Арифметика DevC++

В современной жизни очень сложно обойтись без арифметических операций. Нам постоянно приходится что-то считать: складывать, умножать, вычитать, делить и т.д. Программирование – не исключение. Вам в 99.9% случаев придется ими пользоваться, при написании своих программ. Бояться их не стоит – все арифметические операции – просты, понятны и знакомы нам со школы.

Рассмотрим арифметические операции в следующей таблице

Оператор	Операция, которая проводится с данными
+	сложение данных
-	вычитание данных
*	умножение данных
/	деление данных
%	деление данных по модулю

Тут особое внимание следует уделить делению по модулю (%). Эта операция достаточно часто используется в решении определённых задач. Пример её применения: если нам необходимо поделить по модулю 9 на 4 (9 % 4), результат будет равен 1 (это остаток – то, что на 4 уже не делится на цело). Еще примеры: 20 % 8 = 4 (8 помещается в 20-ти 2 раза: 8 * 2 = 16, 20 – 16 = 4 остаток от деления), 3 % 2 = 1, 99 % 10 = 9, 9 % 10 = 9. Важно:

- деление по модулю применяется только к целочисленным переменным ;
- нельзя делить по модулю на 0;

```
#include <iostream>
using namespace std;

int main()
{
setlocale(0, "");

int number1 = 18;
int number2 = 4;

cout << "number1 = " << number1 << endl;
cout << "number2 = " << number2 << endl;
cout << "number1 + number2 = " << number1 + number2 << endl;
cout << "number1 - number2 = " << number1 - number2 << endl;
cout << "number1 * number2 = " << number1 * number2 << endl;
cout << "number1 / number2 = " << number1 / number2 << endl;
cout << "number1 % number2 = " << number1 % number2 << endl;
cout << endl;

return 0;
}
```

- Тут вы видите, что при делении **num1** на **num2**, на экране появилась только целая часть – 4 (хотя точное значение 4.5). Дробная часть отсекается, так как переменные определены, как целочисленные – **int**. А в результате деления по модулю мы видим 2 – то что осталось в остатке от деления 18 на 4.
- Еще что хотелось бы рассмотреть это так называемые **комбинированные (или составные) операторы**. Помимо выполнения своей арифметической роли, они одновременно выполняют роль присваивания значения переменным. Вот список таких составных операторов:

Комбинированный оператор	Операция (что происходит при использовании в коде)
+=	сложение данных с присваиванием
-=	вычитание данных с присваиванием
***=**	умножение данных с присваиванием
/=	деление данных с присваиванием
%=	деление данных по модулю с присваиванием

```
#include <iostream>
using namespace std;

int main()
{
setlocale(0, "");

int number1 = 10;
int number2 = 4;

cout << "number1 = " << number1 << endl;
cout << "number2 = " << number2 << endl;

number1 += number2;  // эквивалентно записи number1 = number1 + number2
cout << "Результат от += : " << "number1 = " << number1 << endl;
number1 -= number2;  // number1 = number1 - number2 и т.д.
cout << "Результат от -= : " << "number1 = " << number1 << endl;
number1 *= number2;
cout << "Результат от *= : " << "number1 = " << number1 << endl;
number1 /= number2;
cout << "Результат от /= : " << "number1 = " << number1 << endl;
number1 %= number2;
cout << "Результат от %= : " << "number1 = " << number1 << endl;

cout << endl;

return 0;
}
```