Урок 11. Форматированный ввод-вывод в C++

Для организации ввода-вывода в C++ можно использовать средства языка C (conio.h). Однако в C++ существует стандартная библиотека классов, ориентированная на организацию потокового ввода-вывода. Классы ввода-вывода образуют иерархию по принципу наследования. Базовым в этой иерархии является класс ios (исключение составляют лишь классы буферизированных потоков). В классе ios объединены базовые данные и методы для ввода-вывода. Прямыми потомками класса ios являются классы istream и ostream. Класс istream — это класс входных потоков; ostream — класс выходных потоков. Потомком этих двух классов является iostream — класс двунаправленных потоков ввода-вывода. С этим классом мы уже много раз имели дело, подключая его к программам с помощью головного файла iostream.h.

Объект cout принадлежит к классу ostream и представляет собой поток вывода, связанный с дисплеем. Объект cin принадлежит классу istream и является потоком ввода, связанным с клавиатурой. Оба эти объекта наследуются классом iostream.

Знак << обозначает перегруженную операцию вставки символов в поток вывода cout, а >> — знак операции извлечения из потока ввода cin.

Для организации форматированного потокового ввода-вывода в C++ существуют два средства:

- применение функций-членов класса ios для управления флагами форматирования;
- применение функций-манипуляторов.

Управление флагами форматирования. Флаги форматирования — двоичные коды, управляющие форматом выводимых значений. В заголовочном файле iostream. h определено следующее перечисление, задающее флаги форматирования:

```
enum{
skipws          = 0x0001 отбрасывание пробелов
left            = 0x0002 выравнивание по левому краю поля
right           = 0x0004 выравнивание по правому краю поля
internal        = 0x0008 заполнение пустых позиций
dec             = 0x0010 выдача в десятичном виде
oct             = 0x0020 выдача в восьмеричном виде
hex             = 0x0040 выдача в шестнадцатеричном виде
showbase        = 0x0080 выдача основания сист. счисления
showpoint       = 0x0100 выдача позиции точки
uppercase       = 0x0200 выдача в формате xx.xxxx Exx
showpos         = 0x0400 выдача знака у положит. числа
scientific      = 0x0800 выдача в форме с плавающ. точкой
fixed           = 0x1000 выдача в форме с фиксир. точкой
unibuf          = 0x2000 улучшенная выдача
stdio           = 0x4000 освобождение потока
```

Фактически в этом списке содержатся имена констант, определяющие флаги соответствующих назначений. Коду формата соответствует целый тип long.

Изменить состояние флагов формата можно с помощью функции-члена класса ios, имеющей прототип

```
long  setf (long  flags)
```

Например, чтобы установить флаг showbase в активный режим (включить) применительно к стандартному потоку вывода cout, используется оператор

```
cout.setf(ios::showbase);
```

Для установки флагов можно использовать побитовые операции. Например:

```
cout.setf(ios::left|ios::hex);
```

В результате включатся одновременно флаги, управляющие выравниванием по левому краю и выводом целых значений в шестнадцатеричной системе.

Для выключения используется функция

```
unsetf(long  flags ) ;
```

Например, для отмены вывода основания системы счисления используется оператор:

```
cout.unsetf(ios::showbase);
```

Вот еще некоторые функции-члены класса ios:

`long flags(void)` - возвращает текущее состояние флагов;

`int width(int len)` — возвращает текущую ширину поля вывода и устанавливает значение ширины, равное len;

`char fill(char ch)` — возвращает текущий символ заполнения и устанавливает новый символ заполнения ch;

`int precision(int num)` — возвращает текущее число десятичных знаков после точки и устанавливает значение этого параметра равным num.

Пример 1. Следующая программа иллюстрирует применение рассмотренного способа управления форматным выводом.

```
#include  <iostream>
#include  <clocale>

using namespace std;

int  main(void)
{
setlocale(LC_ALL,"RUS");
long fl;
fl = cout.flags( );
cout << "Исходное  состояние флагов: " << fl << "\n";
// Выведется целое число – код флагов,
// установленных по умолчанию
cout.set(ios::shoupos);
cout.set(ios::scientific);
cout << 123 << "  " << 1.2345678 << "\n";
//Выведется:
//+123 +1.23456e+00
cout.set(ios::hex|ios::showbase);
cout.unsetf(ios::showpos);
cout.width(15);
cout.precision(10);
cout << 123 << "  " << 123.456 << "  "1.2345678 << "\n";
//Выведется:
//0x7B 1.23456e+02 1.2345678e+00
cout << "Новое  состояние флагов: " << cout.flags( ) << "\n";
//Выведется:
//Новое состояние  флагов:0x28C1
cout.flags(fl);//Возврат  к исходному  состоянию
```

```
cout << "После  восстановления исходных флагов:\n";
cout << 123 << "  "<<123.456 << 1.2345678 << "\n";
//Выведется:
//После восстановления исходных флагов:
//123 123.456 1.234567
}
```

Использование манипуляторов. Для управления форматами потокового вывода можно использовать специальные функции, называемые манипуляторами. Доступ в программе к стандартным манипуляторам можно получить, подключив файл iomanip.h.

Список стандартных манипуляторов:

```
dec                          Десятичный формат
endl                         Вывод  "\n"  и освобождение буфера
ends                         Вывод NULL
flush                        Освободить поток
hex                          Шестнадцатеричный формат числа
resetiosflags  (long  f)     Отключить флаги, определенные  f
setbase(int  base)           Установить основание системы счисления
setfill(char ch)             Установить символ заполнения
setiosflags(long f)          Включить флаги, указанные  f
setprecision(int p)          Установить  p цифр в дробной части
setw(int w)                  Установить ширину поля выдачи ws
ws                           Режим пропуска символов пробела
```

Пример 2. В следующей программе вычисляется и выводится на экран таблица значений функций sinx и cosx на n шагах в интервале от 0 до p. Для форматирования таблицы результатов используются манипуляторы.

```
#include  <iostream>
#include  <cmath>
#include  <iomanip.h>

using namespace std;

int main()
{
double  a, b, x;
int n = 20;
a = 0;  b = 4*atan (1.);
cout << "  x            sin( x )          cos( x ) " << endl;
cout << "-------------------------------------------------- " << endl;
for(x = a; x <= b; x = x + (b - a)/n)
cout << setprecision (4) << setw(10) << x << "  "
<< setprecision (4) << setw(10) <<sin (x) << "  "
<<setprecision (4) <<setw (10) << cos (x) <<endl;
}
```

Начальная часть таблицы, выводимой по этой программе, имеет вид:

```
x           sin (x)           cos(x)
-------------------------------------------------
0           0                 1
0.1571      0.1564            0.9877
0.3142      0.309             0.9511
0.4712      0.454             0.891
0.6283      0.5878            0.809
0.7854      0.7071            0.7071
……
```

Под каждое число выделяется по 10 позиций на экране. По умолчанию число занимает крайнюю правую позицию в отведенном под него поле. Оставшиеся слева позиции занимает символ-заполнитель. По умолчанию символом-заполнителем является пробел. Однако с помощью манипулятора setfill( ) его можно заменить. Если в крайних правых позициях оказываются нули, то они не выводятся. Действие манипулятора распространяется только на значение, непосредственно следующее за ним в потоке вывода.

Информационные источники:

1. Семакин И.Г., Шестаков А.П. Основы программирования: Учебник.- М.: Мастерство, 2002.- 432 с.