Лекция 6. Операции языка Си

Любое выражение языка состоит из операндов (переменных, констант и др.), соединенных знаками операций. Знак операции - это символ или группа символов, которые сообщают компилятору о необходимости выполнения определенных арифметических, логических или других действий.

Операции выполняются в строгой последовательности. Величина, определяющая преимущественное право на выполнение той или иной операции, называется приоритетом. В табл. 6.1 перечислены различные операции языка Си. Их приоритеты для каждой группы одинаковы (группы выделены цветом). Чем большим преимуществом пользуется соответствующая группа операций, тем выше она расположена в таблице. Порядок выполнения операций может регулироваться с помощью круглых скобок.

Таблица 6.1.

Операции языка Си

Знак операции	**Назначение операции**
()	Вызов функции
[ ]	Выделение элемента массива
.	Выделение элемента записи
->	Выделение элемента записи
!	Логическое отрицание
~	Поразрядное отрицание
-	Изменение знака
++	Увеличение на единицу
--	Уменьшение на единицу
&	Взятие адреса
*****	Обращение по адресу
(тип)	Преобразование типа
sizeof( )	Определение размера в байтах
*****	Умножение
/	Деление
%	Определение остатка от деления
+	Сложение
-	Вычитание
<<	Сдвиг влево
>>	Сдвиг вправо
<	Меньше, чем
<=	Меньше или равно
>	Больше, чем
>=	Больше или равно
= =	Равно

!=	Не равно
&	Поразрядное логическое "И"
^	Поразрядное исключающее "ИЛИ"
|	Поразрядное логическое "ИЛИ"
&&	Логическое "И"
||	Логическое "ИЛИ"
?:	Условная (тернарная) операция
=	Присваивание
+=, - =, *=, /=, %=, <<=, >>=, &=, |=, ^=	Составные операции присваивания (например, a*= b, т.е. a = a*b)
,	Операция запятая

Для исключения путаницы в понятиях "операция" и "оператор", отметим, что оператор - это наименьшая исполняемая единица программы. Различают операторы выражения, действие которых состоит в вычислении заданных выражений (например, ***a*=*sin*(*b*)+*c*** или ***j***++), операторы объявления, составные операторы, пустые операторы, операторы метки, цикла и т.д. Для обозначения конца оператора в языке Си используется точка с запятой. Что касается составного оператора (или блока), представляющего собой набор логически связанных операторов, помещенных между открывающей (**{**) и закрывающей (**}**) фигурными скобками ("операторными скобками"), то за ним точка с запятой не ставится. Отметим, что блок отличается от составного оператора наличием определений в теле блока.

Охарактеризуем основные операции языка Си

Сначала рассмотрим одну из них - ***операцию присваивания*** (=). Выражение вида

***x*=*y*;**

присваивает переменной ***x*** значение переменной ***y***. Операцию "=" разрешается использовать многократно в одном выражении, например,

***x*=*y*=*z*=100;**

Пример 6.1. Пример программы ввода числа и присвоения его другим переменным.

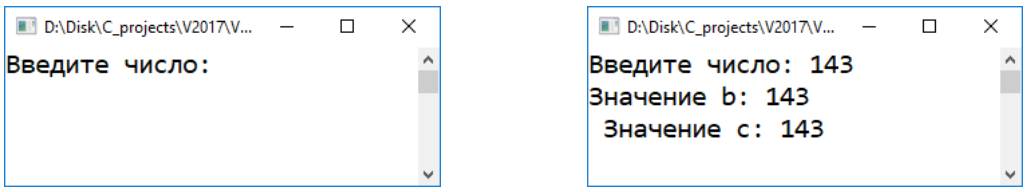
```
#include <stdio.h>                  //подключение заголовочного файла
#include <stdlib.h>                 //для перехода на русский язык

int a,b,c;                          //объявление переменных типа integer

void main()                         //основной цикл
{                                   //начало основного цикла
        system("chcp 1251");        //переходим на русский язык
        system("cls");              //очищаем окно консоли
        printf("Введите число: ");  //выводим в консоль фразу
        scanf_s("%d", &a);          //считывание числа в a
        b=a;                        //операция присвоения
        c=b;
        printf("Значение b: %d\n ", b);   //выводим в консоль
```

```
        printf("Значение c: %d\n ", c);        //выводим в консоль
        getchar(); getchar();
}                                              //конец основного цикла
```

При выполнении кода программы сначала необходимо ввести число и нажать ***Enter*** (рис. 6.1а). После ввода числа оно будет присвоена переменным ***a***, ***b*** и ***c*** (рис. 6.1б).



а *б*

Рис. 6.1. Программа ввода числа и присвоения его другим переменным

Различают ***унарные*** и ***бинарные*** операции. У первых из них один операнд, а у вторых - два. Начнем их рассмотрение с операций, отнесенных к первой из следующих традиционных групп:

- арифметические операции;
- логические операции и операции отношения;
- операции с битами.

Арифметические операции задаются следующими символами: **+, -, *, /, %.** Последнюю из них нельзя применять к переменным вещественного типа. Например,

a=b+c;
x=y-z;
r=t*v;
s=k/l;
p=q%w.

При этом в случае если операция деления присваивается в переменную типа ***integer***, то она будет хранить целую часть от деления.

Пример 6.2. Пример программы ввода двух чисел и вывода их суммы, произведения, разности, результата деления и остатка от деления.

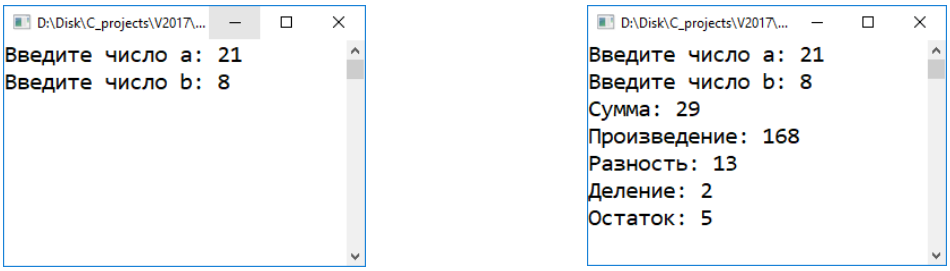
```
#include <stdio.h>                     //подключение заголовочного файла
#include <stdlib.h>                    //для перехода на русский язык

int a,b,c,d,e,f,g;                     //объявление переменных типа integer

void main()                            //основной цикл
{                                      //начало основного цикла
        system("chcp 1251");           //переходим на русский язык
        system("cls");                 //очищаем окно консоли
        printf("Введите число a: ");   //выводим в консоль фразу
        scanf_s("%d", &a);             //считывание числа в a
        printf("Введите число b: ");   //выводим в консоль фразу
        scanf_s("%d", &b);             //считывание числа в b
        c=a+b;                         //расчет суммы
```

```
    d=a*b;                          //расчет произведения
    e=a-b;                          //расчет разности
    f=a/b;                          //деление
    g=a%b;                          //остаток от деления
    printf("Сумма: %d\n", c);       //выводим в консоль
    printf("Произведение: %d\n", d); //выводим в консоль
    printf("Разность: %d\n", e);    //выводим в консоль
    printf("Деление: %d\n", f);     //выводим в консоль
    printf("Остаток: %d\n", g);     //выводим в консоль
    getchar(); getchar(); getchar(); getchar(); getchar();
}                                   //конец основного цикла
```

При выполнении кода программы сначала необходимо ввести два числа и нажать ***Enter*** (рис. 6.2а). После ввода чисел будут выведены результаты арифметических операций с этими числами (рис. 6.2б).



а *б*

Рис. 6.2. Программа ввода двух чисел и вывода их суммы, произведения, разности, результата деления и остатка от деления

Логические операции отношения задаются следующими символами: && ("И"), || ("ИЛИ"), ! ("НЕ"), >, >=, <, <= , = = (равно), != (не равно). Традиционно эти операции должны давать одно из двух значений: истину или ложь. В языке Си принято следующее правило: истина - это любое ненулевое значение; ложь - это нулевое значение. Выражения, использующие логические операции и операции отношения, возвращают 0 для ложного значения и 1 для истинного. Ниже приводится таблица истинности для логических операций.

Таблица 6.2.

Таблица истинности логических операций

x	**y**	**x&&y**	**x||y**	**!x**
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Битовые операции можно применять к переменным, имеющим типы ***int***, ***char***. Их нельзя применять к переменным типов ***float***, ***double***, ***void*** (или более сложных типов). Эти операции задаются следующими символами: ~ (поразрядное отрицание), << (сдвиг влево), >> (сдвиг вправо), & (поразрядное "И"), ^ (поразрядное исключающее "ИЛИ"), | (поразрядное "ИЛИ").

Пример 6.3: Программа вычисления ряда логических операций с числами ***a* = 0000 1111** и ***b* = 1000 1000**:

~*a*=1111 0000 → 240_{10} или -16_{10},
a<<1=0001 1110 → 30_{10},
a>>1=0000 0111 → 7_{10},
a&b=0000 1000 → 8_{10},
a^b=1000 0111 → 135_{10},
a|b=1000 1111 → 143_{10}.

```
#include <stdio.h>                      //подключение заголовочного файла
#include <stdlib.h>                     //для перехода на русский язык

unsigned int a, b, c, d, e, f, g, h;    //объявление переменных типа integer

void main()                             //основной цикл
{                                       //начало основного цикла
        system("chcp 1251");            //переходим на русский язык
        system("cls");                  //очищаем окно консоли
        a = 15;
        b = 136;
        c = ~a;
        d = a<<1;
        e = a>>1;
        f = a&b;
        g = a^b;
        h = a | b;
        printf("Операция ~: %d\n", c);  //выводим в консоль
        printf("Операция <<: %d\n", d); //выводим в консоль
        printf("Операция >>: %d\n", e); //выводим в консоль
        printf("Операция &: %d\n", f);  //выводим в консоль
        printf("Операция ^: %d\n", g);  //выводим в консоль
        printf("Операция |: %d\n", h);  //выводим в консоль
        getchar(); getchar(); getchar();
        getchar(); getchar();
}                                       //конец основного цикла
```

После выполнения данной программы окно вывода будет иметь вид, представленный на рис. 6.3:

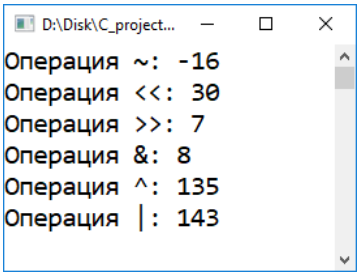


Рис. 6.3. Программа вычисления ряда логических операций

В языке предусмотрены две нетрадиционные операции инкремента (++) и декремента (--). Они предназначены для увеличения и уменьшения на единицу значения операнда. Операции ++ и -- можно записывать как перед операндом, так и после него. В первом случае (**++*n*** или **--*n***) значение операнда

(***n***) изменяется перед его использованием в соответствующем выражении, а во втором (***n***++ или ***n***--) - после его использования.

Пример 6.4: Рассмотрим программу, реализующую операции:

a=b+c++;

a1=b1+ ++c1;

```
#include <stdio.h>                      //подключение заголовочного файла
#include <stdlib.h>                     //для перехода на русский язык

unsigned int a, a1, b, b1, c, c1;       //объявление переменных типа integer

void main()                             //основной цикл
{                                       //начало основного цикла
        system("chcp 1251");            //переходим на русский язык
        system("cls");                  //очищаем окно консоли
        b = 2;
        b1 = b;
        c = 4;
        c1 = c;
        a = b + c++;
        a1 = b1 + ++c1;
        printf("a=%d, b=%d, c=%d\n", a,b,c);          //выводим в консоль
        printf("a1=%d, b1=%d, c1=%d\n", a1, b1, c1);  //выводим в консоль

        getchar(); getchar();
}                                       //конец основного цикла
```

Предположим, что ***b=b1*=2**, ***c=c1*=4**. Тогда после выполнения операций: ***a*=6**, ***b*=2**, ***c*=5**, ***a1*=7**, ***b1*=2**, ***c1*=5**.

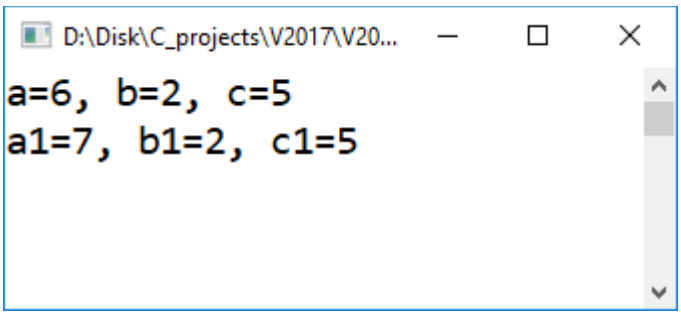


Рис. 6.4. Программа, реализующая операции a=b+c++; a1=b1+ ++c1

Широкое распространение находят также выражения с еще одной нетрадиционной тернарной или условной операцией **?**:. В выражении

y=x?a:b;

y=a, если ***x*** не равен нулю (т.е. истинно), и ***y=b***, если ***x*** равен нулю (ложно). Следующее выражение

y=(a>b)?a:b;

позволяет присвоить переменной ***y*** значение большей переменной (***a*** или ***b***), т.е. ***y=max(a,b)***.

Еще одним отличием языка является то, что выражение вида ***a=a+5***; можно записать в другой форме: ***a+=5***;. Вместо знака + можно использовать и символы других бинарных операций (табл. 6.2).

Пример 6.5: Рассмотрим программу, реализующую операции:

a=a+5;
b+=5;
c=c-10;
d-=10;

```
#include <stdio.h>                          //подключение заголовочного файла
#include <stdlib.h>                         //для перехода на русский язык

int a, b, c, d;                             //объявление переменных типа integer

void main()                                 //основной цикл
{                                           //начало основного цикла
        system("chcp 1251");                //переходим на русский язык
        system("cls");                      //очищаем окно консоли
        a=b=c=d=20;
        a=a+5;
        b+=5;
        c=c-10;
        d-=10;
        printf("a=%d, b=%d, c=%d, d=%d\n", a, b, c,d);     //выводим в консоль
        getchar();
}                                                          //конец основного цикла
```

Предположим, что ***a=b=c=d=20***, после выполнения операций результат будет выведен в консоль (рис. 6.5).

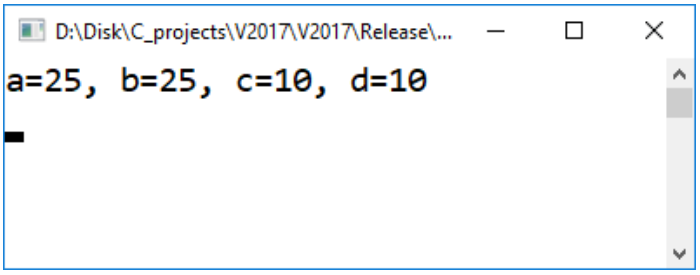


Рис. 6.5. Программа, реализующая операции a=a+5; b+=5; c=c-10; d-=10