оператор, сондықтан оны одақ операторы деп атауға болмайды. Ол бекітуді тексеру нәтижесін қайтарғыңыз келгенде қолданылады.

Мәлімдеме	!Мәлімдеме
true	false
false	true

```
// на экране будет 1, так как (5==3) ложь и её инверсия это - истина.
cout<<!(5==3);
```

//на экране будет 0, так как (3!=2) истина и её инверсия это - ложь. cout<<!(3!=2);

Логикалық терістеу жалған мәлімдеменің орнына қайтып оралады, егер соңғысы дұрыс болса және керісінше, егер жалған болса. Бұл оператор шартты қысқарту үшін қолданылуы мүмкін. Мысалы, өрнек:

```
b==0
```

инверсия көмегімен қысқартуға болады:

```
!b
```

егер b нөлге тең болса, екі жазба да нәтиже береді.

If логикалық таңдау конструкциясы

Енді біз қарапайым сызықтық бағдарламаны «ойлау» бағдарламасына айналдыруға мүмкіндік беретін оператормен танысамыз. Бұл оператор қандай да бір мәлімдемені (өрнекті) шындыққа тексеруге және алынған нәтижеге байланысты осы немесе басқа әрекетті орындауға мүмкіндік береді. Алдымен осы операторды жалпы синтаксисін қарастырайық:

```
if (утверждение или выражение)
{
действие 1;
else
{
действие 2;
}
```

If операторының негізгі принциптері.

1. Логикалық операторлары немесе арифметикалық өрнегі бар кез келген құрылым мәлімдеме немесе өрнек бола алады.

- if (X> Y) - егер X шынымен Y -ден үлкен болса, әдеттегі тұжырым дұрыс болады

```
int X=10,Y=5;
if(X>Y){ // истина cout<<"Test!!!";// на экране Test
}
```

- if (A> B&& A <C) - егер екі бөлік те ақиқат болса, екі бөліктен тұратын біріктірілген мәлімдеме ақиқат болады

```
int A=10,B=5,C=12;
if(A>B&&A<C){ // истина
cout<<"A between B and C";// на экране A between B and C
```

```
}
```

- егер (A - B) - арифметикалық өрнек, егер A B -ға тең болмаса, дұрыс болады, себебі басқаша (егер олар тең болса) олардың айырмасы нөлге тең болады, ал нөл - жалған
-

```
int A=10,B=15;
if(A-B){ // -5 это истина
cout<<"A != B";// на экране A != B
}
```

- if (++ A) - арифметикалық өрнек, егер A –1 -ге тең болмаса, дұрыс болады, себебі A –1 -ге тең болса, 1 -ге көбейту нөл береді, ал нөл - жалған

```
int A=0;
if(++A){ // 1 это истина
cout<<"Best test!!";// на экране Best test!!
}
```

- if (A ++) - арифметикалық өрнек, егер A 0 -ге тең болмаса, дұрыс болады, себебі бұл жағдайда постфикстің өсу формасы қолданылады, шарт бірінші тексеріледі және нөл табылады, содан кейін бірге көбейтіледі.

```
int A=0;
if(A++){ // 0 это ложь
cout<<"Best test!!";// эту фразу мы не увидим, т. к. if не выполнится
}
```

- f (A == Z) - әдеттегі тұжырым, егер A Z -ге тең болса, ол дұрыс болады
- if (A = Z) - тағайындау әрекеті, егер Z нөлге тең болмаса, өрнек ақиқат болады

Ескерту: Әдеттегі қателік. Көбінесе, теңдікті тексеру операциясының орнына ==, тағайындау операциясы = кездейсоқ түрде көрсетіледі және өрнектің мағынасы түбегейлі өзгеруі мүмкін. Мұндай қарапайым қате бүкіл бағдарламаның дұрыс жұмыс істемеуіне әкелуі мүмкін. Бірдей көрінетін екі мысалды қарастырайық.

Дұрыс мысал.

```
#include <iostream> using namespace std; void main(){
int A,B; //объявляем две переменные
//просим пользователя ввести в них данные cout<<"Enter first digit:\n";
cin>>A;
cout<<"Enter second digit:\n"; cin>>B;
if(B==0){ // если B содержит ноль
```

```
cout<<"You can’t divide by zero!!!";// сообщаем об ошибке
}
else{ // в противном случае
cout<<"Result A/B="<<A<<"/"<<B<<"="<<A/B;// выдаем результат деления A на B
}
cout<<"\n The end. \n"; // конец
}
```

Қате мысалы.

```
#include <iostream> using namespace std; void main(){
int A,B; //объявляем две переменные
//просим пользователя ввести в них данные cout<<"Enter first digit:\n";
cin>>A;
cout<<"Enter second digit:\n"; cin>>B;
// Приравниваем B к нулю и проверяем условие, оно автоматически ложно if(B=0){ // эта часть не выполнится никогда, т. к. условие всегда ложно
cout<<"You can’t divide by zero!!!";
// сообщаем об ошибке
}
else{ // всегда выполняется эта часть, в которой A делится на новоиспеченный нуль
/* В этой строке произойдет
ошибка на этапе выполнения, т. к. компьютер попытается разделить число на нуль */ cout<<"Result A/B="<<A<<"/"<<B<<"="<<A/B;
}
cout<<"\n The end. \n"; // Эту фразу мы не увидим никогда.
}
```

1. Сіз байқағандай, егер жақшаның мазмұны шын болса, онда if құрылымының бұйра жақшаларымен қоршалған 1 -әрекет орындалады, ал басқа блоктың 2 -әрекеті еленбейді.

2. Егер жақшаның мазмұны жалған болса, else операторының бұйра жақшасына алынған 2 -әрекет орындалады және 1 -әрекет еленбейді.

3. Басқа сөйлем міндетті емес. Бұл дегеніміз, егер мәлімдеме жалған болса, бірдеңе істеудің қажеті болмаса, бұл құрылысты елемеуге болады. Мысалы, нөлге бөлуден қорғаныстық қолданатын бағдарламаны былай жазуға болады:

```
#include <iostream> using namespace std; void main(){
int A,B; //объявляем две переменные
//просим пользователя ввести в них данные cout<<"Enter first digit:\n";
cin>>A;
```

```
cout<<"Enter second digit:\n"; cin>>B;
if(B!=0){ // если B не равно нулю
cout<<"Result A/B="<<A<<"/"<<B<<"="<<A/B; // производим вычисления
}
// в противном случае не делаем ничего cout<<"\nThe end.\n";
}
```

4. Егер if немесе else блогына бір ғана команда қатысты болса, онда бұйра жақшаларды алып тастауға болады. Осы ережені қолдана отырып, біз бағдарламаны одан да қысқартамыз:

5.

```
#include <iostream> using namespace std; void main(){
int A,B; //объявляем две переменные
//просим пользователя ввести в них данные cout<<"Enter first digit:\n";
cin>>A;
cout<<"Enter second digit:\n"; cin>>B;
if(B!=0) // если B не равно нулю
cout<<"Result A/B="<<A<<"/"<<B<<"="<<A/B; // производим вычисления
// в противном случае не делаем ничего cout<<"\nThe end.\n";
}
```

Біз if операторын көрдік және оның жұмысының негізгі принциптерін талқыладық. Егер нақты практикалық мысалдардың ерекшеліктерін қарастыруға көшпес бұрын, кішкене шегініс жасап, қарапайым шартты қоюға болатын басқа операторды қарастырайық.

Ескерту: Сақ болыңыз: if операторы мен else операторы ажырамас! Олардың арасында код жолын енгізуге әрекет жасау компиляция уақытында қатеге әкеледі.

Қате бар код фрагменті.

```
....
if(B==0){ // если B содержит ноль
cout<<"You can’t divide by zero!!!";// сообщаем об ошибке
}
cout<<"Hello";// Ошибка!!!! Разрыв конструкции if - else!!! else{ // в противном случае
cout<<"Result = "<<A/B;// выдаем результат деления A на B
}
....
```

Үштік оператор.

Кейбір шарттар өте қарабайыр. Мысалы, екі санды бөлу бағдарламасын алайық. Бұл әрекет тұрғысынан да, код тұрғысынан да қарапайым. If және else операторларының әрқайсысында бір код жолы бар - әрекеттер. Мұндай бағдарламаны үштік оператордың көмегімен одан әрі жеңілдетуге болады.

Алдымен оның синтаксисін қарастырайық:

УТВЕРЖДЕНИЕ ИЛИ ВЫРАЖЕНИЕ?ДЕЙСТВИЕ1:ДЕЙСТВИЕ2;

Жұмыс принципі қарапайым - егер МӘЛІМДЕМЕ НЕМЕСЕ ӨРНЕК шын болса, 1 -ӘРЕКЕТ орындалады, егер ол жалған болса, 2 -ӘРЕКЕТ орындалады.

Мысал арқылы осы оператордың әрекетін қарастырайық:

```
#include <iostream> using namespace std; void main(){
int A,B; //объявляем две переменные
//просим пользователя ввести в них данные cout<<"Enter first digit:\n";
cin>>A;
cout<<"Enter second digit:\n" cin>>B;
/* В данном случае, если В не будет равно нулю, выполниться та команда, которая стоит после знака вопроса и на экране покажется результат деления. В
противном случае выполниться команда стоящая после знака двоеточие и на экране будет сообщение об ошибке деления на нуль.*/
(B!=0)?cout<<"Result A/B="<<A<<"/!"<<B<<"="<<A/B:cout<<"You can’t divide by zero!!!";
//конец программы cout<<"\n The end. \n";
}
```

Код одан да жақсы емес пе?! Алынған ақпаратты бекіту үшін біз тағы бір күрделі мысал келтіреміз. Бағдарлама пайдаланушы енгізген екі санның қайсысы үлкен, қайсысы кіші екенін анықтайды.

```
#include <iostream> using namespace std; void main(){
int a,b; //объявляем две переменные
//просим пользователя ввести в них данные cout<<"Enter first digit:\n";
cin>>a;
cout<<"Enter second digit:\n"; cin>>b;
/*Если, (b>a), то на место оператора ?: подставится b, в противном случае на место оператора подставится a, таким образом, то число, которое больше запишется
в переменную max.*/ int max=(b>a)?b:a;
/*Если, (b<a), то на место оператора ?: подставится b, в противном случае на место оператора подставится a, таким образом, то число, которое больше запишется
```

```
в переменную min.*/ int min=(b<a)?b:a;
// Вывод результата на экран. cout<<"\n Maximum is \n"<<max;
cout<<"\n Minimum is \n"<<min<<"\n";
}
```

Сонымен, келесіні нақты түсінейік: егер шарт пен оған байланысты әрекеттер жеткілікті қарапайым болса, онда біз үштік операторды қолданамыз. Егер бізге күрделі құрылыс қажет болса, онда, әрине, if операторын қолданамыз.

Баспалдақ if – else if

Оқулықтың соңғы бөлімінде сіз шартты операторлардың болуы туралы білдіңіз. Енді олардың жұмысының ерекшеліктері туралы ақпарат алған жақсы болар еді.

Бізге сомаға байланысты ақшалай жеңілдіктерді есепке алу үшін бағдарлама жазу керек делік. Мысалы, егер тұтынушы 100 грннан жоғары тауар сатып алса, оған 5% жеңілдік беріледі. 500 грн астам. - 10%, соңында 1000 грн. - 25%. өтінім сатып алушы жеңілдік алған жағдайда төлеуге тиісті соманы көрсетуі керек. Енді сіз мәселенің ең жақсы шешімін табуыңыз керек. Жобаның атауы **Discount.**

№1 шешім нұсқасы.

```
#include <iostream> using namespace std; void main(){
// объявляется переменная,для хранения первоначальной суммы int
summa;
// запрос на ввод суммы с клавиатуры cout<<"Enter item of summa:\n";
cin>>summa;
if(summa>100){ // если сумма больше 100 грн., скидка 5% cout<<"You
have 5% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*5<<"\n";
}
if(summa>500){ // если сумма больше 500 грн., скидка 10% cout<<"You
have 10% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*10<<"\n";
}
if(summa>1000){ // если сумма больше 1000 грн., скидка 25%
cout<<"You have 25% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*25<<"\n";
}
else{ // в противном случае, скидки нет cout<<"You have not
discount!!!\n"; cout<<"You must pay - "<<summa<<"\n";
}
}
```

Бұл мысал, бір қарағанда, бастаушы бағдарламашыға ешқандай шағым тудырмайды, алайда, бағдарлама өте дұрыс жұмыс жасамайтын жағдайды қарастырайық. Пернетақтада енгізілген сома 5000. Бұл сан 1000 -нан асады, сондықтан біз 25% жеңілдік алуымыз керек. Алайда мүлде басқа нәрсе болады.

1. If if операторының әрқайсысы тәуелсіз және басқа ifs тәуелді емес, сондықтан, егер орындалса да, шарт барлық мәлімдемелер үшін тексеріледі.

2. Алдымен, егер (summa> 100) орындалатындығын тексеру. 5000, әрине 100 -ден көп, шарт дұрыс және if денесі орындалады. Экранда біз аламыз:

You have 5% discount!!!
You must pay - 4750

3. Алайда, бағдарлама мұнымен тоқтап қалмайды - if шарты (summa> 500) талданатын болса. 5000 500 -ден үлкен болса, шарт қайтадан орындалады және if денесі орындалады. Экранда біз аламыз:

You have 10% discount!!!
You must pay - 4500

4. Ақырында, бағдарлама if шартын тексереді (summa > 1000), бұл да дұрыс болады, өйткені 5000 1000 -нан үлкен. Және if -мен байланысты әрекет қазір орындалады. Экран көрсетіледі:

You have 25% discount!!! Y
ou must pay - 3750

Осылайша, бір ақпараттық белгі орнына біз үш аламыз. Мәселенің мұндай шешімі тиімсіз. Оны оңтайландыруға тырысайық. Жобаның атауы **Discount2.**

№2 шешім нұсқасы.

```
#include <iostream> using namespace std; void main(){
// объявляется переменная,для хранения первоначальной суммы int summa;
// запрос на ввод суммы с клавиатуры cout<<"Enter item of summa:\n"; cin>>summa;
// если сумма в диапазоне от 100 грн. до 500 грн.,
скидка 5%
if(summa>100&&summa<=500){
cout<<"You have 5% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*5<<"\n";
}
```

```
// если сумма в диапазоне от 500 грн. до 1000 грн., скидка 5%
if(summa>500&&summa<=1000)
{
cout<<"You have 10% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*10<<"\n";
}
if(summa>1000){ // если сумма больше 1000 грн., скидка 25%
cout<<"You have 25% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*25<<"\n";
}
else{ // в противном случае, скидки нет cout<<"You have not
discount!!!\n"; cout<<"You must pay - "<<summa<<"\n";
}
}
```

Алдымен, қолданушы 5000 грн сомасын енгізді деп тағы елестетейік.

1. Алдымен шарт тексеріледі, if (summa> 100 && summa <= 500). 5000 көрсетілген диапазонға кірмейді, шарт жалған және if денесі орындалмайды.

2. Әрі қарай, if (summa> 500 && summa <= 1000) шарты талданады. 5000 бұл диапазонға кірмейді, шарт қайтадан жалған болады және егер дене орындалмаса.

3. Ақырында, бағдарлама if шартын тексереді (summa > 1000), бұл шындыққа сәйкес келеді, себебі 5000 1000 -нан үлкен. Және if -мен байланысты әрекет орындалады. Экранда көрсетіледі:

```
You have 25% discount!!!
You must pay - 3750
```

Бұған тоқталуға болатын сияқты, бірақ тағы бір нұсқаны тексерейік. Мысалы, пайдаланушы 600 мәнін енгізеді. Және келесі деректер экранда пайда болады:

```
Enter item of summa:
600
You have 10% discount!!! You must pay - 540
You have not discount!!! You must pay - 600
Press any key to continue
```

Оқиғалардың бұл түрін оңай түсіндіруге болады:

1. Біріншіден, if (summa> 100 && summa <= 500) шарты тексерілетін болса. 5000 көрсетілген диапазонға кірмейді, шарт жалған және if денесі орындалмайды.

2. Әрі қарай, if (summa> 500 && summa <= 1000) шарты талданады. Бұл диапазонға 5000 қосылады, шарт дұрыс және if орындалатын болса, экранда 10% жеңілдік туралы хабарлама шығады.

3. Ақырында, бағдарлама if (summa> 1000) шартын тексереді, ол жалған болып шығады. If -пен байланысты әрекет орындалмайды, бірақ бұл тәуелсіз if операторының өзгесі бар, ол біздің жағдайда жұмыс істейді. Экранда жеңілдіктер туралы хабарлама көрсетіледі.

Қорытынды: біріншіден, біз else операторының соңғы if -ге ғана қатысты екенін білдік. Екіншіден, біз бұл бағдарламаның орындалуы бізге де сәйкес келмейді деген қорытындыға келдік. Шешімнің басқа мысалын қарастырайық. Жобаның атауы **Discount3.**

№3 шешім нұсқасы.

```
#include<iostream> using namespace std; void main(){
// объявляется переменная,для хранения первоначальной суммы int summa;
// запрос на ввод суммы с клавиатуры cout<<"Enter item of summa:\n"; cin>>summa;
if(summa>1000){ // если сумма больше 1000 грн., скидка 25% cout<<"You have 25% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*25<<"\n";
}
else{ // если сумма не больше 1000 грн. продолжаем анализ if(summa>500){ // если сумма больше 500 грн., скидка 10%
cout<<"You have 10% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*10<<"\n";
}
else{ // если сумма не больше 500 грн. продолжаем анализ if(summa>100){ // если сумма больше 100 грн., скидка 5%
cout<<"You have 5% discount!!!\n";
cout<<"You must pay - "<<summa-summa/100*5<<"\n";
}
else{ // если сумма не больше 100 грн. скидки нет cout<<"You have not discount!!!\n";
cout<<"You must pay - "<<summa<<"\n";
}
}
}
}
```

Бұл мысалды мұқият талдай отырып, сіз келесі әрекетті орындауға болатынын байқайсыз, егер ол тек «предшественник» орындалмаған болса, себебі ол соңғысының else құрылымында. Осылайша біз оңтайлы енгізу кодын таптық. Біз жасаған құрылымды «Лесенка if else if» деп атайды,

өйткені ондағы жағдай баспалдақ тәрізді реттелген. Енді бұл құрылыстың пайдалы екенін бәріміз білеміз. Соңғы әсер қалады:

Кодты оптимизациялау.

Сабақтың алдыңғы бөлімінде ереже болды: егер if немесе else блогына бір ғана команда қатысты болса, онда бұйра жақшаларды алып тастауға болады. Мәселе мынада, if else құрылымы біртұтас командалық құрылым болып саналады. Сондықтан, егер біреудің ішінде кірістірілген конструкциядан басқа ештеңе болмаса, басқалардың бұйра жақшаларын алып тастауға болады:

```
#include <iostream>usingnamespacestd;voidmain(){
//объявляетсяпеременная,дляхраненияпервоначальнойсуммыintsumma;
//запроснавводсуммысклавиатурысout<<"Enter item of
summa:\n";cin>>summa;
if(summa>1000){//еслисуммабольше1000грн.,скидка25%cout<<"Youhav
e25%discount!!!\n";
cout<<"Youmustpay-"<<summa-summa/100*25<<"\n";
}
//еслисуммanебольше1000грн.продолжаеманализ
elseif(summa>500){//еслисуммабольше500грн.,скидка10%cout<<"Youha
ve10%discount!!!\n";
cout<<"Youmustpay-"<<summa-summa/100*10<<"\n";
}
//еслисуммanебольше500грн.продолжаеманализ
elseif(summa>100){//еслисуммабольше100грн.,скидка5%
cout<<"Youhave5%discount!!!\n";
cout<<"Youmustpay-"<<summa-summa/100*5<<"\n";
}
else{//еслисуммanебольше100грн.скидкинет
cout<<"You have not discount!!!\n";cout<<"Youmustpay-"<<summa<<"\n";
}
```

Практикалық мысал: мәтіндік квест құру

Тапсырма тұжырымы

Сізге квест сияқты ойындар жанры белгілі. Мұндай ойынның кейіпкері әр түрлі тапсырмаларды орындауы, сұрақтарға жауап беруі, ойынның нәтижесі тәуелді болатын шешімдер қабылдауы тиіс. Біз енді мәтіндік квест (графикасыз квест) құруға тырысамыз. Біздің міндет - кейіпкерге әрекеттің нұсқаларын ұсыну және оның таңдауына байланысты жағдайды құру. Жобаныңаты - Quest.

Реализация коды.

```
#include <iostream>usingnamespacestd;voidmain()
{
```

```
        //Добропожаловать.Трииспытаниячести.Злоймагпохитил
        //принцессуиеесудьбавтвоихруках.Онпредлагаеттебе
        //пройти3испытаниячестивеголабиринте.
        cout<<"Welcome.Threetestsofhonour.Themaliciousmagicianhasstolen\n\n";
cout<<"\nprincessanditsdestinyinyourhands.Itsuggestsyou\n";
cout<<"\ntopass3testsofhonourinitslabyrinth.\n";
        boolgoldTaken,diamondsTaken,killByDragon;
        //Тывходишьвпервуюкомнату,здесьоченьмногозолота.
        cout<<"Youenterintothefirstroom,hereitisalotofgold.\n\n";
        //Возьмешьлитыего?
        cout<<"Whetheryouwilltakeit?(1-yes,0-no)\n\n";cin>>goldTaken;
        if(goldTaken)//есливозьмешь
        {
        cout<<"Goldremainstoyou,butyouhaveruinedtest.GAMEisover!!!\n\n";
        }
        else//еслинет
        {
        //Поздравляю,тыпрошелпервоеиспытаниечести!
        cout<<"Icongratulate,youhavepassedthefirsttestabuse!\n\n";
        //Тыпереходишь в следующуюкомнату.Онаполнабриллиантов
cout<<"Youpassinafollowingroom.Itisfullofbrilliants\n\n";
        //Возьмешьлитыбриллианты?
        cout<<"Whetheryouwilltakebrilliants?(1-yes,0-
no)\n\n";cin>>diamondsTaken;
        if(diamondsTaken)//есливозьмешь
        {
        //Бриллиантыостаютсятебе,нотыпровалилвтороеиспытаниеcout<<"Brilli
antsremaintoyou,butyouhaveruinedthesecondtest\n\n";
        //ИГРАОКОНЧЕНА!!!
        cout<<"GAMEisover!!!\n\n";
        }
        else//еслинет
        {
        //Поздравляю,тыпрошелвтороеиспытаниечести!!!
        cout<<"Icongratulate,youhavepassedthesecondtestabuse!!!\n\n";
        //Тывходишьвтретьюкомнату.
        cout<<"Youenterintothethirdroom.\n\n";
        //Накрестьянинанапалдракон!Двигатьсядальше
        cout<<"Thepersonwasattackedbyadragon!Tomovefurther\n\n";
        //необращаянанихвнимания
        cout<<"Notpayingtothemofattention(1-yes,0-no)?\n\n";cin>>killByDragon;
        if(killByDragon)//есливозьмешь
        {
```

```
//Тыпытаешьсяпроскользнутьмимо,
нодраконcout<<"Youtrytopasspast,butadragon\n\n";
//замечаеттвоеприсутствие.cout<<"noticesyourpresence\n\n";
//Онпревращаеттебявпепел.Тымертв!!!
cout<<"Ittransformsyouintoashes.Youaredead!!!\n\n";
//ИГРАОКОНЧЕНА!!!
cout<<"GAMEisover!!!\n\n";
}
else//еслинет
{
//Поздравляю,тысчестьюпрошелвсеиспытания!!!
cout<<"Icongratulate,youwithhonourhavewastestedall!!!\n\n";
//Принцесса достается тебе!!!cout<<"Princessgetstoyou!!!\n\n";
}
}
}
```

Бірнеше таңдау жүйесі

Switch

Біз шарттарды талдайтын құрылым мен - if конструкциясы, сондай –ақ үшті коператормен таныспыз. Тағы бір таңдау мәлімдемесі - switch операторы. Сіз бес элементтен тұратын мәзірді қолданатын бағдарлама жазғыңыз келетінін елестетіп көріңіз. Мысалы, кішкентай балаларға арналған қосуға, азайтуға және т.б. қосуға болатын шағын бағдарлама. Таңдауды өңдеуді if else сатысын қолдана аласыз, мысалы:

```
#include<iostream>usingnamespacestd;voidmain(){
//объявлениепеременныхивводзначениясклавиатурыfloatA,B,RES;
cout<<"Enterfirstdigit:\n\";cin>>A;
cout<<"Enterseconddigit:\n":;cin>>B;
//реализацияпрограммногоменюcharkey;
cout<<"\nSelectoperator:\n»;
cout<<"\n + - ifyouwanttoseeSUM.\n";cout<<"\n--
ifyouwanttoseeDIFFERENCE.\n";cout<<"\n*-
ifyouwanttoseePRODUCT.\n0";cout<<"\n/-ifyouwanttoseeQUOTIENT.\n";
//ожиданиевыборапользователяcin>>key;
if(key==’+’){//еслипользовательвыбралсложение
. RES=A+B;
.       cout<<"\nAnswer:"<<RES<<"\n";
}
elseif(key==’-’){//еслипользовательвыбралвычитание
RES=A-B;
cout<<"\nAnswer:"<<RES<<"\n";
}
```

```
elseif(key==’*’){//еслипользовательвыбралумножение
.       RES=A*B;
.       cout<<"\nAnswer:"<<RES<<"\n";
}
RES=A/B;
cout<<"\nAnswer:"<<RES<<"\n";
else{ // если делитель равен нулюcout<<"\nError!!!Dividebynull!!!!\n";
}
else{//если введенный символне корректен
.       cout<<"\nError!!!Thisoperatorisn’tcorrect\n";
}
```

Жоғарыда келтірілген мысал өте дұрыс, бірақ біршама қиын көрінеді. Бұл кодты айтарлықтай жеңілдетуге болады, ол үшінswitch қолданылады. Ол айнымалының мәнін тұтас мәндер сериясымен салыстыруға және сәйкестікті кездестіргенде белгілі бір әрекетті орындауға мүмкіндік береді.

Жалпы синтаксис және жұмыс принципі.

Алдымен оператордың жалпы синтаксисін қарастырайық:

```
switch(выражение){casезначение1:
действие1;
break;casезначение2:
действие2;
break;casезначение3:
действие3;
break;
.....
default:действие_по_умолчанию;
break;
}
```

Осы орналастыру формасын талдайық:

Өрнек - сәйкестігі тексерілуі қажет мәліметтер. Мұнда айнымалы мәнді көрсетуге болады (бірақ тек char немесе бүтін сан түрінде) немесе нәтижесі бүтін деректер болып табылатын өрнек.

case Значение1, case значение2, case значение3 — өрнек тексерілетін бүтін немесе символдық тұрақты мәндер.

Действие1,действие2,действие3—Егер өрнектің мәні case мәніне сәйкес келсе, орындалатын әрекеттер.

Егер сәйкестік орын алса және сәйкестендірілген жағдайға байланысты әрекет сәтті орындалса, switch жұмысын тоқтатады және бағдарлама қосқыш операторының бұйра жақшасы жабылғаннан кейін бағдарлама келесі жолға өтеді. Break операторы бұл функцияға жауап береді; ол коммутатордың орындалуын тоқтатады.

Егер сәйкестікті талдау кезінде default бөлім іске қосылады және default_action орындалады. Default оператор else операторға ұқсас.

Енді тақырыптың басында келтірілген мысалды қалай жеңілдетуге болатынын қарастырайық.

Мысалды оптимизациялау.

```
#include<iostream>usingnamespacestd;voidmain(){
//объявлениепеременныхивводзначениясклавиатурыfloatA,B,RES;
cout<<"Enterfirstdigit:\n";cin>>A;
cout<<"Enterseconddigit:\n";cin>>B;
//реализацияпрограммногоменюcharkey;
cout<<"\nSelectoperator:\n";
cout<<"\n + - ifyouwanttoseeSUM.\n";cout<<"\n--
ifyouwanttoseeDIFFERENCE.\n";cout<<"\n*-
ifyouwanttoseePRODUCT.\n";cout<<"\n/-ifyouwanttoseeQUOTIENT.\n";
//ожиданиевыборапользователяcin>>key;
//проверяетсязначениепеременнойkeyswitch(key){
case‘+’:  //еслипользовательвыбралсложениеRES=A+B;
cout<<"\nAnswer: "<<RES<<"\n";break;//остановкаswitch
case‘-’:  //еслипользовательвыбралвычитаниеRES=A-B;
cout<<"\nAnswer: "<<RES<<"\n";break;//остановкаswitch
case‘*’:  //еслипользовательвыбралумножениеRES=A*B;
cout<<"\nAnswer:"<<RES<<"\n";
break;//остановкаswitchcase‘/’:
// если пользователь выбрал делениеif(B){//еслиделительнеравеннулю
RES=A-B;
cout<<"\nAnswer:"<<RES<<"\n";
}
else{ // если делитель равен нулюcout<<"\nError!!!Dividebynull!!!!\n";
}
break;//остановкаswitch
default:  // если введенный символ
некорректенcout<<"\nError!!!Thisoperatorisn’tcorrect\n";break;//остановкаswitc
h
}
}
```

Көріп отырғаныңыздай, код қазір әлде қайда қарапайым және оқуға оңай болып көрінеді.

Switch операторын қолдану өт қарапайым, бірақ сіз оның жұмысының кейбір ерекшеліктерін білуіңіз керек:

1.Егер case таңбалық мәндерді қолданса, олар бір тырнақшаға алынады, егер олар бүтін сандар болса, онда тырнақшасыз.

2. Default операторы switch жүйесінің кез келген жерінде орналасуы мүмкін, егер сәйкестік болмаса, ол әліде орындалады. Дегенмен, бүкіл құрылыстың соңында default мәнді қолдану жақсы тәжірибе.

```
switch(выражение){casезначение1:
действие1;break;
casезначение2:
действие2;break;
default:
действие_по_умолчанию;break;
casезначение3:
действие3;break;
}
```

3. Тізімдегі ең соңғы операторлан кейін (caseнемесе default болсын) break операторын алып тастауға болады.

```
switch(выражение){casезначение1:
действие1;break;
casезначение2:
действие2;break;
default:
действие_по_умолчанию;break;
casезначение3:
действие3;
}
switch(выражение){casезначение1:
действие1;break;
casезначение2:
действие2;break;
casезначение3:
действие3;break;
default:
действие_по_умолчанию;
}
```

4. Defaultоператорды мүлде алып тастауға болады, егер сәйкестік табылмаса, ештеңе болмайды.

```
switch(выражение){casезначение1:
действие1;
break;casезначение2:
действие2;
break;casезначение3:
```

```
действие3;
break;
}
```

5. Егер тексерілген өрнектің әр түрлі мәндері үшін бірдей әрекеттер жиынтығын орындау қажет болса, қатарға бірнеше белгілерді жазуға болады. Әріпті бағалау жүйесін цифрлық жүйеге түрлендіретін бағдарламаның мысалын қарастырайық.

```
#include<iostream>usingnamespacestd;voidmain(){
//объявлениепеременной,дляхраненияbуквеннойоценкиcharcRate;
//просьбаввестибуквеннуюоценкусout<<"Inputyourchar-
rate\n";cin>>cRate;
//анализведенногозначенияswitch(cRate){
case‘A’:
case‘a’:
//оценкаАилиаравноценна5cout<<"Your rate is 5\n";break;
case‘B’:
case‘b’:
//оценкаВилиbравноценна4cout<<"Your rate is 4\n";break;
case‘C’:
case‘c’:
//оценкаСилисравноценна3cout<<"Your rate is 3\n";break;
case‘D’:
case‘d’:
//оценкаDилиdравноценна2cout<<"Your rate is 2\n";break;
default:
// остальныесимволынекорректныcout<<"Thisrateisn’tcorrect\n";
}
}
```

Мысал регистрлік сезімталдыққа кезектескен caseкөмегімен қол жеткізілетіндігімен ерекшеленеді. Яғни, пайдаланушының қай әріпке кіретіні маңызды емес - үлкен немесе кіші.

Жалпы кездесетін қате.

Switch туралы барлық маңызды нәрселер айтылды, тек осы операторды пайдалану кезінде бағдарламашы қандай проблемаға тап болуы мүмкін екендігі туралы ақпарат алу ғана қалды.

Егер сіз кездейсоқ кез келген CASE блогында BREAK жіберіп алмасаңыз, соңғысынан басқа, және бұл блок кейіннен жұмыс істейді, онда SWITCH орындалуы тоқтамайды. Іс бойынша мәлімдеме блогы, ол орындалғаннан кейін болады, сонымен қатар тексерусіз орындалады.

Қатенің мысалы.

```
#include<iostream>usingnamespacestd;voidmain(){
//реализацияпрограммногоменюintaction;
cout<<"\nSelectaction:\n";
cout<<"\n1-if you wanttoseecourseofdollar.\n";cout<<"\n2-
ifyouwanttoseecourseofeuro.\n";cout<<"\n3-if you wanttoseecourseofrub.\n";
//ожиданиевыборапользователясin>>action;
//проверяется значение переменной actionswitch(action){
case1://еслипользовательвыбралдолларcout<<"\nCource:5.2gr.\n";
break;//остановкаswitch
case2://еслипользовательвыбралевроcout<<"\nCource:6.2gr.\n";
//break;закомментированнаостановкаswitchcase3:
//еслипользовательвыбралрубли
cout<<"\nCource:0.18gr.\n";break;//остановкаswitch
default:       //есливыборнекорректен
cout<<"\nError!!!Thisoperatorisn’tcorrect\n";break;//остановка switch
```

Мәзірдің 2 -тармағы таңдалса, қате пайда болады. Егер мән 2 болса, BREAK stop операторы түсіндіріледі. Экранда мұндай қатенің нәтижесі осылай көрінеді:

```
Selectaction:
1-ifyouwanttoseecourseofdollar.2-ifyouwanttoseecourseofeuro.
3-ifyouwanttoseecourseofrub.2
Cource:6.2gr.
Cource:0.18gr.
Pressanykeytocontinue
```

Цикл туралы түсінік

Көбінесе өмірде де, бағдарламаны жазу кезінде де әрекетті бірнеше рет қайталау қажет болады. Мысалы, ыдыс жууды жүзеге асыратын алгоритмді елестетіп көрейік.

```
Взять тарелку израковины.
Намылить тарелку средством для мытья посуды.
Потереть тарелку мочалкой.
Смыть мыльную пен устарелки.
Вытереть тарелку.
Поставить тарелку на полку.
Конец программы.
```

Бұл, бір қарағанда, ақылға қонымды алгоритмде бір кішкене сәйкессіздік бар - егер бірнеше пластиналар болса, бәрібір олардың біреуі ғана жуылады. Бұл бағдарлама барлық әрекеттерді сызықты түрде - жоғарыдан төменге қарай ретімен орындайтындығына байланысты.

Сондықтан, біз бағдарламаны нақты әрекеттер жиынтығын қалай қайталауға болатынын анықтауымыз керек және сонымен бірге қайталанудың қажетті санын анықтауымыз керек. Дұрыс алгоритм осылай көрінеді.

Взять тарелку из раковины.
Намылить тарелку средством для мытья посуды.
Потеретьтарелкумочалкой.
Смытьмыльнуюпенустарелки.
Вытереть тарелку.
Поставить тарелку на полку.
Если есть еще грязные тарелки вернуться к пункту 0.
Конец программы.

Әрекеттерді қайталау керектігін анықтау үшін алдымен шартты қолданыңыз«Еслиестьещегрязныетарелки».Егер бұл шарт дұрыс болса, онда әрекеттер қайталанады, егер жалған болса, алгоритмнің келесі 7 -ші пункті орындалады.

Сонымен, біз қайталауға арналған әрекеттер жиынтығын қамтитын қандай да бір құрылым қажет деген қорытындыға келдік. Бұл жағдайда қайталау саны сол құрылымдағы кейбір шарттарға байланысты болуы керек.

Цикл - бұл белгілі бір шартқа байланысты әрекетті қажетті ретпен орындауға болатын программалау тілінің арнайы операторы.

С тілінде цикл сияқты форманың бірнеше іске асырылуы бар. Бұл сабақта осындай екі іске асыруға тоқталамыз - **while** және **do while**.

While циклы

While циклінің жалпы синтаксисі мен орындалу тәртібі

```
while(утверждение)
{
действиедляповторения;
}
```

1. Бірінші қадам - мақұлдауды тексеру.
2. Егер жақшадағы тұжырым рас болса, бұйра жақша ішіндегі әрекет орындалады.
3. Егер жақша ішіндегі оператор жалған болса, онда бағдарлама циклдің бұйра жақшасы жабылғаннан кейін келесі жолға өтеді.
4. Егер жақшадағы сөйлем шын болса және әрекет сәтті болса, онда мәлімдеме қайтадан тексеріледі.

Көріп отырғаныңыздай, бекіту циклы іске қосылған сайын қайталанады. Ақиқат болмайынша, цикл аяқталады. Назар аударыңыз, егер мәлімдеме басынан бастап жалған болса, цикл ішіндегі әрекет бір рет те орындалмайды.

Сурет 20. Цикл

Мысалды қарастырайық.

Белгілі бір адамға әлемнің 7 кереметі туралы эссе жазу керек делік. Мұны жасамас бұрын, ол барып, кереметтердің әрқайсысына қарау керек. Содан кейін ғана соңғысы туралы жазыңыз. Жобаның аты - **Miracles.**

```
#include <iostream>usingnamespacestd;voidmain()
{
//объявлениеуправляющейпеременнойintcounter=0;
while(counter<7)//проверкаутверждения
{
counter++;//изменениеуправляющейпеременной
//действиедляповторения
//выувидели...чудосвета
cout<<"Youseen»<<counter<<”miracleofworld!!!\n”;
}
cout<<"Now,youcanstartyourwork.\n";
}
```

Енді біздің мысал қалай жұмыс істейтінін егжей -тегжейлі қарастырайық.

1. Біз бастапқыда 0 -ге тең айнымалы деп жариялаймыз

2. Әрі қарай, цикл жағдайында біз айнымалы мәнін тексереміз. Цикл орындалатынына немесе болмайтынына осы мән байланысты болғандықтан, мұндай айнымалы циклды басқару айнымалысы деп аталады.

3. Біз айнымалының мәнін бірге арттырамыз.

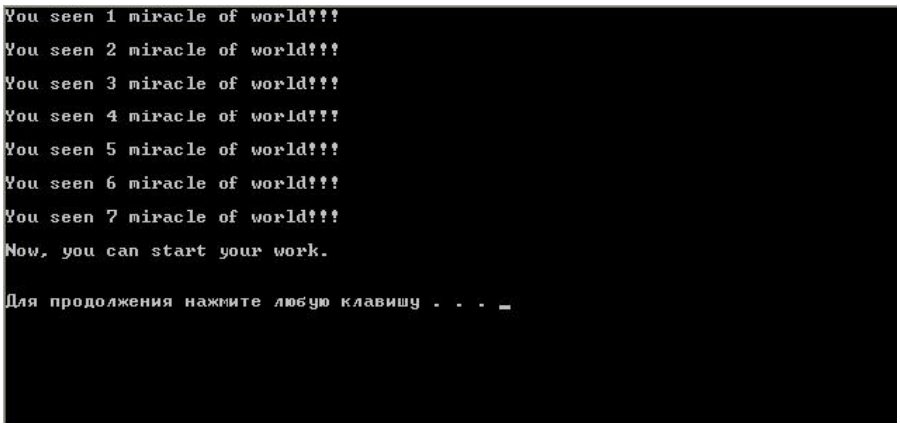
Ескерту: Бұл әрекет қажет, себебі егер сіз циклды басқаратын айнымалының мәнін өзгертпесеңіз, бекітуді тексеру нәтижесі ешқашан өзгермейді. Бұл мәңгілік цикл деп аталатын өте жиі кездесетін қатеге әкелуі мүмкін. Егер цикл операторы ақиқат болса және басқару айнымалы мәні әрқашан бірдей болса, онда оператор әрқашан ақиқат болады. Елестетіп көріңізші, лас ыдыстар ешқашан таусылмайды - олардың саны үнемі тұрақты. Ыдыс жуғыш машина қанша уақытқа созылады?! Ұзақ емес, иә? Сонымен, бағдарлама мұндай шабуылға төтеп бере алмайды, және біраз уақыттан кейін мәңгілік

циклды бастағаннан кейін, ол орындалу сатысында қате береді. Мұндай қателіктерге жол бермеу үшін басқару айнымалысының цикл денесінің ішінде өзгеретініне көз жеткізу керек.

4. Әрі қарай, біз айнымалы мәннің ағымдағы мәнін әлемнің таңғажайып саны туралы хабарлама түрінде көрсетеміз.

5. Тағы да біз шартқа оралып, басқару айнымалысының мәнін тексереміз.

Айнымалының мәні 7 -ге тең болғанға дейін цикл өз жұмысын жалғастырады. Бұл жағдайда экранда «You seen 7 miracle ofworld!!!» деген жол шығады, содан кейін бағдарлама шартты тексеруге оралады. 7 <7 - жалған. Бағдарлама енді циклге енбейді және «Now,youcanstartyourwork.» жолына өтеді. Бағдарламаны орындау барысында экранда келесі суретті көреміз:



Сурет 21.Бағдарламаны орындау барысы

8. Dowhile конструкциясы

Жалпы синтаксис жәнedowhile жұмыс принципі:

```
do
{
действие;
}
while(условие);
```

DO WHILE циклы WHILE цикліне ұқсас. Айырмашылығы - WHILE жағдайында циклға кіргенде бірден шарт тексеріледі, содан кейін ғана, егер шарт дұрыс болса, әрекет орындалады. DO WHILE ішінде, кез келген жағдайда, әрекет алдымен орындалады, содан кейін ғана шарт тексеріледі. Егер шарт шын болса, әрекеттің орындалуы жалғасады, ал егер олай болмаса, онда WHILE операторынан кейін келесі операторға ауысады. Басқаша айтқанда, WHILE -ден айырмашылығы, DO WHILE ішінде әрекет кем дегенде бір рет орындалады. Мұны диаграммада қарастырайық:

Сурет 22. Әрекет циклі

Практикада dowhile қолдану

Пайдаланушыға бірнеше рет қатарынан әрекетті таңдау құқығы берілген бағдарлама жазу керек делік. Бұл тапсырманы алдымен WHILE көмегімен, содан кейін DO WHILE көмегімен іске асырайық. Жобаның атауы Calc.

```
// запрос на выбор операциисout<<"\nSelectoperation:\n";
cout<<"\n 1 - if you want to see SUM.\n";cout<<"\n2-
ifyouwanttoseeDIFFERENCE.\n";cout<<"\n3-ifyouwanttoexit.\n";
cin>>answer;
}
cout<<»\nBye.      \n»;
}
```

Бұл мысалда пайдаланушыға әрекетті таңдау ұсынылады. Содан кейін енгізгеннен кейін бағдарлама тексереді: егер бұл әрекет бағдарламадан шығу болса, бағдарлама аяқталады, егер олай болмаса, онда ол циклге кіреді, әрекетті талдайды және математикалық операцияны орындайды. Содан кейін бағдарлама пайдаланушыдан не істегісі келетінін сұрайды.

Бұл код оңтайлы шешім емес. Көріп отырғаныңыздай фрагмент

```
// запрос на выбор операциисout
<<»\nSelectoperation:\n»;
cout<<»\n 1 - ifyouwanttoseeSUM.\n»;cout<<»\n2-
ifyouwanttoseeDIFFERENCE.\n»;cout<<»\n 3 -
ifyouwanttoexit.\n»;cin>>answer;
```

бірнеше рет қайталанады. Бұл жағдайда do while қолдану керек. Бұл конструкция кодты тиісті формасына жеткізеді. Жобаның аты CalcDoWhile.

```
#include<iostream>usingnamespacestd;voidmain()
{
int answer,A,B,RES;do{//входвцикл
// запрос на выбор операциисout<<"\nSelectoperation:\n";
cout<<"\n 1 - if you want to see SUM.\n";cout<<"\n2-
ifyouwanttoseeDIFFERENCE.\n";cout<<"\n3-ifyouwanttoexit.\n";
```

```
cin>>answer;
//анализдействияswitch(answer){
case1://еслипользовательвыбралсложениеcout<<"Enter first
digit:\n";cin>>A;
cout<<"Enterseconddigit:\n";cin>>B;
RES=A+B;
cout<<"\nAnswer: "<<RES<<"\n";break;//остановкаswitch
case2://еслипользовательвыбралвычитаниеcout<<"Enterfirstdigit:\n";
cin>>A;
cout<<"Enterseconddigit:\n";cin>>B;
RES=A-B;
cout<<"\nAnswer: "<<RES<<"\n";break;//остановкаswitch
case3://еслипользовательвыбралвыходcout<<"\nEXIT!!!\n";
break;
default:                // если выбранное действие некорректно
cout<<"\nError!!!Thisoperatorisn’tcorrect\n";
}
```

Жоғарыда айтылғандарға сүйене отырып, сіз бүгінгі сабақта сипатталған конструкциялардың екеуі де пайдалы екенін түсінуіңіз керек. Сіз тек тапсырмаға байланысты біреуін немесе екіншісін таңдауды үйренуіңіз керек.

For құрылымы

Өткен сабақта біз цикл сияқты ұғыммен таныстық және C тілінде циклды білдіретін кейбір конструкцияларды қарастырдық. Атап айтқанда - while жәнеdo while. Енді біз циклдің басқа түрін қарастырамыз - for операторы. Бұл оператор теориялық тұрғыдан WHILE -ге толық ұқсастығы болып табылады, бірақ іс жүзінде ол ыңғайлы басқаруымен циклды ұйымдастыруға мүмкіндік береді.

Жалпы синтаксис және for құрылымының жұмыс принципі

```
for(инициализацияпеременной;проверкаусловия;измененипеременной
)
{
действие;
}
```

Цикл принципі:

Инициализация переменной.
Проверкаусловия.
Выполнение действия, если условие истинно.
Если условие ложно, выполнении еследующего за циклом оператору.

Если условие было истинно-изменение управляющей переменной.
Проверка условия. Далее сновапункт 3 или 4.

Сурет 23. Цикл принципі

Қолдану мысалы

Қарапайым таныс мысалды қарастырайық: 1 -ден 5 -ке дейінгі сандарды көрсету үшін циклды қолдану. Мұны FOR операторының көмегімен жасайық.

```
#include<iostream>voidmain()
{
for(inti=1;i<=5;i++)
{
cout<<i;
}
}
```

Мысалға түсініктеме.

1. Цикл ішінде і айнымалысы 1 -ге тең деп жарияланады. Бұл басқару айнымалысы болады.

2. Содан кейін, бұл айнымалының мәні і <= 5 шарты арқылы тексеріледі;

3. Егер шарт дұрыс болса (және ол і мәніне 6 жеткенше болады), экранда і мәні көрсетіледі (cout << i;) және і айнымалы айнымалысы 1 -ге өзгереді (і ++) . Содан кейін шарт қайтадан тексеріледі.

4. Егер шарт жалған болса (яғни і мәні 6 -ға тең болса), онда бағдарлама циклдің бұйра жақшасы жабылғаннан кейін келесі жолға өтеді.

Ескерту: Назар аударыңыз, бірінші қадам – АЙНЫМАЛЫ ЖАСАУ ЖӘНЕ БАСТАУ –әрқашан бір рет орындалады.

For синтаксистің кейбір ерекшеліктері

Оператордың жұмысының қарапайымдылығына қарамастан, ол жазба формаларының кейбір ерекшеліктеріне ие.

Айнымалы инициализацияны басқару

1.Айнымалы инициализация және құру циклде орындалады.

```
for(intx=1;x<=100;x++)
{
cout<<x;
}
```

2.Айнымалы құру цикл алдында, ал инициализация циклде орындалады.

```
int x;for(x=1;x<=100;x++)
{
cout<<x;
}
```

3.Айнымалы инициализация және құру цикл алдында орындалады.
4.

```
int x;for(x=1;x<=100;x++)
{
cout<<x;
```

Барлық үш мысал мүлдем функционалды және теңдестірілген.

Басқару айнымалысын өзгерту.

Басқару айнымалысының өзгеруін while және do while болатындай цикл денесінің ішінде жүргізуге болады.

```
for(intx=1;x<=100;)
{
cout<<x;x++;
}
```

Шарт.

Құрылыстың шартын да елемеуге болады, бірақ бұл жағдайда ол әдепкі бойынша дұрыс болып саналады. Осылайша, біз үнемі шынайы жағдайды аламыз және нәтижесінде МӘҢГІЛІКЦИКЛді аламыз.

```
for(intx=1;;x++)
{
cout<<x;
}
```

Ескерту: Егер сіз шартты өткізіп жіберуді және мәңгілік циклден аулақ болуды үйренгіңіз келсе, оқулықтың келесі бөлімін оқыңыз.

Жоғарыда айтылғандардың негізінде мынадай қорытынды жасауға болады: **For циклінің бөліктерінің ешқайсысы қажет емес.**

Көріп отырғаныңыздай, for қарапайым және whileға ұқсас. Не таңдау керек?! Бұл сіздің тапсырмаңызға және сіздің шешіміңізге байланысты.

Break операторы

Көбінесе циклмен жұмыс кезінде циклдің орындалуын жасанды түрде тоқтату қажет болады.Мұны істеу үшін сізге таныс break операторын қолданыңыз (switch-тық зерттеуден).Бұл мәлімдеме цикл денесінде, тоқтағыңыз келетін жерде болуы керек.Мысалы, дәл осы оператордың көмегімен біз for циклінде шарт көрсетілмеген жағдайда мәңгілік цикл мәселесін шеше аламыз. Мысал қарастырайық:

```
#include<iostream>usingnamespacestd;voidmain()
{
for(intx=1;;x++)
{
if(x==4)break;//еслихсталравен4-остановитьциклcout<<x;

}
cout<<”Bye!”;
}
```

Мысалға пікірлер

1. Ережеге сәйкес, цикл шарты әрқашан ақиқат, себебі ол жоқ.

2. X айнымалысының 1, 2 және 3 мәндері үшін if операторының шарты орындалмайды. Break, әрине, жұмыс істемейді, себебі ол if денесінде. Бұл кезде экранда 1, 2, 3 сандары ретімен көрсетіледі.

3. X 4 -ке тең болғанда, бағдарлама if денесіне кіреді және break орындалады. Цикл дереу тоқтайды және for операторының бұйра жақшасы жабылғаннан кейін бағдарламаның орындалуы келесі жолға жалғасады.

4. Экранда Bye! жазуы шығады

5. 4 саны экранда ешқашан пайда болмайды, себебі егер break басталса, оның астындағы циклдегілердің бәрі енді орындалмайды.

Ескерту: break циклде де, switch операторында да қолданыла алады. Кез келген басқа орналастыру компиляция уақытында қатеге әкеледі.

Continue операторы

Continue операторы ағымдағы циклді қайталауды тоқтату және келесі қадамға өту үшін қолданылады. Кейбір жағдайларда мұндай әрекеттер қажет. Егер Continue операторы орындалса, онда цикл түріне байланысты келесі жағдай орын алады:

While пendo while циклдары қадамды тоқтатады және жағдайды тексеруге өтеді.

For циклы да қадамның орындалуын тоқтатады. Бірақ алдымен басқару айнымалысын өзгертуге, содан кейін шартты тексеруге көшеді.

Мысалды қарастырайық: экранда нөлден 25 -ке дейінгі диапазонда барлық тақ бүтін сандарды көрсету. Жобаның атауы - Odd.

```
#include <iostream>usingnamespacestd;voidmain()
{
for(inti=0;i<26;i++)
{
if(i%2==0)//есличислоделитсянадвабезостатка
{
continue;//остановитьитерациюциклаиперейтикі++
}
cout<<i<<"\n";
}
```

Мысалға пікірлер

1. Цикл нөлден басталады және 25 -ке дейін қайталанудан өтеді.
2. Цикл ішінде шарт бар: егер і саны жұп болса, циклдің ағымдағы қадамын тоқтатып (continue;) және і ++ құрылысына өту керек.
3. Ағымдағы қадамда іске қосылған жалғастыру операторының астындағы кез келген нәрсе енді орындалмайды.
4. Егер if шарты орындалмаса, онда і саны тақ болады, if еленбейді, және нөмір экранда көрсетіледі.

Енді сабақтың теориялық материалдарымен таныс болдық, келесі бөлімге көшейік, онда біз бірнеше практикалық тапсырмаларды қарастырамыз.

Кірістірілген құрылым

Алдыңғы сабақтарда сіз цикл деп аталатын конструкциямен және C тілінде циклды енгізу нұсқаларымен таныстық.Сіз байқағандай, цикл - бағдарламалаудың негізгі конструкцияларының бірі.Оның көмегімен көптеген міндеттер шешіледі. Сондай -ақ, сіз IF және SWITCH сияқты логикалық таңдау конструкцияларын циклге салуға болатынын кездестірдіңіз. Дегенмен, біз мұнымен тоқтап қалмай, соған ұқсас құрылысты циклге, яғни басқа циклге енгізуге тырысамыз. Қарапайым мысалды қарастырайық:

```
#include <iostream>usingnamespacestd;voidmain()
{
int i=0,j;while(i<3){
cout<<"\nOut!!!\n";j=0;
while(j<3){
cout<<"\nIn!!!\n";j++;
}i++;
}
cout<<"\nEnd!!!\n";
```

}

тырысамыз. Қарапайым мысалды қарастырайық:

Жобаның атауы

NestedLoop.

Мысалға талдау жасайық:

1. Бағдарлама i <3 шартын тексереді, 0 3 -тен кіші болғандықтан, шарт ақиқат және бағдарлама сыртқы циклге кіреді.

2. ЭкрандаOut!!!көрсетіледі

3. j айнымалысы тазаланады.

4. Енді j <3 шарты тексеріледі, 0 3 -тен кіші болғандықтан, шарт ақиқат және бағдарлама ішкі циклге кіреді.

5. ЭкрандаIn!!!көрсетіледі

6. j басқару айнымалысы өзгертіледі.

7. j <3 шарты қайтадан тексеріледі, себебі 1 3 -тен кіші, шарт ақиқат және бағдарлама ішкі циклге кіреді.

8. ЭкрандаIn!!!көрсетіледі

9. j басқару айнымалысы өзгертіледі.

10. j <3 шарты қайтадан тексеріледі, себебі 2 3 -тен кіші болғандықтан, шарт ақиқат және бағдарлама ішкі циклге кіреді.

11. ЭкрандаIn!!!көрсетіледі

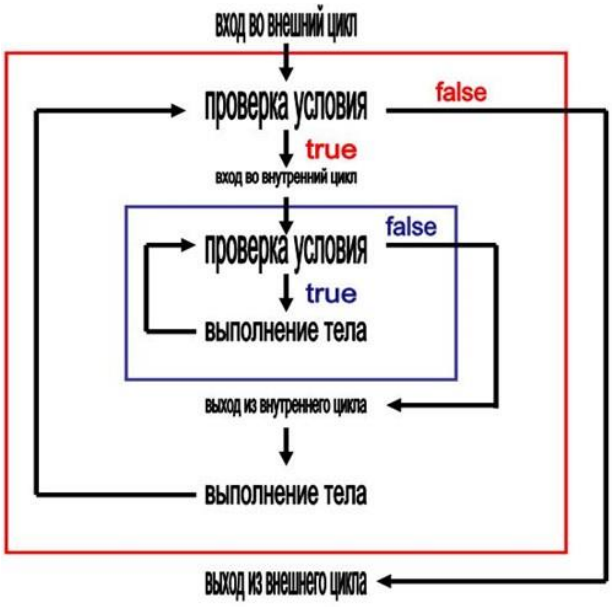
12. j басқару айнымалысы өзгертіледі.

13. j <3 шарты қайтадан тексеріледі, себебі 3 кем дегенде 3, шарт жалған және бағдарлама ішкі циклден шығады.

Содан кейін код 1 -қадамға оралады. Жоғарыда көрсетілген барлық әрекеттер (1–13) 3 рет қайталанады, яғни i 3 -ке тең болғанға дейін. Содан кейін бағдарлама сыртқы циклден шығады және End!!!

Кірістірілген циклды іске асыратын бағдарламаның жұмыс принципі сыртқы циклдің басынан аяғына дейін әрбір циклінде ішкі цикл толығымен орындалатындығына негізделген. Басқаша айтқанда, бағдарлама кірістірілген циклден шыққанша сыртқы циклдің орындалуы жалғаспайды. Төменде кірістірілген ілмектер қалай жұмыс істейтіні көрсетілген.

Жұмыс схемасы:



Сурет 24. Жұмыс схемасы

Көріп отырғаныңыздай, бәрі қарапайым, бірақ бұған қарамастан, салынған конструкциялар күрделі алгоритмдердің орындалуын айтарлықтай жеңілдетеді. Бұған көз жеткізіңіз, біз сізге бірнеше мысалдар дайындаған оқулықтың келесі бөлімін қараңыз.

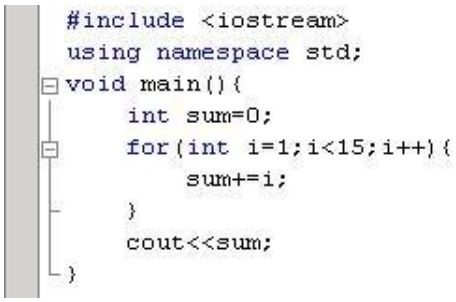
9. Microsoft Visual Studio интеграцияланған отладчикті қолдану

Түзету туралы түсінік. Отладчикті қолдану қажеттілігі. Өздеріңіз білетіндей, бағдарламалық қателердің екі түрі бар.

Компиляция уақытындағы қате-бұл бағдарламалау тілінің синтаксис қатесі. Мұндай қателерді немесе қателерді компилятор бақылайды. Мұндай қате бар бағдарлама жай орындалмайды және компилятор қатенің қай код жолында болғанын көрсетеді. Жұмыс уақыты қатесі - бұл бағдарламаның дұрыс жұмыс істемеуіне немесе соңғысының толық тоқтауына әкелетін қате. Мұндай қатені компилятор бақыламайтынын есте ұстаған жөн. Тек сирек жағдайларда ғана компилятор қате нұсқау туралы ескерту жасай алады, бірақ тұтастай алғанда, мұндай жағдайларда бағдарламашы өзін шығаруға мәжбүр болады.

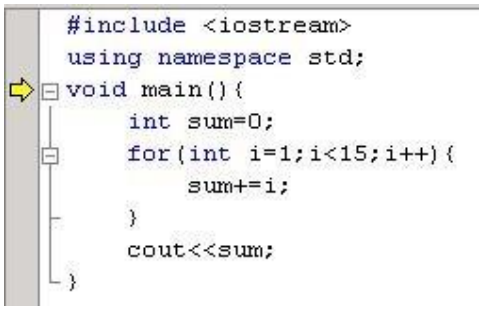
Бұл талқыланатын жұмыс кезеңіндегі қателер туралы. Көбінесе мұндай қатені анықтау үшін, бағдарлама жұмыс істеп тұрғандай, бағдарламаның белгілі бір фрагментін кезең -кезеңімен өту қажет. Әрине, бұл жағдайда нақты айнымалыларда белгілі бір уақытта қандай мән болатынын нақты есептеген жөн. Сіз, әрине, қағаз бетінде бағдарламаны жол бойынша талдау арқылы жасай аласыз, бірақ Visual Studio әзірлеу ортасында бағдарламалық талдауды ұйымдастырудың арнайы құралы бар - отладчик.

Бағдарламаны кезең -кезеңмен орындау.
Келесі кодты талдайтын боламыз делік:



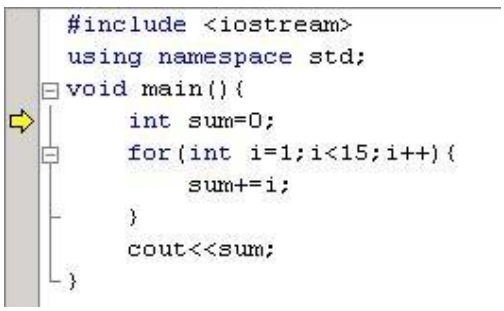
Сурет 25. кодтар

Алдымен жоба жасаңыз және осы кодты теріңіз. Оны құрастырыңыз және синтаксистік қателер жоқ екеніне көз жеткізіңіз. Енді бастайық. Пернетақтадағы F10 функция пернесін басыңыз. Экранда сіз көрсеткен кодтың бірінші жолының жанында сары көрсеткі пайда болады.



Сурет 26. Орындалатын код жолы

Дәл осы көрсеткі дәл қазір қандай код жолының «орындалатынын» көрсетеді. Бағдарламаның келесі қадамына өту үшін F10 пернесін қайта басыңыз. Және сізді келесі жолға апарады:

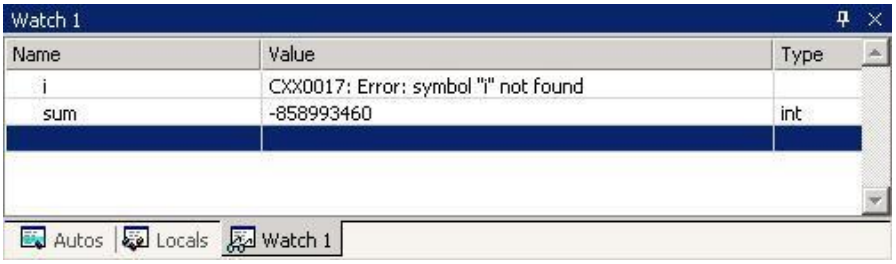


Сурет 27. Бағдарламаның қадамы

Назар аударыңыз, экранның төменгі жағында айнымалыларды талдауға арналған қойындылар жиынтығы бар:

Autos—бұл қойынды ағымдағы код жолын орындау кезінде бар айнымалылардың мәндерін қарауға арналған.Бұл қойындыға өзіңізден ештеңе енгізе алмайсыз - бұл автоматты функция.

Сурет 28. ағымдағы код жолын



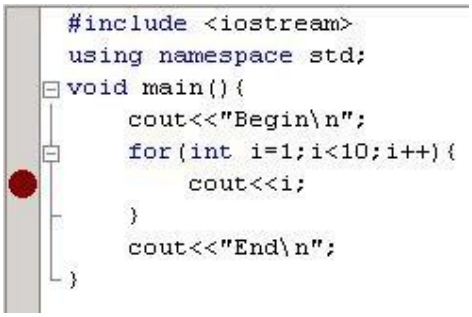
Сурет 29. Айнымалы код жолы

Watch—өзіңізді қарау үшін айнымалыны таңдау қажет болған жағдайда арналған. Сіз жай ғана Name өрісіне айнымалы атауды енгізесіз және ол орындалатын кодқа қарамастан көрсетіледі. Енді F10 пернесін басу арқылы кодты «өтіңіз» және қойындылардағы мәліметтер қалай өзгеретінін көріңіз.

Тоқтау нүктесі.

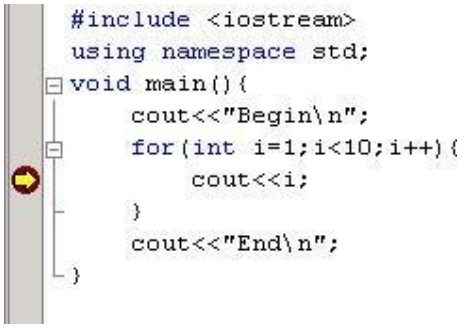
Біз кодтың бір бөлігін орындау қажет болған жағдайды қарастырып, белгілі бір жерде тоқтап, отладчикті іске қосамыз. Ол үшін тоқтау нүктесі деп аталады.

Келесі кодты теріңіз - cout << і жолының жолына қойыңыз; және F9 пернесін басыңыз. Сызықтың жанында қызыл нүкте пайда болады, бұл тоқтау нүктесі.



Сурет 30. Тоқтау нүктесі.

Енді F5 пернесін басыңыз, бағдарлама іске қосылады, тоқтау нүктесі орнатылғанша орындалады және отладчик режиміне өтеді.



Сурет 31. Отладчик режиміне

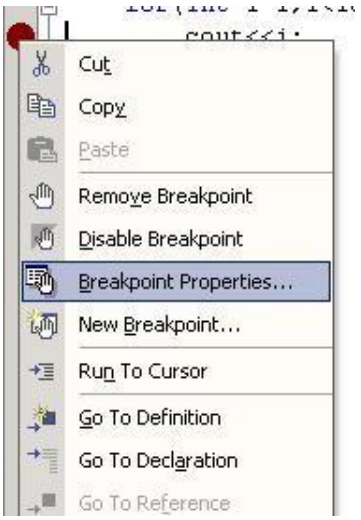
Консольдің күйіне назар аударыңыз (бағдарлама терезесі). Болғанның бәрі мына жерде көрсетіледі:

Келесі - отладчиктің қалыпты жұмысы. Сары көрсеткіні F10 көмегімен жылжытыңыз және айнымалылармен не болатынын қараңыз. Сонымен қатар, консоль терезесін қараңыз, кодтағы барлық өзгерістер сол жерде көрсетіледі.

«Ақылды»тоқтау нүктесі.

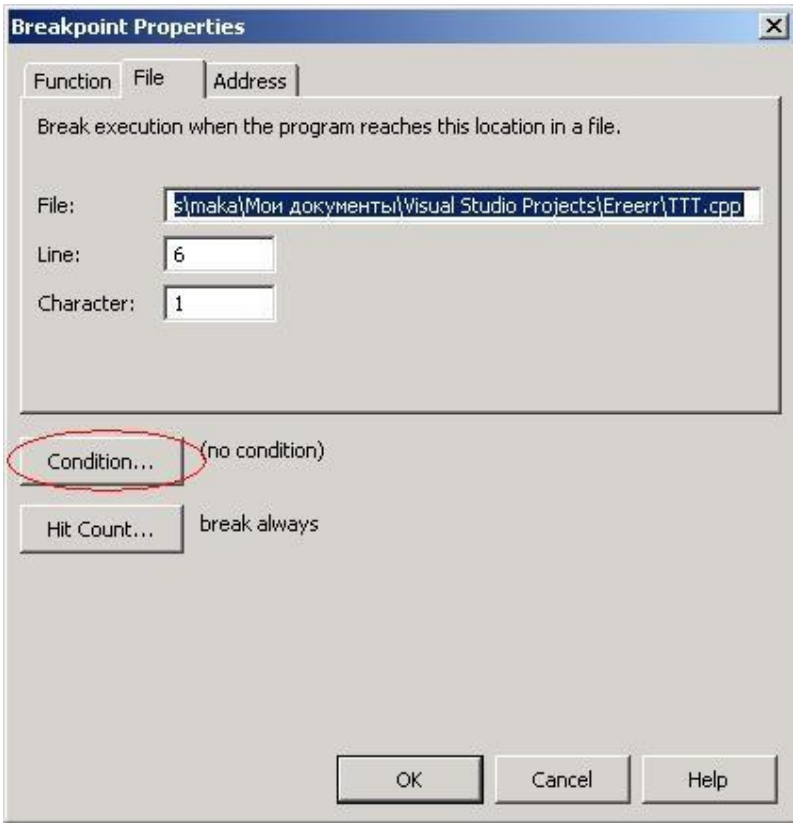
Біз бағдарламаны белгілі бір жерден талдай бастадық. Алайда, отладчик бірден цикл денесін орындау басталғаннан кейін, яғни бірінші қайталаудан басталғанын ескеріңіз. Егер қайталанулар көп болса, бұл ыңғайсыз, және олардың бірнешеуін өткізіп жіберу қажет. Басқаша айтқанда, сіз талдауды, мысалы, циклдің 5 -ші қайталаудан бастағыңыз келеді. Шешім қарапайым - тоқтау нүктесін «ақылды» етіңіз.

Мұны істеу үшін нүктенің өзін тінтуірдің оң жақ түймешігімен нұқыңыз және ашылатын мәзірде таңдаңыз.



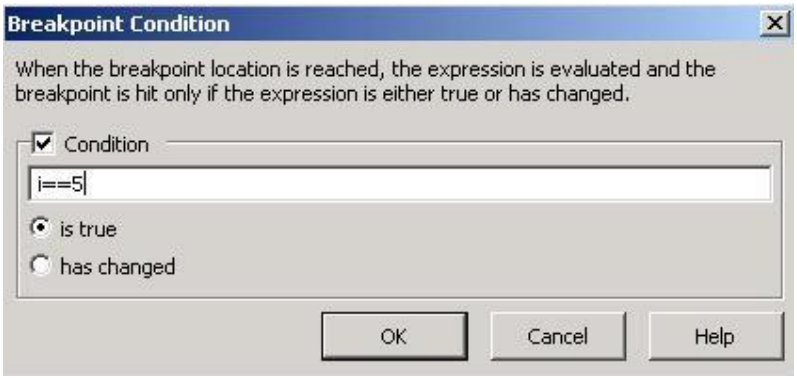
Сурет 32. Breakpoint Properties командасы

Сіздің алдыңызда терезе пайда болды. File, Lineжәне Character - бұл нүкте орнатылатын файл, жол және позиция. Бізді Condition түймесі қызықтырады, оның жанында - (no condition). Біз оны басамыз.



Сурет 33. Condition түймесі

Терезе пайда болды, онда біз отладчик басталатын шартты енгізуіміз керек. Біздің шарт - i == 5. Біз is true таңдаймыз, яғни шарт орындалған кезде тоқтатамыз. OK түймесін басыңыз.

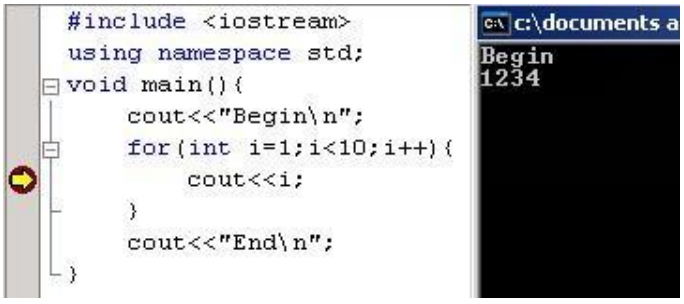


Сурет 34. Шарт орынлуы

Енді түйменің жанында тағы бір жазу бар. Негізгі терезеде OK түймесін басыңыз.

Сурет 35. Негізгі терезе

Барлығы дайын болғанда, F5 пернесін басып, не болатынын көріңіз. Көріп отырғаныңыздай, бәрі жақсы - отладчик бізге қажет сәтте басталды.



Сурет 36. Отладчик

10. Мәліметтерді топтастыру қажеттілігі

Бүгін біз сізбен деректерді сақтау туралы сөйлесетін боламыз. Алғашқы сабақтардың бірінде біз айнымалының бар екенін білдік және оны ақпаратты сақтауға арналған жедел жадтың бөлігі ретінде анықтадық. Кәдімгі бағдарлама айнымалыларсыз өмір сүре алмайтыны сөзсіз, бірақ кейде қарапайым айнымалылар деректерді өңдеу мәселелерін шешпейді. Ал, алдыңғы сабақтарда қарастырылған айнымалылардың әрқайсысы бір уақытта бір ғана ақпаратты сақтай алатындығында. Екіншісін сақтау үшін басқа

айнымалы жасау қажет. Бірақ сізге біртекті деректер түрлерінің көптеген элементтерін сақтау қажет болса ше? Әр элемент үшін айнымалыны құру өте ыңғайсыз болады. Бірақ сізге жүздеген элементтермен жұмыс істеу қажет болса ше? Тапсырма тез орындалмайды. Келісіңіз, бірнеше жүз айнымалы жасау – бұл ақылсыздық.

Мұндай қиын болып көрінетін тапсырманы қалай шешуге болады?!

Біздің жағдайда, шешім массивтер деп аталады. Анықтамасы мен ерекшеліктерін қарастырыңыз.

Массив түсінігі.

1. Массив - бір типті бірнеше мәндерді сақтауға мүмкіндік беретін айнымалылар жиынтығы.

2. Бұл жиынтықтың барлық мағыналары бір атпен біріктірілген.

3. Сонымен қатар, массивтегі әр айнымалы элемент деп аталатын тәуелсіз бірлік болып табылады.

4. Әр элементтің өзіндік реттік нөмірі - индексі болады. Индекс бойынша сіз массивтің белгілі бір элементіне сілтеме жасай аласыз.

5. Массив элементтерінің нөмірленуі нөлден басталады.

Схема:

Жоғарыда келтірілген мәлімдемелерге сүйене отырып - массивті ұсынудың жалпы схемасы келесідей болады:

Сурет 37. Массив

Массивтің жадта орналасуы:

Массив жадыға элемент бойынша элемент ретімен орналастырылады. Алдымен нөл, содан кейін бірінші және т.б. Элементтер адрестің өсу ретімен орналасады: Массивтің бір элементі екіншісінен массивтің негізгі түріне тең байт санымен орналасады. Массивте орналасу орындалатын формула:

базалық адрес + базалық тип өлшемі * индекс;

Егер адрес дұрыс көрсетілмесе, базалық адрес формулалар бойынша есептелген мекен -жайға орналастырылады. Бұл жағдайда бағдарлама жад ұяшығының мазмұнына толық қол жеткізеді, ол іс жүзінде оған жатпайды. Бұл жұмыс уақытында қатеге әкелуі мүмкін.

Сурет 38. Массив Элементі

Қорытындылай келе, массивтің әрбір элементінің барлық массивтің түріне тікелей тәуелді болатын өзіндік өлшемі бар екенін атап өткен жөн. Мысалы, егер массив int деректер түрінен болса, ондағы әрбір элементтің өлшемі 4 байт. Осылайша, бүкіл массивтің жалпы өлшемі формула бойынша есептеледі:

ЖАЛПЫ ӨЛШЕМІ = ДЕРЕКТЕР ТҮРІ МӘЛІМІ * ЖИЫНТЫҚТАҒЫ ЭЛЕМЕНТТЕР САНЫ

11. Массив құру және оны мәліметтермен толтыру

Массив декларациясының синтаксисі.

Біріншіден, біз массив құруды үйренуіміз керек. Және бұл үшін - біріншіден, жалпы синтаксисті анықтау. Екіншіден, бұл синтаксис қандай ережелер мен шектеулерге бағынатынын анықтаңыз.

тип_данныхимя_массива[количество_элементов];

1.Тип_данных—сізге белгілі кез келген деректер түрлері. Бұл массивтің әрбір элементінде болатын түрі.

2. Имя_массива—«айнымалы атаулар ережелеріне» бағынатын кез келген атау (біз бұл ережелерді сізбен бірінші сабақта қарастырдық).

3. Количество_элементов — массив элементтерінің саны. Бұл орын - бүтін тұрақты мән болуы керек. Бұл мән бүтін сан немесе тұрақты бүтін айнымалы болуы мүмкін.

Ескерту: Назар аударыңыз, массив элементтерінің саны бағдарламаны құру кезеңінде анықталуы керек. Басқаша айтқанда, қандай да бір шартқа немесе пайдаланушының шешіміне байланысты мұндай массивтің өлшемін орнату мүмкін емес. Бұл компиляция уақытында қатеге әкеледі.

Бірінші нұсқа.

5 элементтен тұратын ar массиві жарияланады, олардың әрқайсысы int деректер түріне жатады.

intar[5];

Екінші нұсқа.

Тұрақты size жарияланады, оның мәні 3, содан кейін, әрқайсысы double деректер түріне жататын 3 элементтен тұратын br массиві.

constintsize=3;doubleb
r[size];

Массив элементтеріне қол жеткізу.

Нақты массив элементіне қалай сілтеме жасау керектігін қарастырайық.

запись
значенияимя_массива[индекс_элемента]=значение;
получение значенияcout<<имя_массива[индекс_элемента];

Мұнда элементтер индексінің орнына кез келген бүтін санды ауыстыруға болады, оның ішінде өрнегі бар, нәтижесі бүтін сан.

constintsize=5;
intar[size];//создание массива
ar[2]=25;//запись значения 25 в элемент синдексом2
//вывод на экран значения элемента синдексом 2-25cout<<ar[2]<<"\n";

Массивті инициализациялау нұсқалары:

Деректермен массивті толтырудың екі әдісі бар:

Бірінші әдіс— құру кезінде инициализациялау.

деректер_түрі массив_атауы[элементтер саны]={1мән,2мән,…n мән;

constintsize=3;

Бұл инициализация формасының кейбір ерекшеліктері бар:

1. Инициализация тізімінің барлық мәндері массивтің өзінде бірдей деректер түріне ие, сондықтан элементтер санын құру кезінде көрсетудің қажеті жоқ. Операциялық жүйе инициализация тізіміндегі элементтер санына байланысты массивтің өлшемін өзі анықтайды.

деректер_түрі массив_атауы[элементтер саны]={1мән,2мән,…n мән};

intar[]={1,30,2};
/*Вданнойстрокемассивавтоматическиполучитразмер3.*/

2. Егер инициализация тізіміндегі элементтер саны массив элементтерінің санынан аз болса, онда қалған мәндер автоматты түрде нөлдермен толтырылады:

intar[5]={1,2,3}

мұндай жазба келесі белгіге тең:

intar[5]={1,2,3,0,0};

3. Егер инициализация тізіміндегі мәндер массив элементтерінің санынан үлкен болса, онда құрастыру сатысында қате пайда болады:

intarray[2]={1,2,3};//ошибканаэтапекомпиляции

4. Массивтерді инициализациялау кезінде сіз бұрыннан белгілі бірыңғай инициализацияны қолдана аласыз

intarr[]{1,2,3};

Екінші әдіс— циклды қолданып массивті инициализациялау.

Бұл жағдайда массивті мәндерді қолданушыны қолдана отырып толтыруға болады. Жобаның атауы - InitArray.

```
#include<iostream>usingnamespacestd;voidmain()
{
constintsize=3;
intar[size];//созданиемассиваизтрехэлементов
//циклперебирающийэлементымассивafor(inti=0;i<size;i++)
{
cout<<"Enterelement\n";
/*накаждойитерациициклапользователюподставляетсяэлементсиндексомідлязаполнения.секретвтом,чтоi-каждыйразновоезначение*/
cin>>ar[i];
}
}
```

Массив мазмұнын экранда көрсету.

Сіз элементтерді кезекпен қайталай отырып, ілмектерді қолдана отырып, массивтермен операциялардың көпшілігін жүргізу ыңғайлы деп болжаған шығарсыз. Бұл дұрыс және скринингтен тыс қалмайды. Міне, массив құратын, толтыратын және көрсететін толық бағдарламаның мысалы. Жоба атауы ShowArray.

```
#include<iostream>usingnamespacestd;voidmain()
```

```
{
constintsize=3;
intar[size];//созданиемассиваизтрехэлементов
//циклперебирающийэлементымассиваfor(inti=0;i<size;i++)
{
cout<<"Enterelement\n";
/*накаждойитерациициклапользователюподставляетсяэлементсиндексо
мідлязаполнения.секретвтом,чтоi-каждыйразновоезначение*/
cin>>ar[i];
}
cout<<"\n\n";
//циклперебирающийэлементымассиваfor(inti=0;i<size;i++)
{
//показэлементасиндексомінаэкранcout<<ar[i]<<"\n";
}
}
```

Енді біз массивтермен таныс болғандықтан, оқулықтың келесі бөлімдеріне көшіп, олармен жұмыс істеудің практикалық мысалдарын қарастырайық.

Бақылау сұрақтары.

1. Бағдарламалауды автоматтандыру әдістері.
2. Алгоритмдік тілдер.
3. Объектілі-бағытталған бағдарламалау және процедуралы-бағытталған тілдер туралы түсініктер.
4. Бағдарламаларды жобалау деңгейлері мен кезеңдері.
5. Алгоритмдердің құрылымдық схемасын жобалау.
6. Бағдарламалық қамтамасыз етуді жобалау әдістері.
7. Модульді бағдарламалау.
8. Бағдарламалау сапасының көрсеткіштері.
9. Бағдарламалық құжаттың бір тұтас жүйесі (БҚБЖ).
10. СИ бағдарламалау жүйесіне кіріспе.
11. Препроцессор директивасы.
12. Мәліметтер типі.
13. СИ тілінің операциясы.
14. СИ тілінің операторлары.
15. Шартты операторлары.
16. Цикл операторлары.
17. Енгізу және шығару функциялары.
18. Функциялар.
19. Жады класстары.
20. Көрсеткіштер мен адрестік арифметика.
21. Жадыны ұйымдастыру және адресация.
22. Бір өлшемді массивтер.
23. Жолдар. Екі өлшемді массивтер.
24. Мәліметтер құрылымы.
25. Объектілер мен класстар.
26. Операцияларды жүктеу.
27. Унарлы операцияларды жүктеу.
28. Операцияларды жүктеу.
29. Бинарлы операцияларды жүктеу.
30. Мұрагерлік. Базалы және туынды класстар.
31. Класстар иерархиясы.
32. Көрсеткіштер мен адрестер.
33. Жадыны басқару.
34. Объектілерге көрсеткіштер.
35. Виртуальды функциялар.
36. Достық функциялар.
37. Ағындар мен файлдар.
38. Ағындық енгізу/шығару.
39. Көп фалды бағдарламалар.
40. Файл аралық өзара әрекеттер.
41. Класстар шаблондары.

42. Ескертулер.
43. Шаблондардың стандартты кітапханасы.
44. Алгоритмдер.
45. Контейнерлер.
46. Итераторлар.
47. Ассоциативті контейнерлер.
48. Қолданушылық объектілерді сақтау.
49. Функциональды объектілер.
50. Көрсеткіштерге көрсеткіштер.

Тест тапсырмасы.
І-нұсқа.

1. C++ тілінде ең басты функция қандай?
A.main()
B.class()
C.result()
D.private()
T.global()

2. #include дегеніміз не?
A.Препроцессор директивасы
A.Функция
A.Библиотека
A.Оператор
A.Класс

3. Бағдарламада пікір қалдыру үшін қандай таңба қолданамыз?
A.// және /* */
D.;
C.:
D.&&
E.||

4. using namespace std; бағдарламаға не еңгізеді?
A.стандартты есімдер кеңістігін
B.стандартты функция
C.стандартты библиотека
D.арнайы класс
E.арнайы объект

5. int age; бұл бағдарламадағы age айнымалысы қандай дерек түрін сақтайды?
A.нақты сандар
B.таңба
C.жолдама
D.функция
E.класс

6. C++тілінде "Hello world" шығарудың дұрыс синтаксисі қандай?
A.cout << “Hello, world”;
B.print(“Hello, world”)
C.Console.WriteLine(“Hello, world”);
D.cout << “Hello, world”
E.cout >> “Hello, world”;

7. Мәтін сақталатын айнымалы құру үшін қандай деректер қолданылады?
A.string
B.int
C.char
D.short int
E.long long int

8. Таңба сақталатын айнымалы құру үшін қандай деректер түрі қолданылады?
A.char
B.int
C.long long int
D.string
E.short int

9. Нақты сан сақталатын айнымалы құру үшін қандай деректер түрі қолданылады?
A.int
B.float
C.double
D.string
E.char

10. 5 санысақталатын айнымалыны қалай құрамыз?
A.int x = 5;
B.string x = 5;
C.double x = 6;
D.class object x = 5;
E.number = 5;

11. Қандай библиотека файлы енгізу және шығару нысандарымен жұмыс істеуге мүмкіндік береді?
A.#include <iostream>
B.#include <string>
C.#include <vector>
D.using namespace std;
E.int main() {};

12. Екі мәнді теңділікке салыстыру үшін қандай операторды қолдануға болады?
A.==
B.=
C.&&

D.||
E.>>

13. Ситіліндециклоператорының нешетипібар?
A.3
B.1
C.2
D.4
E.9

14. Ситілі. Төмендегісипатталғанмассивтеқаншаэлементбар? int a[n][m]
A.m * N
B.m - n
C.m + n
D.m : n
E.0

15. Ситілі. int a[10] жазуы нені білдіреді:
A.массивтІ хабарлайды;
B.айнымалыны хабарлайды;
C. функцияны хабарлайды
D.құрылымды хабарлайды
T.дұрыс жауап жоқ

16. Ситілі. Массив - бұл
A.Бір типті элементтер жиынтығы
B.кез-келген элементтер жиынтығы
C.әр түрлі элементтер жиынтығы
D.айнымалылыр тізбегі
E.ережелер тізбегі

17. Ситілі. int a[] = {3,5,8,0,2} жазуы нені білдіреді?
A.Массив элементтерін енгізеді
B.құрылым элементтерін енгізеді
C.жиын элементтерін енгізеді
D.цикл бастамасы
E.функция бастамасы

18. Си тілі. Көпөлшемді массивті сипаттаудың дұрысын көрсет:
A.int a[2] [3] [2];
B.int a[2,3], [2];
C.int a[2,3,2];
D.int a[2] [3,2];
E.int a[2] [3,1];

19. Ситілі. continue ,break операторлары қандай команданы басқару үшін қолданылады?
A.циклді
B.тармақталукомандасын
C.таңдаукомандасын
D.деректердіенгізу, шығаруды
E.функция

20. Айнымалы –бұл …
A. белгілі бір типті мәліметтер сақталатын ат берілген жады обылысы
B. программалық объекті аты
C. константалардан, шамалардан, операция белгілерінен, функциялардан, жақшалардан тұратын құрылым
D. операндаларға қолданылатын әрекеттерді анықтайтын бір немесе одан көп символдар
E. өзгермейтін шама

21. Бөлгендегі қалдық табу опреациясы қалай беріледі?
A.%
B./
C.*
D.<
E.&

22. Сілтеме -бұл …
A. программа объектісі орналасатын жады өрісінің адресі
A. жады облысында үздіксіз орналасқан бір типті элементтер құрылымы
A. Си-дегі негізгі программалық бірлік
A. жады фрагменті
A. жады участогы

23. Экранға келесі программа не шығарады, егер 5,4,3 сандары енгізілсе?
{int a,b;
cin>>a>>b>>a;
cout<<a<<b<<a; }
A. 3 4 3
B. 4 3 5
C. 3 5 5
D. 3 4 5
E. 5 4 3

24. Өрнектің есептеу нәтижесінің қайсысы қате?
A. 3 / 7 = 3

B. 10 % 7 = 3
C. 1 / 5 = 0
D. 5 % 3 = 2
E. 4 % 6 = 4

25. Цикл деп… айтамыз.
A.командалардың бірнеше рет орындалуын
B.Шартсыз өту әрекеттерін
C.Бірінен соң бірі о рындалатын әрекеттерді
D.Шарттардың бірі орындалуын қамтамасыз ететін әрекеттер
E. Итерациялық

26. Программаның жұмысын жөндеу үшін менюдің қай командасы қолданылады?
A.Debug
B.Build
C.Option
D.Compile
E.Make

27. Редакторға соңғы тиелген программаның компиляциясын орындайтын нұсқау?
A.Compile
B.Build
C.Option
D.Debug
E.Make

28. Келесі циклдардың қайсысы “әзірше“?
A.While…
B.While.. Do …
C.For (i=1; i<=24; i++)
D.For (N=9; N<=4; i--)
E.Repeat…Do.

29. Массив элементтері бір типтес болуы шарт па?
A.ия
B.жоқ
C.олар әр жағдайда әр түрлі
D.массивтің өлшеміне байланысты
E.массивтің жазылуына байланысты

30. Компьютерге түсінікті тілде жазылған алгоритм …
A.программа

B.шама
C.өрнек
D.айнымалы
E.тұрақты

II-нұсқа.

1. While true қандай әрекеттердің орындалуын талап етеді?
A.шексіз цикл
B.1 рет орындалады да бітеді
C.орындалмайды
D.Си-де жазылуы дұрыс емес
E.кем дегенде 1 рет орындалады

2. Кітапханалық функциялар қайда орналасқан?
A.кітапханалық файлдарда
B.Си ішінде
C.компьютерде
D.Интернетте
E.жадыда

3. Символдық типтегі шамаларға компилятор жадында қанша байт орын бөлінеді?
A.1
B.11
C.10
D.8
E.27

4. Келесі циклнешереторындалады: for (i=i; i<=i; i++) ?
A.1 рет
B.0 рет
C.2
D.3
E.4

5. Келесі циклнешереторындалады: for (i=1; i<=100; i++)?
A.100 рет
B.256 рет
C.121
D.196
E.124

6. Идентификатор дегеніміз не?

A.программадағы объектінің аты
B.динамикалық жады
C.жиыннан тұратын массив
D.программаның берілу жолы
E.компиляторға арналған жолдар

7. #include <функция аты> не үшін қолданылады?
A.Программаға стандартты библиотекаларды қосу үшін қолданады;
B.Макростарды анықтау үшін қолданады;
C.Макростарды жою үшін қолданады;
D.Компиляциялауға дейін орындалатын программа;
E.Компиляциялаудан кейін орындалатын программа;

8. "= =" операциясы қандай операция?
A.логикалық "тең болса " операциясы;
B.логикалық "және" операциясы
C.логикалық терістеу операциясы;
D.тең операциясы;
E.теңсіздікті салыстыру;

9. "!=" операциясы қандай операция?
A.логикалық терістеу операциясы;
B.логикалық "және" операциясы
C.логикалық "тең" операциясы;
D.теңдікті салыстыру операциясы;
E.теңсіздікті салыстыру;

10."&&" операциясы қандай операция?
A.Логикалық "және" операциясы;
B.Теңсіздікті салыстыру;
C.Логикалық "тең" операциясы;
D.Теңдікті салыстыру операциясы;
E.Логикалық терістеу лоперациясы

11. "||" операциясы қандай операция?
A.логикалық "немесе" операциясы;
B.теңдікті салыстыру операциясы;
C.теңсіздікті салыстыру;
D.логикалық "және" операциясы;
E.логикалық "тең" операциясы

12. C++ тілінде келесі формауланы жаз: $\sqrt{x^8 + 8^x}$
A.sqrt(pow(x,8)+ pow(8,x))
B.pow(x)*8+pow(8,x)

C.pow(8,x)+pow(x,8)
D.(pow(x,8)+pow(8,1/2)
E.pow(pow(x,8)+pow(8,x),2)

13. const түйіндісөзі ... болыптабылады
A.константа
B.идентификатор
C.процедура
D.директива
E.функция

14. C++ та тұрақтыларды баяндау үшін қандай түйінді сөз қолданылады
A.const
B.int
C.var
D.procedure
E.function

16.Өзін-өзі шақыратын функция
A.рекурсия
B.процедура
C.функция
D.файл
E.директива

17. C++ та кез келген программаның орындалуы қай функциядан басталады
A.main()
B. random
C.randonize()
D.clrscr()
E.әр программада әр түрлі

18. Қандай да бір амалды айқындайтын операциялар мен сипаттамаларды айқындалған тізбегін ... деп атайды.
A.функция
B.процедура
C.рекурсия
D.директива
E.файл

19. Мәліметтерді форматталған түрде енгізу функциясын таңдаңыз
A.scanf()
B.printf()
C.getchar()

D.putchar()
E.puts()

20. Көлденеңтабуляциясимволы
A. \t
B. \v
C. \n
D.\\
E. \b

21. Iostream.h стандарттықітапханасыныңдеректердіенгізуфункциясы:
A.cin
B.cout
C. skanf
D. print
E.printf

22. Ағым қандай функцияның көмегімен жабылады
A.Fclose
B.feof
C.ferror
D.return
E.atoi

23. Файлға жолды жазу функциясын көрсетіңіз
A.Fputs()
B.fred()
C.puts()
D.printf()
E.fgets()

24. Жолды бүтін санға өзгерту функциясын таңдаңыз
A.Atoi()
B.atof()
C.strtod()
D.strchr()
E.strrev()

25. Жолдарды көшіру функциясын таңдаңыз
A.strcpy()
B.strcat()
C.strcmp()
D.strlen()
E.strtod()

26. Жолды нақты санға өзгерту функциясын таңдаңыз
A.Atof()
B.strstr()
C.strset()
D.strrev()
E.atoi()

27. Келесі түйінді сөздердің қайсысы енгізу операторы болып табылады
A.cin
B.cout
C.float
D.char
E.printf

28. 10 нақты саннан тұратын массивті сипаттау
A.int a[10];
B.cin a[10]
C.float a(10)
D.float a[1..10]
E.int a[1..10]

29. Стандарттыбиблиотеканыңобъектілеріқайатаукеңістігіндеанықталған
A.std
B.com
C.dio
D.cst
E.cpp

30. Келесі циклнешереторындалады: for (i=i; i<=i; i++) ?
A.1 рет
B.0 рет
C.2
D.3
E. 4

III-нұсқа.

1. Келесі циклнешереторындалады: for (i=1; i<=100; i++)?
A.100 рет
B.256 рет
C.121
D.196
E.124

2. Идентификатор дегеніміз не?
A.программадағы объектінің аты
B.динамикалық жады
C.жиыннан тұратын массив
D.программаның берілу жолы
E.компиляторға арналған жолдар

3. #include <функция аты> не үшін қолданылады?
A.Программаға стандартты библиотекаларды қосу үшін қолданады;
B.Макростарды анықтау үшін қолданады;
C.Макростарды жою үшін қолданады;
D.Компиляциялауға дейін орындалатын программа;
E.Компиляциялаудан кейін орындалатын программа;

3. "= =" операциясы қандай операция?
A.логикалық "тең болса " операциясы;
B.логикалық "және" операциясы
C.логикалық терістеу операциясы;
D.тең операциясы;
E.теңсіздікті салыстыру;

4. Екі мәнді теңділікке салыстыру үшін қандай операторды қолдануға болады?
A.==
B.=
C.&&
D.||
E.>>

5. децикл операторының неше типі бар?
A.3
B.1
C.2
D.4
E.9

6. Төмендегі сипатталған массивте қанша элемент бар? int a[n][m]
A.m * N
B.m - n
C.m + n
D.m : n
E.0

7. int a[10] жазуы нені білдіреді:
A.массивті хабарлайды
B.айнымалыны хабарлайды
C.функцияны хабарлайды
D.құрылымды хабарлайды
E.дұрыс жауап жоқ

8. Массив - бұл
A.Бір типті элементтер жиынтығы
B.кез-келген элементтер жиынтығы
C.әртүрлі элементтер жиынтығы
D.айнымалылыр тізбегі
E.ережелер тізбегі

9. int a[] = {3,5,8,0,2} жазуы нені білдіреді?
A.Массив элементтері ненгізеді
B.құрылым элементтерін енгізеді
C.жиын элементтерін енгізеді
D.цикл бастамасы
E.функция бастамасы

10. C++ тілінде ең басты функция қандай?
A.main()
B.class()
C.result()
D.private()
E.global()

11. #include дегеніміз не?
A.Препроцессор директивасы
B.Функция
C.Библиотека
D.Оператор
E.Класс

12. Бағдарламада пікір қалдыру үшін қандай таңба қолданамыз?
A.// және /* */
B.;
C.:
D.&&
E.||

12. using namespace std; бағдарламаға не еңгізеді?
A.стандартты есімдер кеңістігін

B.стандартты функция
C.стандартты библиотека
D.арнайы класс
E.арнайы объект

13. int age; бұл бағдарламадағы age айнымалысы қандай дерек түрін сақтайды?
A.нақты сандар
B.таңба
C.жолдама
D.функция
E.класс

14. Программаның келесі үзіндісі орындалу барысындағы нәтижесі қандай: if 0= true then ClrScr?
A.Экран тазармайды
B.Экран тазарылады
C.Экран тисі өзгереді
D.Паскальде жазылуы дұрыс емес
E.Экран түрлі-түсті болады

15. Процедуралар мен функциялар бөлімінде тек қана стандарттық процедуралар жазылады ма?
A.Стандарттық та пайдаланушының да
B.Ия, тек стандарттық
C.Жоқ, тек пайдаланушының
D.Программада бұл бөлім жоқ
E.Процедуралар жазылады

16. Жай құрылымдық емес берілгендер және күрделі типті берілгендер қалай аталады?
A.құрылымдық
B.структуралық
C.таблицалық
D.аналитикалық
E.жолдық

17. Тек бір ғана емес бірнеше есептер класын шешуге бағытталған алгоритм қасиеті:
A.жалпылық
B.детерминистілік

C.анықтылық
D.дискреттілік
E.түсініктілік

18. Компьютерге түсінікті тілде жазылған алгоритм …
A.программа
B.шама
C.өрнек
D.айнымалы
E.тұрақты

19. Алгоритмдік тілде қызметші сөздер деп нені айтады?
A.Мағынасы мен пайдалану әдісі бір рет және әрқашанға берілген сөздер
B.Командалар жазу үшін қолданылатын сөздер
C.Басқа алгоритмдер құрамында қолданылатын көмекші алгоритмдер
D.Тұрақты мәні бар константалар
E.Константалар

20. StringVar = ‘ABCD’: Delete(StringVar, 1, Length(StringVar)) орындалунәтижесі?
A.StringVar = «»
B.StringVar =’A’
C.StringVar = ‘ABCD’
D.Паскальдежазылуыдұрысемес
E.StringVar = «B»

21. COPY(‘ABCDEFGHIJ’, 4, 3)= орындалунәтижесі?
A.‘DEF’
B.ABCD’
C.CDEF’
D.ABC’
E.GHI’

22. POS(‘DE’,’ABCDEFGH’) = орындалунәтижесі?
A.4
B.3
C.5
D.6
E.2

23. Str(25, StringVar); орындалунәтижесі?
A.StringVar =’25’{1 пробел}
B.StringVar =’25. 3

C.StringVar =’25’
D.Паскальдажазуыдұрысемес
E.StringVar =’250’

24. 16 div 3 орындалу нәтижесі келесі?
A.5
B.4
C.3
D.2
E.1

25. 16 mod 3 орындалу нәтижесі келесі?
A.1
B.3
C.7
D.4
E.5

26. Массив элементтері бір типтес болуы шарт па?
A.ия
B.жоқ
C.олар әр жағдайда әр түрлі
D.массивтің өлшеміне байланысты
E.массивтің жазылуына байланысты

27. Массив элементтері бір типтес болуы мүмкін бе?
A.Ия
B.Жоқ
C.Массивтің өлшеміне байланысты
D.Компиляторға байланысты
E.олар әр жағдайда әр түрлі

28. Массивтің неше өлшемді болуы мүмкін?
A.Тек машина жадының көлемімен шектеледі
B.2
C.4
D.1
E.5

29. Қандай типті мәліметтер Write операторымен экранға шығарылмайды?
A.set
B.string
C.char
D.real

E.byte

30. Паскальдафонның түсінбояуүшіннешетүрлі түсбар ?
A.0-7
B.0-15
C.1-32
D.1-4
E.0-3

IV-нұсқа.

1. Дыбыс неше уакытта созылуы мүмкін?
A.Шексіз
B.1 минут
C.1 микросекунд
D.256 секунд
E.10 минут

2. Паскаль-программадабірмезгілденешедыбысшығаруғаболады?
A.1
B.3
C.2
D.4
E.0

2. Жоғары дәрежелі шешімділікте неше түс рұқсат етілген
A.2
B.6
C.4
D.8
E.3

3. Graph Color Mode режимында экранда неше нүкте бар?
A.64000
B.32000
C.10000
D.100000
E.24000

4. Егерх=1, ондау-ке 2 мәнін меншікте, ал егер х=2 тең болса у-ке 1 мәнін меншікте
A.if (x=1) y=2; else if (x=2) y=1
B.if (x=1) y=2 if (x=2) y=1
C.if (x=1) y=2; else y=1

D.if (x=1) y=1; else y=2
E.if (x=1) y=1; else y=3

5. Егер x=1, онда y-ке 2 мәнін меншікте, әйтпесе y-ке 1 мәнін меншікте:
A.if (x=1) y=2; else y=1
B.if (x=1) y=2; else if (x=2) y=1
C.if (x=1) y=2 else if (x=2) y=1
D.if (x=1) y=2
E.if (x=1) y=1; else y=3

6. Келесі циклдардың қайсысы «Дейін?»
A.Do…While
B.While….
C.For (i:=1; i<=24; i++)
D.For (N=9; N<=4; i--)
E.Repeat…Do.

7. Келесі циклдардың қайсысы “әзірше“?
A.While…
B.While.. Do …
C.For (i=1; i<=24; i++)
D.For (N=9; N<=4; i--)
E.Repeat…Do.

8. Бүкіл программа бір оператор сияқты бір-ақ кадамда орындалу үшін трассировканы қалай орындауға болады?
A.Step Over
B.Go To Cursor
C.Tracc Into
D.User Screen
E.Run

9. Орындалып жатқан программаның орындалу қорытындысын бейнелейтін экранын қалай көруге болады?
A.User Screen
B.Go To Cursor
C.Tracc Into
D.Step Over
E.Run

10. Редакторға соңғы тиелген программаның компиляциясын орындайтын нұсқау?
A.Compile
B.Build

C.Option
D.Debug
E.Make

11. Программаны құрайтын барлық файлдардың трансляциясын қай нұскау орындайды?
A.Build
B.Compile
C.Option
D.Debug
E.Make

12. Программаның жұмыс істеу режимін білдіретін кілт?
A.Option
B.Build
C.Compile
D.Debug
E.Make

13. Программаның жұмысын жөнге салу үшін менюдің қай командасы қолданылады?
A.Debug
B.Build
C.Option
D.Compile
E.Make

14. Тізбектей құрылым деп… айтамыз.
A.Бірінен соң бірі орындалатын әрекеттерді
B.Шарттардың бірі орындалуын қамтамасыз ететін әрекеттер
C.Кейбір командалардың бірнеше рет орындалуын
D.Шартсыз өту әрекеттерін
E.цикл

15. Тармақталу құрылымы деп… айтамыз.
A.Шарттардың бірі орындалуын қамтамасыз ететін әрекеттер
B.Бірінен соң бірі орындалатын әрекеттерді
C.Кейбір командалардың бірнеше рет орындалуын
D.Шартсыз өту әрекеттерін
E.Итерациялық әрекеттер

16. Цикл деп… айтамыз.
A.Кейбір командалардың бірнеше рет орындалуын
B.Шартсыз өту әрекеттерін

C.Бірінен соң бірі о рындалатын әрекеттерді
D.Шарттардың бірі орындалуын қамтамасыз ететін әрекеттер
E.Итерациялық

17. Модуль-бұл….
A.Логика түрінде байланысқан операциялар тізбегі
B.Программаның бөлігі
C.Бір-бірімен байланысқан алгоритмдер тізбегі
D.Базалық алгоритмдер құрылымының суперпозициясы
E. Ережелер жиынтығы

18. Декомпозиция деп нені айтамыз?
A.Мазмұны жағынан басты есепке бағынатындай есептерге бөлу
B.Дайын кітапхананы пайдалану
C.Программаны біртұтас етіп біріктіру
D.Алгоритмдік құрылым
C.Программаны талдау

19. Қандай мәліметтер құрылымды деп аталады?
A.Қарапайым мәліметтер комбинациясы көмегімен алынған мәліметтер
B.Мәліметтерді өңдерудің процедурасынан құралатын мәліметтер
C.Жиындарды сызықтық тізімдерді
D.Объектілерді
E.Процедуралар

20. Программалауда қандай қатынас құрылымды деп аталады?
A.үш базалық құрылым комбинацияларына бағытталған қатынас
B.операцияларды компьютермен орындауға бағытталған қатынас
C.операторлардың жеке топтары көмекші алгоритмдерге біріктіру
D.ережелер жиынтығымен суреттелу
E.алгоритм анық болса

21. Өрнектің есептеу нәтижесінің қайсысы қате?
A. 3 / 7 = 3
B. 10 % 7 = 3
C. 1 / 5 = 0
D. 5 % 3 = 2
E. 4 % 6 = 4

22. % операциясы қандай типке қолданылады
A. бүтін сандарға
B. символдарға
C. кез келген сандарға
D. нақты сандарға

E.Мұндай операция жоқ

23. Келесі тұжырымның қайсысы қате?
A. X - = a+b эквивалентті x = -(a+b)
B. A + = 2 эквивалентті a = a + 2
C. M * = n эквивалентті m=m*n
D. R % = 5 эквивалентті r=r%5
E. P / =10 эквивалентті p=p/10

24. Экранға келесі программа не шығарады, егер 5,4,3 сандары енгізілсе?
{int a,b;
cin>>a>>b>>a;
cout<<a<<b<<a; }
A. 3 4 3
B. 4 3 5
C. 3 5 5
D. 3 4 5
E. 5 4 3

25. s=0; i= 0;
while (i<5) {i++; s=s+i;} операторлары орындалғаннан кейін s айнымалысының мәнін анықта
A. 15
B. 1/5
C. 10
D. 0
E. ¼

26. Шартты операторлардың құрылымы келесі түрде болады:
A. if (өрнек) оператор1; else оператор2;
B. switch (бүтін санды өрнек)
{ caseконстанта1: операторлар тізімі;
……..
default:операторлар тізімі; }
C. dooператорwhile (өрнек);
D. while (өрнек) оператор;
E. for (өрнек _1; өрнек _2; өрнек _3) оператор;

27. # include <iostream.h>
void main ( )
{ int s=1, i, f=1;
for (i=2; i<=3; i++) {f=f*i; s=s+f;}
cout<<”s=”<<s; }
программасы экранға қандай нәтиже шығарады

A. s=9
B. s=18
C. s=1
D. s=4
E. s=5

28. Массивтің кез келген компонентін…
A. Тікелей қолдануға болады
B. жауаптардың барлығы дұрыс
C. Тікелей және тізбектей қолдануға болады
D. тізбектей қолдануға болады
E. жауаптардың барлығы қате

29. Берілген программа фрагменті a [7][7] матрицасы үшін не есептейді?

```
sum = 0;
        for  (i= 1; i<=7; i++)
        for  (j= 1; j<=7; j++)
             sum = sum + a[i,j];
     cout<<"sum="<<sum;
```

A. матрица элементтерінің қосындысын
B. әр бір бағанның қосындысын
C. әр бір жолдың көбейтіндісін
D. әр бір жолдың қосындысын
E. әр бір бағанның көбетіндісін

30. Сілтеме -бұл …
A. программа объектісі орналасатын жады өрісінің адресі
B. жады облысында үздіксіз орналасқан бір типті элементтер құрылымы
C. Си-дегі негізгі программалық бірлік
D. жады фрагменті
E. жады участогы

V-нұсқа.

1. Айнымалы –бұл …
A. белгілі бір типті мәліметтер сақталатын ат берілген жады обылысы
B. программалық объекті аты
C. константалардан, шамалардан, операция белгілерінен, функциялардан, жақшалардан тұратын құрылым
D. операндаларға қолданылатын әрекеттерді анықтайтын бір немесе одан көп символдар
E. өзгермейтін шама

2. Келесі программа экранға не шығарады:

```
#include <iostream.h>
  void main ( )
{ int x, a;
a=(x=3, 5*x);     x++;
cout<<”x=”<<x;
cout<<”a=”<<a; }
```

+A:=5, B:= 4, A=B+4 программа фрагменті орындалғаннан кейін A және B айнымалыларының мәндері:

A. A= 8; B= 4;
B. A= 5; B=9;
C. A = 4; B = 8;
D. A =5; B= 4;
E. A=5;   B=8;

3. Шартты оператор кайжерде дұрыс жазылмаған вариантын таңдаңыз:

A. If (<оператор>) <Өрнек> Else <оператор>
B. If (<Өрнек>) <оператор>; Else <оператор>
C. If (<Өрнек>) {S1; S2; S3} Else S4
D. If(<Өрнек>) { S1; S2; S3}
E. If(<Өрнек>) {S1; S2; } Else S3; S4

4. Берілген X пенY үлкенін табу керек болса, дұрыс жауабын табыңыз:

A. If (X>Y) Max=X; else Max=Y; printf(“Max=%d”, Max)
B. If (X>Y) Max:=X Else Max:=Y
C. If (X>Y) Max:=X; If (X<=Y) Max:=Y
D. Барлығыдұрысемес
E. Max:= Y; If (X>Y) Max:=У; printf(“%d”, Max)

5. Бірінші 13 санның квадраттарының косындысын есептеңіз:

A. S= 0; For (i= 1; i<=13;i++) S=S+i*i; printf(“%d”, s)
B. S= 1; For (i= 1; i<=13; i++) S= S* i; printf(“%d”, s)
C. scanf(“%d”,&n); s= n*(n+1)/2; printf(“%d”, s)
D. scanf(“%d”,&n); s= (n*(n+1); printf(“%d”, s)
E. S= 1; For (i= 11; i<=13; i++) S; printf(“%d”, s)

6. Бірінші 13 санның көбейтіндісін есептеңіз:

A. P= 1; For (i=1; i<=13;i++) P= P*i; printf(“%d”, P)
B. P=0; For (i=1; i<=13; i++) P=P+i* i; printf(“%d”, P)
C. scanf(“%d”,&n); s= n*(n+1)/2; printf(“%d”, P)
D. scanf(“%d”,&n); s= sqr(n*(n+1)/2; printf(“%d”, P)
E. scanf(“%d”,&n);For(i=1; i<=13; i++) P= P*i; printf(“%d”, P)

7. Төмендегі операторлардың қайсысы экранға 14 және 7 шығарады?

A. Number= 7; printf(“%d”,2*Number); printf(“%d”,Number)

B. Number= 7; Number=2*number; printf(“%d”,Number)
C. Number= 7; printf(“%d”,Number); number=Number;
D. Number= 7; number=number+1; printf(“%d”,Number)
E. Number= 7; printf(“%d”,Number); Number= 2; printf(“%d”,Number)

7. Төмендегі операторлардың қайсысы экранға 14 шығарады?
A. Number= 7; Number=2*Number; printf(“%d”,Number)
B. Number= 7; printf(“%d”,2*Number); printf(“%d”,Number)
C. Number= 7; writeln(number); number:=number+1;
D. Number= 7; number=number+1; printf(“%d”,Number)
E. Number= 7; number=number+5; printf(“%d”,Number)

8. Төмендегі операторлардың қайсысы экранға 7 шығарады?
A. Number= 7; printf(“%d”,Number); Number:=Number+1
B. Number= 7; printf(“%d”,2*Number); printf(“%d”,Number)
C. Number= 7; Number=Number+1; printf(“%d”,Number)
D. Number= 7; Number=2*Number; printf(“%d”,Number)
E. Number= 7;printf(“%d”,7*Number); printf(“%d”,Number)

9. Төмендегі операторлардың қайсысы экранға 8 шығарады?
A. Number= 7; Number=Number+1; printf(“%d”,Number)
B. Number= 7; printf(“%d”,2*Number); printf(“%d”,Number)
C. Number= 7; Number=2*Number+1; printf(“%d”,Number)
D. Number= 7; Number=2*Number; printf(“%d”,Number)
E. Number= 8; printf(“%d”,Number); printf(“%d”,Number)

10. Байтқа қанша ақпарат сыяды?
A.8 бит
B.Сандар жиынтығы
C.16 бит
D.Цифралар жиынтығы
E.32 бит

11. Компьютер - бұл:
A. Ақпаратты жинау, өңдеу, сақтау құралы
B. Ойын автоматы
C. Сұхбаттасу құралы
D. Оқыту құралы
E. Есептеуіш

12. Мына өрнекті $\sum_{k=1}^{n} x + ky$ есептеу үшін WHILE циклының дұрыс сипаттамасын таңда
A. k=1 while (k<=n) { z=x+k*y; k++; }
B. k=1 while (k<n) { z=x+k*y; k++;}

C. k=1 while (k<=n) { z=x+y; }
D. k=1 while (k<n) z=x+k*y;
E. k=1 while (k<=n) z=x+k*y; k++;

13. ε дәлдікпен $\sum_{k=1}^{\infty} \frac{(-1)^k x^k}{3^k + 2^k}$ қосындының дұрысын таңда.
A. k=1 do { ak=pow(-1,k)*pow(x,k)/(pow(3,k) +pow(2,k)); k++; s+=ak; } while (abs(ak)>ε);
B. k=1 do { ak=pow(-1,k)*pow(x,k)/(pow(3,k)+ pow(2,k)); k++; s+=ak; } while (abs(ak)<ε);
C. k=1 do { ak=pow(-1,k)*pow(x,k)/(pow(3,k) +pow(2,k)); k++; s+=ak; } while (abs(s)<ε);
D. k=1 do { ak=pow(-1,k)*pow(x,k)/pow(3,k) +pow(2,k); k++; } s+=ak; while (abs(s)>ε);
E. k=1 do ak=pow(-1,k)*pow(x,k)/(pow(3,k)+ pow(2,k)); k++; s+=ak; while (abs(ak)>ε);

14. Бөлгендегі қалдық табу опреациясы қалай беріледі?
A. %
B. /
C. *
D. <
E. &

15. Қайсысы унитарлық операцияларға жатады?
A. @, NOT
B. *, /, DIV, MOD, AND, SHL, SHR
C. +,-, OR, XOR
D. =, <>, <, >, <=, >=, IN
E. DIV, @, IN; +,-

16. Қайсысы көбейту типтес операцияларға жатады?
A. *, /, DIV, MOD, AND, SHL, SHR
B. @, NOT
C. +,-, OR, XOR
D. =, <>, <, >, <=, >=, IN
E. @, NOT, =, <>, <, >

17. Қайсысы қосу типтес операцияларға жатады?
A. +,-, OR, XOR
B. @, NOT
C. *, /, DIV, MOD, AND, SHL, SHR
D. =, <>, <, >, <=, >=, IN
E. MOD, AND, @, NOT

18. Қайсысы қатынас типтес операцияларға жатады?
A. =, <>, <, >, <=, >=, IN
B. @, NOT
C. *, /, DIV, MOD, AND, SHL, SHR
D. +,-, OR, XOR
E. OR, <=, >=, SHL, SHR

19. Типтерді өзгертетін функциялар қайсысы?
A. CHR(x), ORD(X), Round(x),Trunc(x)
B. nteger, boolean, char, реттелгентип, real
C. Idd(x), Pred(x), Succ(x)
D. integer, file, record, array, real
E. integer, Pred(x), Succ(x)

20. Арифметикалық функциялар қайсысы?
A. ABS(X), SQRT(X), INT(X), SQR(X)
B. CHR(x), ORD(X), Round(x),Trunc(x)
C. Idd(x), Pred(x), Succ(x)
D. integer, file, record, array, real
E. file, record, Round(x),Trunc(x)

21. Төмендегі операциялардың қайсысы арифметикалық болып табылады?
A. +,- , * , / , DIV, Mod
B. @, NOT, OR, XOR
C. Меншіктеу, процедураны шакыру, оту, бос оператор
D. +, - ,*, and, or
E. +, - ,*, OR, XOR

22. Төмендегі операциялардың қайсысы логикалық болып табылады?
A. NOT, OR, XOR, AND
B. @,+,- , * , / , DIV, Mod
C. Меншіктеу, процедуранышакыру, өту, босоператор;
D. +, - ,*, and, or
E. AND, or +, - ,*

23. Экранға келесі программа не шығарады, егер 5,4,3 сандары енгізілсе?
{int a,b;
cin>>a>>b>>a;
cout<<a<<b<<a; }
A. 3 4 3
B. 4 3 5
C. 3 5 5
D. 3 4 5

E. 5 4 3

24. Өрнектің есептеу нәтижесінің қайсысы қате?
A. 3 / 7 = 3
B. 10 % 7 = 3
C. 1 / 5 = 0
D. 5 % 3 = 2
E. 4 % 6 = 4

25. Цикл деп… айтамыз.
A.командалардың бірнеше рет орындалуын
B.Шартсыз өту әрекеттерін
C.Бірінен соң бірі о рындалатын әрекеттерді
D.Шарттардың бірі орындалуын қамтамасыз ететін әрекеттер
E. Итерациялық

26. Программаның жұмысын жөндеу үшін менюдің қай командасы қолданылады?
A.Debug
B.Build
C.Option
D.Compile
E.Make

27. Редакторға соңғы тиелген программаның компиляциясын орындайтын нұсқау?
A.Compile
B.Build
C.Option
D.Debug
E.Make

28. Келесі циклдардың қайсысы “әзірше“?
A.While…
B.While.. Do …
C.For (i=1; i<=24; i++)
D.For (N=9; N<=4; i--)
E.Repeat…Do.

29. Массив элементтері бір типтес болуы шарт па?
A.ия
B.жоқ
C.олар әр жағдайда әр түрлі
D.массивтің өлшеміне байланысты

E.массивтің жазылуына байланысты

30. Компьютерге түсінікті тілде жазылған алгоритм …
A.программа
B.шама
C.өрнек
D.айнымалы
E.тұрақты

Пайдаланылған әдебиеттер тізімі

1. Бөрібаев Б «Программалау технологиясы С/С++» Алматы 2011
2. Альфред, В. Ахо Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - Москва: Высшая школа, 2015. - 882 с.
3. Балена, Франческо Современная практика программирования на Microsoft Visual Basic и Visual C# / Франческо Балена , Джузеппе Димауро. - М.: Русская Редакция, 2015. - 640 с.
4. Боровский, А. C++ и Pascal в Kylix 3. Разработка интернет-приложений и СУБД / А. Боровский. - М.: БХВ-Петербург, 2015. - 544 с.
5. Давыдов, В. Visual C++. Разработка Windows-приложений с помощью MFC и API-функций / В. Давыдов. - М.: БХВ-Петербург, 2014. - 576 с.
6. Зиборов, В. MS Visual C++ 2010 в среде .NET / В. Зиборов. - М.: Питер, 2012. - 320 с.
7. Кетков, Юлий Практика программирования: Visual Basic, C++ Builder, Delphi. Самоучитель (+ дискета) / Юлий Кетков , Александр Кетков. - М.: БХВ-Петербург, 2012. - 464 с.
8. Мешков, А. Visual C++ и MFC / А. Мешков, Ю. Тихомиров. - М.: БХВ-Петербург, 2013. - 546 с.
9. Неформальное введение в C++ и Turbo Vision. - Москва: ИЛ, 2010. - 384 с.
10. Панюкова, Т. А. Языки и методы программирования. Создание простых GUI-приложений с помощью Visual C++. Учебное пособие / Т.А. Панюкова, А.В. Панюков. - Москва: Мир, 2015. - 144 с.
11. Пахомов, Б. C/C++ и MS Visual C++ 2010 для начинающих / Б. Пахомов. - М.: БХВ-Петербург, 2011. - 736 с.
12. Пахомов, Борис C/C++ и MS Visual C++ 2012 для начинающих / Борис Пахомов. - Москва: СИНТЕГ, 2015. - 518 с.
13. Пахомов, Борис C/C++ и MS Visual C++ 2012 для начинающих / Борис Пахомов. - М.: "БХВ-Петербург", 2013. - 502 с.
14. Полубенцева, М. C/C++. Процедурное программирование / М. Полубенцева. - М.: БХВ-Петербург, 2014. - 448 с.
15. Поляков, А. Методы и алгоритмы компьютерной графики в примерах на Visual C++ / А. Поляков, В. Брусенцев. - М.: БХВ-Петербург, 2011. - 560 с.
16. Понамарев, В. Программирование на C++/C# в Visual Studio .NET 2003 / В. Понамарев. - М.: БХВ-Петербург, 2015. - 917 с.
17. Роберт, С. Сикорд Безопасное программирование на C и C++ / Роберт С. Сикорд. - Москва: РГГУ, 2014. - 496 с.
18. Секунов, Н. Программирование на C++ в Linux / Н. Секунов. - М.: БХВ-Петербург, 2016. - 425 с.