

Лекция 3. Операторы

Условный оператор

`if (условие) оператор_1; else оператор_2;`

Применяется для организации ветвлений.

Если условие истинное, то выполняется оператор_1, иначе оператор_2.

Может использоваться сокращенная форма (без else).

Примеры:

```
if (x>y) x=x+y;
```

```
if (x<100) {x=-x; cout<<x;}
```

```
else {int i=2; x*=i; }
```

Метки, пустой оператор, оператор перехода

Метка – это идентификатор, расположенный слева от оператора и отделенный от него двоеточием.

Например,

`M: x+=15;`

Пустой оператор, изображается символом `;`.

Введен для того, чтобы поставить метку в любом месте программы.

Оператор безусловного перехода:

`goto идентификатор;`

где идентификатор – одна из меток программы.

Например:

`goto M;`

Переключатель

```
switch (выражение)
{
case константа_1: операторы_1;
case константа_2: операторы_2;
...
default: операторы;
}
```

Константы могут быть целыми или
символьными и должны отличаться.

Присутствие default не является обязательным.

Описание выполнения

Сначала вычисляется выражение, стоящее в скобках после ключевого слова `switch`, которое должно быть целым. Эта величина используется в качестве критерия для выбора одного из возможных вариантов. Ее значение сравнивается с константой операторов `case`. При несовпадении выполняется переход к следующему `case` и сравнивается его константа. В случае совпадения "константы `i`" выполняется "оператор `i`", а также все последующие операторы `case` и `default`, если среди операторов после соответствующей константы не было оператора `break`. Если совпадений не было и имеется оператор `default`, то выполняется стоящий за ним оператор, иначе выполнение программы продолжится с оператора, следующего за `switch`.

Пример использования switch

```
#include <iostream.h>
int main()
{ int k=2;
  switch(k)
  { case 0; cout<<"выбор 0\n"; break;
    case 1; cout<<"выбор 1\n"; break;
    case 2; cout<<"выбор 2\n"; break;
    case 3; cout<<"выбор 3\n"; break;
    default: cout<<"default\n";
  }
  return 0;
}
```

Цикл с предусловием

while (условие)

тело цикла

Тело цикла выполняется, пока *условие* не станет ложным (равным 0). Проверка условия осуществляется перед каждым выполнением тела цикла.

Условие может быть логическим или арифметическим выражением. Если в теле цикла два и более оператора, то их заключают в фигурные скобки.

Цикл с постусловием

do

тело цикла

while (условие);

Условие логическое или арифметическое. Цикл выполняется хотя бы один раз. После выполнения тела проверяется истинность *условия* и если оно ложно, то цикл завершается, иначе тело выполняется снова.

Цикл с параметром

for (выражение_1; условие; выражение_2)

тело цикла;

Вычисляется Выражение_1. Проверяется условие, которое определяет продолжается или завершается цикл.

Затем выполняется тело цикла.

Вычисляется выражение_2, определяющее изменения переменных и параметров. Вновь проверяется условие, и все повторяется. Цикл заканчивается, когда станет ложным условие. Если отсутствует условие, то считается, что оно истинно. Тело цикла может быть отдельным или составным оператором

Операторы `break` и `continue`

Оператор `break` прекращает выполнение цикла и передает управление следующему за циклом оператору, используется, если условие продолжения итераций нужно проверять в середине тела цикла.

Оператор `continue` позволяет прервать текущую итерацию в любой точке тела цикла и перейти к проверке условия, удобен, когда тело цикла содержит ветвления.

Пример. Сумма квадратов чисел от 1 до 100

I. С использованием цикла с предусловием

```
#include <iostream>
using namespace std;
int main()
{ int a=1;
  long s=0;
  while (a<=100)
  {s=s+a*a;
  a++;}
  cout<<"s="<<s;
  return 0;
}
```

II. С использованием цикла с постусловием

```
#include <iostream>
using namespace std;
int main()
{ int a=1;
  long s=0;
  do
  {s=s+a*a;
  a++;}
  while (a<=100);
  cout<<"s="<<s;
  return 0;
}
```

III. С использованием цикла с параметром

```
#include <iostream>
using namespace std;
int main()
{ int a;
  long s=0;
  for (a=1;a<=100;a++)
  s=s+a*a;
  cout<<"s="<<s;
  return 0;
}
```