

Лабораторная работа №3

УСЛОВНЫЕ ОПЕРАТОРЫ ЯЗЫКА C

Цель работы: изучить особенности использования условных операторов `if` и `switch`.

Условные операторы `if` и `switch`

Для того чтобы иметь возможность реализовать логику в программе используются условные операторы. Умозрительно эти операторы можно представить в виде узловых пунктов, достигая которых программа делает выбор по какому из возможных направлений двигаться дальше. Например, требуется определить, содержит ли некоторая переменная `arg` положительное или отрицательное число и вывести соответствующее сообщение на экран. Для этого можно воспользоваться оператором `if` (если), который и выполняет подобные проверки.

В самом простом случае синтаксис данного оператора `if` следующий:

```
if (выражение)
    <оператор>
```

Если значение параметра «выражение» равно «истинно», выполняется оператор, иначе он пропускается программой. Следует отметить, что «выражение» является условным выражением, в котором выполняется проверка некоторого условия. В табл. 2 представлены варианты простых логических выражений оператора `if`.

Таблица 2. Простые логические выражения

<code>if(a < b)</code>	Истинно, если переменная <code>a</code> меньше переменной <code>b</code> и ложно в противном случае.
<code>if(a > b)</code>	Истинно, если переменная <code>a</code> больше переменной <code>b</code> и ложно в противном случае.
<code>if(a == b)</code>	Истинно, если переменная <code>a</code> равна переменной <code>b</code> и ложно в противном случае.
<code>if(a <= b)</code>	Истинно, если переменная <code>a</code> меньше либо равна переменной <code>b</code> и ложно в противном случае.
<code>if(a >= b)</code>	Истинно, если переменная <code>a</code> больше либо равна переменной <code>b</code> и ложно в противном случае.

<code>if(a != b)</code>	Истинно, если переменная <code>a</code> не равна переменной <code>b</code> и ложно в противном случае.
<code>if(a)</code>	Истинно, если переменная <code>a</code> не равна нулю, и ложно в противном случае.

Приведем пример использования оператора ветвления `if`. Следующая программа позволяет определять знак введенной переменной.

Листинг 3. Программа определения знака введенного числа.

```
#include <stdio.h>
int main()
{
    float x;
    printf("Введите число: ");
    scanf("%f", &x);
    if(x < 0)
        printf("Введенное число %f является отрицательным.\n", x);
    if(x >= 0)
        printf("Введенное число %f является неотрицательным.\n", x);

    return 0;
}
```

Анализ приведенного текста программы показывает, что два условных оператора можно заменить одним, используя конструкцию

```
if (выражение)
    <оператор1>
else
    <оператор2>
```

которая интерпретируется таким образом. Если «выражение» истинно, то выполняется «оператор1», иначе выполняется «оператор2».

В случаях, когда при выполнении какого-либо условия необходимо записать более одного оператора, необходимо использовать фигурные скобки, т.е. использовать конструкцию вида

```
if (выражение)
{
    <список операторов>
}
else
{
    <список операторов>
}
```

Следует отметить, что после ключевого слова `else` формально можно поставить еще один оператор условия `if`, в результате получим еще более гибкую конструкцию условных переходов:

```
if(выражение1) <оператор1>
else if(выражение2) <опреатор2>
else <оператор3>
```

До сих пор рассматривались простые условия типа $x < 0$. Вместе с тем оператор `if` позволяет реализовывать более сложные условные переходы. В языке C имеются три логические операции:

&& - логическое И
 || - логическое ИЛИ
 ! – логическое НЕТ

На основе этих трех логических операций можно сформировать более сложные условия. Например, если имеются три переменные `exp1`, `exp2` и `exp3`, то они могут составлять логические конструкции, представленные в табл. 3.

Таблица 3. Пример составных логических выражений

<code>if(exp1 > exp2 && exp2 < exp3)</code>	Истинно, если значение переменной <code>exp1</code> больше значения переменной <code>exp2</code> и значение переменной <code>exp2</code> меньше значения переменной <code>exp3</code> .
<code>if(exp1 <= exp2 exp1 >= exp3)</code>	Истинно, если значение переменной <code>exp1</code> меньше либо равно значения переменной <code>exp2</code> или значение переменной <code>exp2</code> больше либо равно значения переменной <code>exp3</code> .
<code>if(exp1 && exp2 && !exp3)</code>	Истинно, если истинное значение <code>exp1</code> и истинно значение <code>exp2</code> и ложно значение <code>exp3</code> .
<code>if(!exp1 !exp2 && exp3)</code>	Истинно, если ложно значение <code>exp1</code> или ложно значение <code>exp2</code> и истинно значение <code>exp3</code> .

Подобно операциям умножения и сложения в математике, логические операции И ИЛИ НЕТ, также имеют свои приоритеты. Самый высокий приоритет имеет операция НЕТ, т.е. такая операция выполняется в первую очередь. Более низкий приоритет у операции И, и наконец самый малый приоритет у операции ИЛИ.

Условная операция `if` облегчает написание программ, в которых необходимо производить выбор между небольшим числом возможных вариантов. Однако иногда в программе необходимо осуществить выбор одного

варианта из множества возможных. Формально для этого можно воспользоваться конструкцией if else if ... else. Однако во многих случаях оказывается более удобным применять оператор switch языка C. Синтаксис данного оператора следующий:

```
switch(переменная)
{
    case константа1:
        <операторы>
    case константа2:
        <операторы>
    ...
    default:
        <операторы>
}
```

Данный оператор последовательно проверяет на равенство переменной константам, стоящим после ключевого слова case. Если ни одна из констант не равна значению переменной, то выполняются операторы, находящиеся после слова default. Оператор switch имеет следующую особенность. Допустим, значение переменной равно значению константы1 и выполняются операторы, стоящие после первого ключевого слова case. После этого выполнение программы продолжится проверкой переменной на равенство константы2, что часто приводит к неоправданным затратам ресурсов ЭВМ. Во избежание такой ситуации следует использовать оператор break для перехода программы к следующему оператору после switch.

В листинге 4 представлен пример программирования условного оператора switch.

Листинг 4. Пример использования оператора switch.

```
#include <stdio.h>
int main()
{
    int x;
    printf("Введите число: ");
    scanf("%d", &x);
    switch(x)
    {
        case 1 : printf("Введено число 1\n"); break;
        case 2 : printf("Введено число 2\n"); break;
        default : printf("Введено другое число\n");
    }
    char ch;
    printf("Введите символ: ");
    scanf("%c", &ch);
    switch(ch)
    {
        case 'a' : printf("Введен символ a\n"); break;
```

```

        case 'b' : printf("Введен символ b\n"); break;
        default : printf("Введен другой символ\n");
    }
    return 0;
}

```

Данный пример демонстрирует два разных варианта использования оператора switch. В первом случае выполняется анализ введенной цифры, во втором – анализ введенного символа. Следует отметить, что данный оператор может производить выбор только на основании равенства своего аргумента одному из перечисленных значений case, т.е. проверка выражений типа $x < 0$ в данном случае невозможна.

Задание на лабораторную работу

1. Написать программу работы с условным оператором if в соответствии с номером своего варианта.
2. Написать программу с использованием оператора switch в соответствии с номером своего варианта.
3. Сделать выводы о полученных результатах работы программ.

Варианты заданий

Вариант	Оператор if	Оператор switch
1	Написать программу вычисления модуля введенного числа	Написать программу перевода введенного символа от а до f в верхний регистр
2	Написать программу проверки попадания введенного числа в диапазон от -2 до 2	Написать программу перевода введенного символа от А до F в нижний регистр
3	Написать программу проверки не вхождения введенного числа в диапазон от 0 до 5	Написать программу замены введенного символа от 0 до 9 соответствующим числом
4	Написать программу проверки на положительность введенного числа	Написать программу замены введенного числа от 0 до 9 соответствующим символом
5	Написать программу проверки на отрицательность введенного числа	Написать программу замены введенного числа от 0 до 5 соответствующим символом, а все другие значения заменять буквой z.
6	Написать программу определения знака введенного числа	Написать программу замены введенного символа от 0 до 5 соответствующим числом, а все другие символы заменять числом -1
7	Написать программу проверки попадания введенного числа в	Написать программу перевода введенного символа от а до f в

	диапазон от -6 до -2	верхний регистр, а другие символы заменять на Z
8	Написать программу проверки не вхождения введенного числа в диапазон от -5 до -1	Написать программу перевода введенного символа от A до F в нижний регистр, а все другие символы заменять на x
9	Написать программу вычисления суммы модулей двух введенных чисел	Написать программу сравнения введенного числа со значениями 0, 4, 8, 9 и 30
10	Написать программу вычисления $1/a$ с проверкой $a \neq 0$	Написать программу сравнения введенного символа с a, s, d, j и e

Содержание отчета

1. Титульный лист с названием лабораторной работы, номером варианта, фамилией студента и группы.
2. Текст программ.
3. Результаты действия программ.
4. Выводы о полученных результатах работы программ.

Контрольные вопросы

1. Запишите условный оператор if для определения знака переменной var.
2. В каких случаях следует использовать оператор switch?
3. Используя условный оператор, выполните проверку на принадлежность значения переменной диапазону [10; 20).
4. Приведите программу замены малых латинских букв большими с использованием оператора switch.
5. Как записывается логическое равенство в операторе if?
6. Приведите обозначение логического знака «не равно».
7. Какими символами обозначаются логические операции И и ИЛИ в условном операторе if?