Условная конструкция. Логические операции.

Условная инструкция if

Часто возникает ситуация, когда программа должна выбрать, какую операцию ей выполнить, в зависимости от определенного условия.

К примеру, мы вводим с клавиатуры целое число. Если оно положительное, то программа должна выполнить одно действие, иначе -- другое.

```
if (num < 0) { // Если введенное число меньше нуля.
            printf("Это число отрицательное");
}             else            {             //             иначе
   printf("Это число неотрицательное");

}
```

Условие всегда записывается в круглых скобках.

Если условие выполнится, то начнется выполнение всех команд, которые находятся между фигурными скобками, если оно окажется ложным, то начнется выполнение всех, которые находятся во вторых фигурных скобках. Если фигурные скобочки отсутствуют, то тогда блоком (последовательностью действий, которая будет выполнено, если условие выполнено или не выполнено) считается одна инструкция. Например, в приведенном выше коде программы фигурные скобочки можно опустить.

Ветка else может отсутствовать, это называется неполным ветвлением, или неполной условной инструкцией.

Операции сравнения чисел

В условных инструкциях, в инструкциях цикла, как правило используются сравнения вида x > 0 или a != b (числа a и b не равны), то есть некоторое логическое выражение. В таких выражениях как правило используются операции сравнения (равно, неравно, меньше, больше и т.д.).

В языке C подобные операции возвращают значение типа int, либо 0, что считается ложью, либо 1, которое означает истину. В языке C++ для этого есть специальный логический тип `bool`.

Переменные логического типа `bool` принимают два значения: `true` (истина) и `false` (ложь). Также любое целочисленное выражение можно трактовать, как логическое выражение, при этом нулевое целое число означает ложь, а ненулевое — истину. Таким образом, если вместо условия

написать `false` или `0`, то оно будет всегда ложно, если же указать `true`, `1` или любое ненулевое число, то условие будет истинно.

Как правило, в качестве проверяемого условия используется результат вычисления одного из следующих операторов сравнения:

<	Меньше — возвращает `true`, если первый операнд меньше второго.
>	Больше — возвращает `true`, если первый операнд больше второго.
<=	Меньше или равно.
>=	Больше или равно.
==	Равенство. Возвращает `true`, если два операнда равны.
!=	Неравенство. Возвращает `true`, если два операнда неравны.

Например, условие `(x * x < 1000)` означает “значение `x * x` меньше 1000”, а условие `(2 * x != y)` означает “удвоенное значение переменной `x` не равно значению переменной `y`”.

Будьте аккуратны: оператор `==` (два знака равенства) — это проверка на равенство двух выражений, а оператор `=` (один знак равенства) — это присваивание одной переменной значения выражения и использование его в условии оператора ветвления в большинстве случаев является ошибкой.

Рассмотрим эту типичную ошибку на следующем примере:

```
int a, b;
cin >> a >> b;
if (a = b)
{
    cout << "Числа равны" << endl;
}
else
{
    cout << "Числа не равны" << endl;
}
```

Здесь по ошибке вместо операции сравнения `==` использована операция присваивания `=`. Поэтому при любых значениях a и b переменной a будет присвоено значение переменной b, при проверке истинности выражения `a = b`. Но оператор присваивания еще и возвращает значение, поэтому если значение b было ненулевым (а это интерпретируется, как истина), то программа выведет строку "Числа равны", а если нулевым — то строку "Числа не равны". При этом значение переменной `a` может быть вообще любым.

Логические операции

Для конструирования составных логических выражений в языке С можно использовать логические операции - конъюнкцию, дизъюнкцию, отрицание.

Операции	Обозначение	Условие	Описание
И (конъюнкция)	&&	a == 2 && b > 4	Составное условие истинно, если истинны оба простых условия
ИЛИ (дизъюнкция)	||	a == 2 || b > 4	Составное условие истинно, если истинно, хотя бы одно из простых условий
НЕ (отрицание)	!	!(a == 5)	Условие истинно, внутренне условие ложно.