Возможность управлять вводом-выводом в С++, обеспечивают форматирующие функции-члены, флаги и манипуляторы. Флаги, функции и манипуляторы выполняют одну и туже задачу — задают определённый формат ввода/вывода информации в потоках. Ввод/вывод на экран/с экрана в С++ осуществляется с помощью операторов cin и cout соответственно, а значит  манипуляторы форматирования используются совместно с данными операторами ввода/вывода. Различие между функциями флагами и манипуляторами форматирования состоит в способе их применения. Теперь рассмотрим способы применения объектов форматирования.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | //Основные форматирующие функции-члены:  cout.fill('/\*symbol\*/');     // устанавливает символ заполнитель  // где symbol - символ заполнитель, символ передаётся в одинарных кавычках    cout.width(/\*width\_field\*/); // задает ширину поля  // где width\_field - количество позиций(одна позиция вмещает один символ)    cout.precision(/\*number\*/);  // задает количество знаков после десятичной точки  // где number - количество знаков после десятичной точки |

Доступ к функциям осуществляется через операцию **точка**, а в круглых скобочках передаётся аргумент. Аргумент функции fill()может передаваться в виде символа, обрамленного одинарными кавычками, или в виде числа(код символа). Одних функций не достаточно для форматирования потоков ввода/вывода, поэтому в С++ предусмотрен ещё один способ форматирования — флаги.

Флаги форматирования позволяют включить или выключить один из параметров ввода/вывода.  Чтобы установить флаг ввода/вывода, необходимо вызвать функцию setf(), если необходимо отключить флаг вывода, то используется функция unsetf(). Далее показаны конструкции установки и снятия флагов вывода.

|  |  |
| --- | --- |
| 1  2  3 | // установка флага вывода  cout.setf( ios::/\*name\_flag\*/ );  // где name\_flag - это имя флага |

Доступ к функциям оператора вывода выполняется через операцию точка. Метод setf()принимает один аргумент — имя флага. Флаги вывода объявлены в классе ios, поэтому, перед тем, как обратиться к флагу, необходимо написать имя класса — ios, после которого, с помощью операции разрешения области действия, вызвать нужный флаг.

|  |  |
| --- | --- |
| 1  2  3 | // снятие флага вывода  cout.unsetf( ios::/\*name\_flag\*/ );  // где name\_flag - это имя флага |

Если   при   вводе/выводе   необходимо   установить(снять) несколько   флагов,   то   можно воспользоваться  [поразрядной логической операцией](http://cppstudio.com/obuchenie_cpp/porazradnye-logicheskie-operatsii)  **ИЛИ**  |.   В  этом   случае   конструкция языка C++ будет такой:

|  |  |
| --- | --- |
| 1  2  3  4  5 | // установка нескольких флагов  cout.setf( ios::/\*name\_flag1\*/ | ios::/\*name\_flag2\*/ | ios::/\*name\_flag\_n\*/ );    // снятие нескольких флагов  cout.unsetf( ios::/\*name\_flag1\*/ | ios::/\*name\_flag2\*/ | ios::/\*name\_flag\_n\*/ ); |

В таблице 1 подробно описаны основные флаги форматирования, а также показаны примеры использования флагов.

| таблица 1 — Флаги форматирования в С++ | | | |
| --- | --- | --- | --- |
| **Флаг** | **Назначение** | **Пример** | **Результат** |
| **boolalpha** | Вывод   логических величин в текстовом виде (true, false) | **cout.setf(ios::boolalpha);**  **bool log\_false = 0,**  **log\_true = 1;**  **cout << log\_false << endl**  **<< log\_true << endl;** | **false true** |
| **oct** | Ввод/вывод   величин   в  восьмеричной системе   счисления (сначала снимаем флаг dec, затем устанавливаем флаг oct) | **cout.unsetf(ios::dec);**  **cout.setf(ios::oct);**  **int value;**  **cin >> value;**  **cout << value << endl;** | ввод:**9910** вывод:**1438** |
| **dec** | Ввод/вывод   величин   в десятичной   системе счисления   (флаг установлен по умолчанию) | **cout.setf(ios::dec);**  **int value = 148;**  **cout << value << endl;** | **148** |
| **hex** | Ввод/вывод   величин   в шестнадцатеричной системе   счисления (сначала снимаем флаг dec, затем устанавливаем флаг hex) | **cout.unsetf(ios::dec);**  **cout.setf(ios::hex);**  **int value;**  **cin >> value;**  **cout << value << endl;** | ввод:**9910** вывод:**6316** |
| **showbase** | Выводить индикатор основания   системы счисления | **cout.unsetf(ios::dec);**  **cout.setf(ios::oct | ios::showbase);**  **int value;**  **cin >> value;**  **cout << value << endl;** | ввод:**9910** вывод:**01438** |
| **uppercase** | В шестнадцатеричной системе счисления использовать буквы верхнего регистра(по умолчанию установлены буквы нижнего регистра) | **cout.unsetf(ios::dec);**  **cout.setf(ios::hex | ios::uppercase);**  **int value;**  **cin >> value;**  **cout << value << endl;** | ввод:**25510** вывод:**FF16** |
| **showpos** | Вывод знака плюс + для положительных чисел | **cout.setf(ios::showpos);**  **int value = 15;**  **cout << value << endl;** | **+15** |
| **scientific** | Вывод чисел с плавающей точкой в экспоненциальной форме | **cout.setf(ios::scientific);**  **double value = 1024.165;**  **cout << value << endl;** | **1.024165e+003** |
| **fixed** | Вывод чисел с плавающей точкой в фиксированной форме(по умолчанию) | **double value = 1024.165;**  **cout << value << endl;** | **1024.165** |
| **right** | Выравнивание по правой границе(по умолчанию). Сначала необходимо установить ширину поля(ширина поля должна быть заведомо большей чем, длина выводимой строки). | **cout.width(40);**  **cout << «cppstudio.com» << endl;** | **\_\_cppstudio.com** |
| **left** | Выравнивание по левой границе. Сначала необходимо установить ширину поля(ширина поля должна быть заведомо большей чем, длина выводимой строки). | **cout.setf(ios::left);**  **cout.width(40);**  **cout << «cppstudio.com» << endl;** | **cppstudio.com\_\_** |

Ещё один способ форматирования — форматирование с помощью манипуляторов. Манипулятор — объект особого типа, который управляет потоками ввода/вывода, для форматирования передаваемой в потоки информации. Отчасти манипуляторы дополняют функционал, для форматирования ввода/вывода. Но большинство манипуляторов выполняют точно, то же самое, что и функции с флагами форматирования. Есть случаи, когда проще пользоваться флагами или функциями форматирования, а иногда удобнее использовать манипуляторы форматирования. Именно по этому в С++ предусмотрено несколько средств форматирования ввода/вывода. В таблице 2 показаны основные манипуляторы форматирования С++.

| **Манипулятор** | **Назначение** | **Пример** | **Результат** |
| --- | --- | --- | --- |
| **endl** | Переход на новую строку при выводе | **cout << «website:» << endl << «cppstudio.com»;** | **website:** **cppstudio.com** |
| **boolalpha** | Вывод   логических  величин в текстовом  виде (true, false) | **bool log\_true = 1;**  **cout << boolalpha << log\_true << endl;** | **true** |
| **noboolalpha** | Вывод   логических  величин в числовом  виде (true, false) | **bool log\_true = true;**  **cout << noboolalpha << log\_true << endl;** | **1** |
| **oct** | Вывод   величин   в  восьмеричной  системе   счисления | **int value = 64;**  **cout << oct << value << endl;** | **1008** |
| **dec** | Вывод   величин   в  десятичной   системе  счисления   (по умолчанию) | **int value = 64;**  **cout << dec << value << endl;** | **6410** |
| **hex** | Вывод   величин   в  шестнадцатеричной  системе   счисления | **int value = 64;**  **cout << hex << value << endl;** | **408** |
| **showbase** | Выводить индикатор  основания   системы  счисления | **int value = 64;**  **cout << showbase << hex << value << endl;** | **0x40** |
| **noshowbase** | Не выводить индикатор  основания   системы  счисления (по умолчанию). | **int value = 64;**  **cout << noshowbase << hex << value << endl;** | **40** |
| **uppercase** | В шестнадцатеричной системе счисления использовать буквы верхнего регистра (по умолчанию установлены буквы нижнего регистра). | **int value = 255;**  **cout << uppercase << hex << value << endl;** | **FF16** |
| **nouppercase** | В шестнадцатеричной системе счисления использовать буквы нижнего регистра (по умолчанию). | **int value = 255;**  **cout << nouppercase << hex << value << endl;** | **ff16** |
| **showpos** | Вывод знака плюс + для положительных чисел | **int value = 255;**  **cout << showpos<< value << endl;** | **+255** |
| **noshowpos** | Не выводить знак плюс **+** для положительных чисел (по умолчанию). | **int value = 255;**  **cout <<noshowpos<< value << endl;** | **255** |
| **scientific** | Вывод чисел с плавающей точкой в экспоненциальной форме | **double value = 1024.165;**  **cout << scientific << value << endl;** | **1.024165e+003** |
| **fixed** | Вывод чисел с плавающей точкой в фиксированной форме (по умолчанию). | **double value = 1024.165;**  **cout << fixed << value << endl;** | **1024.165** |
| **setw(int number)** | Установить ширину поля, где number — количество позиций, символов (выравнивание по умолчанию по правой границе). Манипулятор с параметром. | **cout << setw(40) << «cppstudio.com» << endl;** | **\_\_cppstudio.com** |
| **right** | Выравнивание по правой границе(по умолчанию). Сначала необходимо установить ширину поля(ширина поля должна быть заведомо большей чем, длина выводимой строки). | **cout << setw(40) << right << «cppstudio.com» << endl;** | **\_\_cppstudio.com** |
| **left** | Выравнивание по левой границе. Сначала необходимо установить ширину поля(ширина поля должна быть заведомо большей чем, длина выводимой строки). | **cout << setw(40) << left << «cppstudio.com» << endl;** | **cppstudio.com\_\_** |
| **setprecision(int count)** | Задаёт количество знаков после запятой, где count — количество знаков после десятичной точки | **cout << fixed << setprecision(3) << (13.5 / 2) << endl;** | **6.750** |
| **setfill(int symbol)** | Установить символ заполнитель. Если ширина поля больше, чем выводимая величина, то свободные места поля будут наполняться символом symbol— символ заполнитель | **cout << setfill(‘0′) << setw(4) << 15 << ends << endl;** | **0015** |
|  | | | | |
|  |  | |  |  |

|  |
| --- |
| Таблица 2 — Манипуляторы форматирования в С++ |
|  |