**24 ОАП Лк7 Оператор выбора**

К операторам выбора, называемым операторами управления потоком выполнения программы, относят: условный оператор (*if*…*else*) и *переключатель* (*switch*). Каждый из них служит для выбора пути выполнения программы.

*Синтаксис* условного оператора следующий:

if (выражение) оператор\_1; else оператор\_2;

Здесь нужно пояснить следующее. Выражения, используемые при сравнении двух величин в С++ отличаются от таковых, в других алгоритмических языках.

|  |  |
| --- | --- |
| **Описание** | **Выражение** |
| a больше, чем b | a>b |
| a меньше, чем b | a<b |
| a равно b | a==b |
| a не равно b | a!=b |
| a больше или равно b | a>=b |
| a меньше или равно b | a<=b |

Так при проверке равенства двух величин нужно использовать ==, а не =, т.к. в противном случае получится *оператор присваивания*.

При истинности выражения, выполняется оператор\_1, в противном случае - выполняется оператор\_2. Рассмотрим пример.

x=3;

if (x>0) a=15; else a=12;

Как вы, наверное уже догадались, *переменная* a принимает *значение* 15.

*Операторы* могут быть составными (могут объединяться в блоки).

double y=2;

if (y>3) {cout<<"\nHow old are you?";

cout<<"\nI am 33"<<endl;}

else {cout<<"\nWhat are you doing?";

cout<<"\nI am computing"<<endl;};

Результатом работы этого фрагмента будет выведенная на экран фраза

What are you doing?

I am computing

Допустима сокращенная форма условного оператора, когда отсутствует *else* и оператор\_2. В этом случае при ложности выражения ничего не происходит.

double z=5;

if (z<3) cout<<"\nHi!";

Данный фрагмент ничего не выполнит.

Нужно сказать, что в свою *очередь*, оператор\_1 и оператор\_2 могут быть условными, что позволяет организовать цепочку проверок. В следующем примере показаны некоторые способы использования условных операторов.

//==========================================================

// Name : choice.cpp

// Author : Marat

// Version :

// Copyright : Your copyright notice

// Description : Hello World in C++, Ansi-style

//==========================================================

#include <iostream>

using namespace std;

//Здесь рассматриваются операторы выбора

//В этой программе объясняется использование условных

//операторов

int main() {

int x=5;

int y=10;

cout<<"\nThis program explains 'if' statement using"<<endl;

cout<<"\nx= "<<x<<", "<<"y= "<<y<<endl;

if (x>y) {cout<<"\n"<<x<<" > "<<y<<" ";}

else {cout<<"\n"<<y<<" > "<<x<<" "<<endl;}

double a=3.3;

double b=2.48;

double c=4.65;

double d;

//Использование логической бинарной операции

//&& - конъюнкции (И)

if (a>b && c>b) {d=7;}

else d=3;

//Если а>b и c>b, то d=7, иначе, d=3

cout<<"\na= "<<a<<", "<<"b= "<<b<<", "<<"c= "<<c<<endl;

cout<<"\nBecause "<<a<<" > "<<b<<" && "<<c<<" > "<<b<<" So d = "<<d<<endl;

double f=1;

double g=3;

double h=5;

double k;

if (f>g || h>g) {k=15;}

else k=100;

//Использование логической бинарной операции

//|| - дизъюнкции (ИЛИ)

//Если f>g или h>g, то k=15, иначе, k=100

cout<<"\nf= "<<f<<", "<<"g= "<<g<<", "<<"h= "<<h<<endl;

cout<<"\nBecause "<<f<<" > "<<g<<" || "<<h<<" > "<<g<<" So k = "<<k<<endl;

//При выполнении многочисленных последовательных сравнений

//используется оператор if-else-if

int st=55;

double z;

//Введем фрагмент таблицы Стьюдента для 95% вероятности

if (st>120) z=1.96;//st больше 120

else if (st==120) z=1.98;//st равно 120

else if (st>=100) z=1.982;//st больше или равно 120

else if (st>=90) z=1.986;

else if (st>=80) z=1.989;

else if (st>=70) z=1.994;

else if (st>=60) z=2;

else if (st>=55) z=2.004;

else if (st>=50) z=2.008;

else if (st>=45) z=2.014;

else if (st>=40) z=2.021;

else cout<<"\nThats all right!"<<endl;//Действие по умолчанию

cout<<"\nt-test ("<<st<<") = "<<z<<endl;

return 0;

}

Результат:

This program explains 'if' statment using

x=5, y=10

10>5

a = 3.3, b = 2.48, c = 4.65

Because 3.3>2.48 && 4.65>2.48 So d = 7

f = 1, g = 3, h = 5

Because 1>3 || 5>3 So k = 15

t-test (55) = 2.004

В следующем примере *оператор if* - *else* используется при расчете геометрической прогрессии.

//==========================================================

// Name : geometrical\_progression.cpp

// Author : Marat

// Version :

// Copyright : Your copyright notice

// Description : Hello World in C++, Ansi-style

//==========================================================

#include <iostream>

#include<math.h>

//Здесь описываются математические функции (pow - возведение в степень)

//Эта программа рассчитывает сумму геометрической прогрессии

//S=A+A\*Q+A\*Q^2+A\*Q^3+...+A\*Q^N

using namespace std;

int main() {

//Опишем переменные, используемые в программе

double s; //Сумма геометрической прогрессии

double a; //Первый член прогрессии

double q; //Знаменатель прогрессии

int n; //Количество последних членов геометрической прогрессии

cout<<"\nHi! Let's calculate sum of the geometrical progression!";

cout<<"\nNow enter the first member's value ";

cin>>a;

cout<<"\nThen enter the base of geometrical progression ";

cin>>q;

cout<<"\nEnter the number of last members of geometrical progression ";

cin>>n;

//Используем условный оператор для выбора формулы рассчета суммы прогрессии

//в зависимости от того больше единицы основание прогрессии или нет

if (q<1) {s=a/(1-q);}

else

{

s=0;

for(int i=0;i<n;i++)

{

s+=pow(q,i);

}

s=a\*(s+1);

}

cout<<"\nThe sum of geometrical progression is "<<s<<endl;

return 0;

}

Результат:

Hi! Let's calculate sum of the geometrical progression!

Now enter the first member's value 3

Then enter the base of geometrical progression 2

Enter the number of last members of geometrical progression 6

The sum of geometrical progression is 192

**Переключатель**

***Переключатель*** используется для организации множественного выбора. Синтаксис оператора switch следующий.

switch (переключающее выражение)

{case константное\_выражение\_1: операторы\_1;

case константное\_выражение\_2: операторы\_2;

……………………………………………………

case константное\_выражение\_n: операторы\_n;

default: операторы;

}

Управляющая конструкция switch передает управление тому из помеченных с помощью case операторов, для которого значение константного выражения совпадает со значением переключающего выражения. Значения константных выражений, помещаемых за служебными словами case, приводятся к типу переключающего выражения. В одном переключателе все константные выражения должны иметь различные значения, но быть одного типа. Любой из операторов, помещенных в фигурных скобках после конструкции switch (...), может быть помечен одной или несколькими метками вида

case константное\_выражение:

Если значение переключающего выражения не совпадает ни с одним из константных выражений, то выполняется переход к оператору, отмеченному меткой default:. В каждом переключателе должно быть не больше одной метки default, однако эта метка может и отсутствовать. В случае отсутствия метки default при несовпадении переключающего выражения ни с одним из константных выражений, помещаемых вслед за case, в переключателе не выполняется ни один из операторов.

Сами по себе метки case константное выражение\_i: и default: не изменяют последовательности выполнения операторов. Если не предусмотрены переходы или выход из переключателя, то в нем последовательно выполняются все операторы, начиная с той метки, на которую передано управление. Пример программы с переключателем:

//==========================================================

// Name : switch.cpp

// Author : Marat

// Version :

// Copyright : Your copyright notice

// Description : Hello World in C++, Ansi-style

//==========================================================

#include <iostream>

using namespace std;

int main() {

int n;

cout<<"\nInput integer value between 1 and 7"<<endl;

cin>>n;

switch (n)

{

case 1: cout<<"\nToday is Monday"; break;

case 2: cout<<"\nToday is Tuesday"; break;

case 3: cout<<"\nToday is Wednesday";break;

case 4: cout<<"\nToday is Thursday"; break;

case 5: cout<<"\nToday is Friday"; break;

case 6: cout<<"\nToday is Saturday"; break;

case 7: cout<<"\nToday is Sunday"; break;

default:cout<<"\nThis number is illegal!";

}

cout<<"\n"<<endl;

return 0;

}

Результат:

Input integer value between 1 and 7

5

Today is Friday

Кроме сказанного о возможностях переключателя, приведенная программа иллюстрирует действие оператора break. С его помощью выполняется выход из переключателя.

**Условная операция**

В отличие от унарных и бинарных операций ***условная операция*** используется с тремя операндами. В изображении условной операции два размещенных не подряд символа '?', и ':' и три операнда-выражения:

выражение\_1 ? выражение 2 : выражение 3

Первым вычисляется значение выражения\_1. Если оно истинно, т.е. не равно нулю, то вычисляется значение выражения\_2, которое становится результатом. Если при вычислении выражания\_1 получится 0, то в качестве результата берется значение выражения\_3. Пример:

х <0 ? p=4 : z=7;

Если выражение х<0 истинно, то переменная p принимает значение 4, в противном случае переменной z присваивается значение 7.

//==========================================================

// Name : conditional\_operation.cpp

// Author : Marat

// Version :

// Copyright : Your copyright notice

// Description : Hello World in C++, Ansi-style

//==========================================================

#include <iostream>

using namespace std;

int main() {

double s, t, r;

s=5.7;

t=-0.237;

s<t?r=6:r=10;

cout<<"\ns= "<<s<<" t= "<<t;

cout<<"\ns < t? r =6 : r =10 ";

cout<<"\nSo r = "<<r<<endl;

return 0;

}

Результат:

s=5.7 t=-0.237

s<t?r=6:r=10

So r = 10