Лекция 2

Операции и выражения языка программирования С++. Структура программ в среде Visual C++ 2010. Операторы ввода и вывода.

Цель лекции. Изучить операции и выражения языка С++, порядок и особенности их применения. Изучить операторы ввода-вывода, структуру программ в среде Visual C++ 2010.

Основные вопросы лекции:

1. Операции в С++.
2. Арифметические операции в С++.
3. Операция присваивания в С++.
4. Арифметические операции с присваиванием в С++.
5. Выражения в С++.
6. Стандартные математические функции в среде Visual C++ 2010.
7. Операция приведения типа.
8. Операции инкремент и декремент.
9. Структура программ на языке С++ в среде Visual C++ 2010.
10. Операторы ввода и вывода в среде Visual C++ 2010.

1. Операции в С++

Операция в языке С++ - это проведение строго определенных знаком операции действий над субъектами операции - **операндами**. Каждая операция в языке С++ характеризуется своим приоритетом. **Приоритет операции** – это порядок, в котором проводится вычисление значения выражения (таб. 1).

Каждая операция имеет свой собственный порядок выполнения - слева направо или справа налево. Знак операции – это символ или сочетание символов, сообщающих компилятору о необходимости проведения определенных арифметических, логических или других действий.

Для каждой операции, определенной в С++, определено количество операндов:

1) один операнд – **унарная** операция; например, **x^2;**
2) два операнда – **бинарная** операция; например, операция сложения **y+z**;
3) три операнда – операция **условие**. Такая операция единственная.

Полный список операций языка С++ приведен в таблице 1.

Таблица 1. Операции языка программирования С++

Операция	Название операции	Приоритет	Порядок выполнения
Первичные и постфиксные операции			
[ ]	Індексация массива	16	Слева направо
( )	Вызов функции	16	Слева направо
.	Элемент структуры	16	Слева направо
->	Элемент указателя	16	Слева направо
+ +	Постфиксный инкремент	15	Слева направо
- -	Постфиксный декремент	15	Слева направо
Одноместные (унарные) операции			
+ +	Префиксный инкремент	14	Справа налево
- -	Префиксный декремент	14	Справа налево
sizeof	Размер в байтах	14	Справа налево
(тип)	Приведение типа	14	Справа налево
~	Поразрядное **NOT**		
!	Логическое **Не**	14	Справа налево
-	Унарный минус	14	Справа налево
&	Взятие адреса	14	Справа налево
*****	Разыменование указателя	14	Справа налево
Мультипликативные операции			
*****	Умножение	13	Слева направо
/	Деление	13	Слева направо
%	Взятие по модулю	13	Слева направо
Аддитивные операции			

+	Сложение	12	Слева направо
-	Вычитание	12	Слева направо
Операции поразрядного сдвига			
<<	Сдвиг влево	11	Слева направо
>>	Сдвиг вправо	11	Слева направо
Операции отношения (сравнения)			
<	Меньше	10	Слева направо
<=	Меньше или равно	10	Слева направо
>	Больше	10	Слева направо
>=	Больше или равно	10	Слева направо
=	Равно	9	Слева направо
!=	Не равно	9	Слева направо
Поразрядные операции			
&	Поразрядное **AND**	8	Слева направо
^	Поразрядное **XOR**	7	Слева направо
!	Поразрядное **OR**	6	Слева направо
Логические операции			
&&	Логическое **AND**	5	Слева направо
!!	Логическое **OR**	4	Слева направо
Условная операция			
?:	Условная операция	3	Справа налево
Операции присваивания			
=	Присваивание	2	Справа налево
*=	Присваивание умножения	2	Справа налево
/=	Присваивание деления	2	Справа налево
%=	Присваивание модуля	2	Справа налево
+=	Присваивание суммы	2	Справа налево
-=	Присваивание разности	2	Справа налево
<<=	Присваивание левого сдвига	2	Справа налево
>>=	Присваивание правого сдвига	2	Справа налево
&=	Присваивание **AND**	2	Справа налево
^=	Присваивание **XOR**	2	Справа налево
!=	Присваивание **OR**	2	Справа налево
,	Запятая	1	Справа налево

2. Арифметические операции

В языке С++ используется пять основных математических операций: сложение, вычитание, умножение, деление и деление по модулю (см. таб. 2). Эти операции применяются к данным целого и действительного типов. Операции сложения, вычитания, умножения и деления выполняются слева направо. Если операнды имеют один тип, результат арифметической операции будет иметь тот же тип. Когда эти операции применяются для действительных чисел, это обычные арифметические операции. Но если операция деления применяется к целым операндам, то результат операции будет целым, а остаток от деления отбрасывается. Например, 11/3 будет равно 3, а выражение 1/2 будет равно нулю.

Таблица 2. Арифметические операции в языке С++

Операция	Знак в языке С++	Порядок выполнения
Сложение	+	Слева направо
Вычитание	-	Слева направо
Умножение	*	Слева направо
Деление	/	Слева направо
Деление по модулю	%	Слева направо

Операция **деление по модулю** используется только с целыми операндами и дает остаток от деления первого операнда на второй. Так, результат выполнения операции 11%3 будет равен 3, а результатом операции 5%5 будет ноль.

3. Операция присваивания

Присваивание – единственная операция, изменяющая значение одного из своих операндов (не считая операций инкремента и декремента, которые будут рассмотрены ниже).

Операция присваивания обозначается знаком "=" и записывается следующим образом:

A = V;

где **A** –переменная любого базового типа данных языка C++; V –выражение.

При этом вычисляется значения выражения **V**, стоящего в правой части, а затем вычисленное значение **V** присваивается переменной **A**.

Особенностью операции присваивания в языке C++ есть возможность записать следующий оператор:

a = b = c = x*y;

В этом случае присваивание выполняется справа налево: вначале вычисляется значение **x*y**, а потом это значение присваивается **c,** потом **b,** затем **a**.

4. Арифметические операции с присваиванием

Одним из способов сократить объем программного кода являются арифметические операции с присваиванием. Например, операцию

total = total + item;

можно записать как

total += item;

С присваиванием комбинируется не только операция сложения, но и другие арифметические операции: **-=, *=, /=, %=** и т.д.

5. Выражения в C++

В языке C++ следующим уровнем представления данных после переменных и констант являются выражения. **Выражение** – это некоторая допустимая комбинация переменных, констант, функций и знаков операций для вычислений в программах. Выражения в языке C++ записываются в строчку. Например, формула

$$d = \frac{a + b(c + e)}{c(a + b) + t}$$

в языке C++ запишется следующим образом: $d = (a + b*(c + e))/(c*(a + b) + t))$.

В приведенном выше выражении знак "=" обозначает операцию **присваивания**, которая выполняется следующим образом. Вычисляется значение выражения в правой части и присваивается переменной **d.**

Математические действия с переменными, константами и функциями в языке C++ записываются только в строчку, при этом для соблюдения необходимой по условию задачи очередности операций используются круглые скобки.

6. Стандартные математические функции в среде Visual C++ 2010

В языке C++ в выражения можно вставлять стандартные математические функции, которые вызываются из библиотеки **<math.h>**. Перечень математических функций, которые чаще всего встречаются в вычислениях, приведены в таблице 3.

Таблица 3. Основные стандартные математические функции (библиотека math.h)

Название функции	**Что вычисляет**	**Тип данных функции и аргумента**
abs(x)	Абсолютное значение (модуль) аргумента |x|	**int abs(int x)**
exp(x)	Экспонента e^x	**double exp(double x)**
log(x)	Натуральный логарифм ln x	**double log(double x)**
log10(x)	Десятичный логарифм lg x	**double log10(double x)**
pow(x,y)	Возведение в степень x^y	**double pow(double x, double y)**

Название функции	Что вычисляет	Тип данных функции и аргумента
sqrt(x)	Квадратный корень $\sqrt{x}$.	**double sqrt(double x)**
fmod(x,y)	Остаток от деления x/y	**double fmod(double x, double y)**
sin(x)	Синус (угол задается в радианах)	**double sin(double x)**
asin(x)	Арксинус (угол задается в радианах от –1 до +1)	**double asin(double x)**
cos(x)	Косинус (угол задается в радианах)	**double cos(double x)**
acos(x)	Арккосинус (угол задается в радианах от –1 до +1)	**double acos(double x)**
tan(x)	Тангенс (угол задается в радианах)	**double tan(double x)**
atan(x)	Арктангенс (угол задается в радианах)	**double atan(double x)**

При обращении к этим функциям необходимо придерживаться следующих правил:

1) **x** и **y** должны быть типа **double**;

2) углы (аргументы) в тригонометрических функциях задаются в радианах.

3) вычисляемые функциями данные имеют тип **double.**

Например, выражение

$$y = \sin(x)\frac{e^x + z^5 - 4.5 \cdot 10^2 \sqrt{x}}{tg(a)(z^x + b)}$$

на языке C++ будет иметь вид:

y=sin(x)(exp(x)+pow(z,* 5*)* - 4.5*10*10**sqrt(x))/(tan(x)*(pow(z,x)+b))* .

7. Операция приведения типа.

В арифметическом выражении могут присутствовать операнды разных типов - как целые, так и действительные. И те, и другие могут иметь разную длину (**chort**, **long**). В то же время оба операнда любой арифметической операции должны иметь один и тот же тип. В языке C++ при вычислении значения такого выражения происходит автоматическое приведение типов.

Операция приведения типов состоит в следующем. На каждом шаге вычисления значения выражения выполняется одна операция над одной парой операндов. Если типы операндов не совпадают, то операнд меньшего ранга приводится к типу более высокого ранга. Обычно короткие типы приводятся к более длинным, что обеспечивает сохранение значимости и точности:

(char, short) -> int -> unsigned int -> long -> unsigned long -> float -> double -> long double

Правила, используемые для автоматического приведения типов при вычислении значения арифметического выражения, следующие:

1. Все переменные типа **char**, **short int** преобразуются в **int.**

2. В любой паре операнды приводятся к одному типу. Например, если один из операндов **double**, то и другой преобразуется в **double**.

3. В операторе присваивания конечный результат приводится к типу переменной в левой части оператора. При этом ранг типа может как повышаться, так и понижаться.

Возможно принудительное приведение типа, выполняемое с помощью операции приведения типа, имеющей следующую структуру:

Тип Выражение;

где **тип** – один из стандартных (встроенных) типов данных в C++.

Операция может применяться к любому операнду в выражении. Например, чтобы результат деления на 2 переменной **x** типа **int** имел тип **float**, надо записать

float x / 2; .

8. Операции инкремент и декремент

Унарная операция **инкремента** увеличивает свой операнд (**обязательно переменную**) на единицу и записывается **i ++** ,
что эквивалентно записи **i = i + 1** .

Унарная операция **декремента** уменьшает свой операнд на единицу и записывается **i –** , что эквивалентно записи **i = i – 1;**

Эти операции могут использоваться и в выражениях:

sum = sum + x *i++; .

Инкремент и декремент реализуются в двух формах: префиксной **++i** и постфиксной **i++.**

9. Структура программ в среде Visual C++ 2010

Программа на языке C++ состоит из следующих элементов (см. рис. 1):

1.Директивы препроцессора.
2. Указания компилятору.
3. Объявления и определения.
4. Одна или несколько функций.

директива препроцессора 1 # директива прероцессору 2 ..... # директива прероцессору N
указание компилятору 1 # указание компилятору 2 ..... # указание компилятору M
Объявление и определение (инициализация) переменных, констант, функций
Функция 1 Функция 2 ..... Функция N

Рис. 1. Структура программы на языке C++

Рассмотрим структуру программ на языке C++ для консольного приложения Win 32 среды Visual C++ 2010 на примере одной простой программы. Это программа для вычисления переменной **a** по формуле $a = b\dfrac{c + 2d - cd}{d(5c + 4b)}$ с использованием операторов консольных ввода и вывода данных.

Блок-схема алгоритма решения данной задачи имеет линейный вид и приведена на рис. 2.

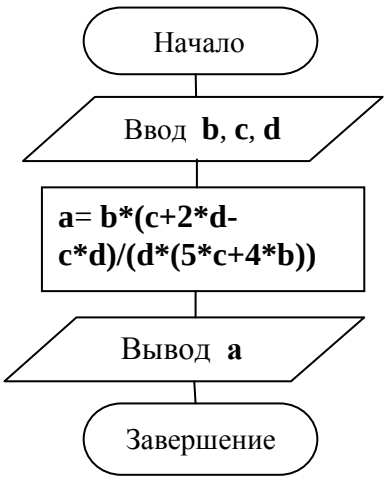

Рис. 2. Блок-схема алгоритма вычисления переменной **a**

Код (текст) программы с комментариями приведен ниже.

```
#include "stdafx.h"
#include <conio.h>                    //Файл conio.h обеспечивает задержку окна DOS на
                                      //экране дисплея
#include <iostream>                   //Директива include подключает файл заголовка
                                      //iostream
using namespace std;                  //Подключает все имена из пространства имен  std
int _tmain(int argc, _TCHAR* argv[ ]) //Объявление главной функции
{                                     //Начало главной функции
      double a;                       //Объявление переменной   a
      double b, c=2, d=3;             //Объявление переменных   b, c, d и их
                                      //инициализация
      cout<<"vvedite b"<<endl;        //Вывод на экран надписи-приглашения  vvedite b .
                                      //Здесь endl – признак перемещения курсора на
                                      //новую  строку
      cin>>b;                         //Ввод значения переменной  b
      a=b*(c+2*d-c*d)/(d*(5*c+4*b));  //Вычисление переменной  a по заданной формуле
      cout<<"a = "<<a;                //Вывод на экран значения a
getch();                              //Функция  задержки окна DOS на экране
return 0;
}
```

Директивы препроцессора предназначены для обработки исходного текста программы перед компиляцией. Любая директива должна начинаться с символа #. На каждой строке может располагаться только одна директива. Например, по директиве

#include <conio.h>

в текст исследуемой нами программы будет вставлено содержимое так называемого заголовочного файла – в данном примере с именем **conio.h** (для задержки на экране дисплея окна DOS). Заголовочные файлы содержат различную информацию, необходимую для успешной компиляции и работы программы. В данной программе это строки 1, 2 и 3.

Указания компилятору осуществляются с помощью директивы препроцессора **pragma**, служащей для установки различных параметров компилятора. Директива имеет вид:

#pragma директива .

Например, директива **#pragma argsused** означает, что компилятору не нужно выдавать предупреждающие сообщения о том, что параметры функции не используются внутри нее.

В анализируемой программе такие указания компилятору отсутствуют.

Функции описывают совокупность тех действий, которые должна выполнить программа.

Если функций несколько, то среди них выделятся одна - **главная**, которая имеет имя **tmain(…)**. С нее начинается выполнение программы. Если программа содержит единственную функцию, то она обязательно должна иметь имя **tmain (…).**

В языке С++ функции являются строительными блоками программы, спроектированными для решения конкретных задач. Тело функции (программа, описывающая ее работу) всегда заключается в фигурные скобки. В нашей программе это строки 6 и 15.

Программа может содержать произвольное число директив, указаний, объявлений и определений. Они могут располагаться в любом месте программы, но действовать в программе они начинают **только с того места**, где расположены.

В исходный текст программы можно вставлять текст из другого файла с помощью директив препроцессора **#include** (включать):

 # include < имя файла > .

Если имя файла размещено в **угловых скобках**, то поиск нужного файла производится только в стандартных каталогах. Если имя файла размещено в **кавычках** - поиск нужного файла производится в текущем каталоге. Например, директива

#include <iostream>

сообщает препроцессору о необходимости включить в программу файл **iostream**, содержащий функции ввода и вывода данных в консольном приложении Win 32 среды Visual C++ 2010.

После директив и указаний препроцессору в исследуемой программе располагается определение функции **tmain(…)**. Любая программа на C++ содержит такую функцию, которая является главной (см. выше). Внутри функции, в свою очередь, должны быть объявлены используемые в ней переменные, константы и функции. Стандартные математические функции (sin, cos и др.) объявлять не надо.

Подробнее о правилах и особенностях создания функций в программе мы остановимся в последующих лекциях. Здесь же укажем, что в анализируемой программе в строках 7 и 8 объявлены используемые переменные, в строках 9, 10 и 12 организован ввод и вывод переменных, в строке 11 записан оператор для вычисления переменной **a**. В строку 13 помещена функция **getch()**, вызываемая из заголовочного файла **conio.h** для задержки на экране дисплея окна DOS, необходимого для ввода и вывода информации при работе программы.

Оператор **return** всегда должен присутствовать в конце программы, описывающей работу функции **tmain(…)**. Подробно работа этого оператора будет рассмотрена при изучении функций в языке C++ в следующих лекциях.

2. Операторы ввода и вывода в консольном приложении Win 32 среды Visual C++ 2010

В Консольном приложении Win32 среды программирования Visual C++ 2010, в котором консольный (с использованием клавиатуры) ввод данных производится при помощи оператора **cin**. В C++ этот оператор называется также потоком ввода. Например, для ввода значений трех переменных надо записать:

cin>>a >>b>>c;

где **>> -** символ операции извлечения данных из потока; **a**, **b** и **c –** переменные, значения которых будут вводится. Вводимые значения должны разделяться пробелами, а ввод завершается нажатием клавиши **<Enter>**. Потоковый ввод и его операции автоматически распознают переменные и данные любого типа.

Консольный (на экран дисплея) вывод данных производится при помощи оператора **cout**. В C++ этот оператор называется также потоком вывода. Например, для вывода значений трех переменных надо записать:

cout<<a<<b<<c;

где **<< -** символ операции вставки данных в поток; **a**, **b** и **c –** переменные, значения которых будут выводиться на экран. Потоковый вывод и его операции автоматически распознают переменные и данные любого типа. Вывод в Win32 производится в командную строку окна DOS. Помимо данных можно выводить и текстовую строку, заключив ее в кавычки:

cout<<"Summa a+b+c = "<<d;

Стандартные функции ввода и вывода находятся в библиотечном файле **iostream**. Чтобы связать разрабатываемую программу со стандартной библиотекой ввода-вывода, необходимо в начале программы указать оператор подключения этой библиотеки:

 # include <iostream> .

Точка с запятой после операторов **include** не ставится.

Оператор **cout** часто используется с различными опциями (управляющими последовательностями) для расширения его функций (см. раздел 1 лекции 2).

В операторе **cout** допускается производить арифметические действия. Например, при выполнении оператора

cout <<2 + 3 + 4

на экран дисплея будет выведена результат – цифра 9.

Вопросы для самоконтроля

1. Дайте определение операции в языке *C++*.
2. Что такое операнд?
3. Дайте определение приоритета операции и приведите примеры.
4. В чем уникальность операции присваивания?
5. Как записываются математические действия с переменными, константами и функциями в языке *C++*?
6. Каковы правила обращений к математическим функциям в языке *C++*?
7. Что такое выражение?
8. Охарактеризуйте операции инкремента и декремента.
9. Чем отличаются программы с линейной структурой?
10. Как вывести на экран значение переменной?
11. Как в одном операторе объявить тип переменной и задать ее значение?
12. Перечислите опции оператора *cout* и раскройте их действие.
13. Какие функции выполняет оператор **include**?
14. Какие функции выполняет файл заголовка **iostream**?
15. Для чего служат комментарии в программном коде?
16. Что такое проект в **Visual C++** 2010?
17. Какие основные части программы в **Visual C++** 2010?