COP 3014: Fall 2021

Stream Formatting - with Flags and Manipulators

November 29, 2021

1 Member functions and flags

Output streams (class ostream and related classes) have some useful member functions for controlling output formatting. Note that these can be used not only with cout, but with other types of output streams. (We'll learn about file output streams soon).

- **setf()** the "set flags" function. Takes as a parameter the flag to be turned "on". Some of the flags that can be turned on or off are:
 - ios::fixed to specify that floating-point numbers will be printed in fixed notation.
 - ios::scientific to specify that floating-point numbers will be printed in scientific (exponential) notation.
 - ios::showpoint specifies that the decimal point will always be printed for floating point types (even if the value is a whole number, like 4.0
 - ios::right right-justifies an output item in a field, if a field width is specified
 - ios::left left-justifies an output item in a field, if a field width is specified
 - See the table below for more formatting flags
- unsetf() the "unset flags" function. Call this to turn off one of the flags
- **precision()** sets the precision for floating-point values to a specific number of significant digits after the decimal point. Takes that number as a parameter
- width() used to specify the "field width" for the next item that is output. Number of character positions is specified as a parameter. Left and right justify flags will apply when this function is used to specify field widths. Extra "space" in the field will be filled with a fill character, which is set to a space by default:

```
int x = 1234;
cout.setf(ios::right);
cout.width(10);
cout << "Hello";
cout.width(15);
cout << x;
// output of the above is:
// Hello 1234
```

• fill() – used to specify the fill character to be used to pad out extra space in a field (when using width()). Takes the character as a parameter.

```
int x = 1234;
cout.setf(ios::right);
cout.fill('.'); // change the fill character
```

```
cout.width(10); // set field width to 10
cout << x; // print x
// output of the above is:</pre>
```

```
// .....1234
```

2 Stream Manipulators

- A stream manipulator is a symbol or function that is used by placing it on the right side of the insertion operator <<.
 - A plain manipulator is just a symbol, like a variable: cout << endl; // endl is a stream manipulator</p>
- A parameterized stream manipulator looks like a function call it has one or more parameters: cout << setw(10); // setw() is a parameterized manipulator
- To use parameterized stream manipulators, you need to include the <iomanip>library #include <iomanip>
- Many of the stream manipulators are just alternate ways of doing tasks performed by member functions. A nice benefit is that cascading can be used, intermixing manipulators and other output statements that use the insertion operator cout << setw(10) << "Hello" << endl;
- setprecision() is a parameterized stream manipulator that performs the same task as the member function precision()
 cout.precision(2); // sets decimal precision to 2 significant digits
 cout << setprecision(2); // does the same thing!
- setw() is a parameterized stream manipulator that performs the same task as the member function width()
 cout.width(10); // sets field width to 10 for next output
 cout << setw(10); // does the same thing!
- setfill() is a parameterized stream manipulator that performs the same task as the member function fill()
 cout.fill('*'); // sets fill character to '*'
 cout << setfill('*'); // does the same thing!
- setiosflags() is a parameterized stream manipulator that performs the same task as the member function setf()
 cout.setf(ios::left); // sets left justification flag
 cout << setiosflags(ios::left); // does the same thing!
- There are also some newer stream manipulators that correspond to some of the formatting flags. For example:
 cout.setf(ios::left); // sets left justification for cout
 cout << left; // also sets left justification for cout
- Caution: Some of these manipulators that correspond to formatting flags were introduced in a newer version of the <iomanip>library, just a few years ago. Some older compilers (still in use) may not recognize them!
- More stream manipulators will be given in a table below, along with the corresponding member functions and/or formatting flags

3 Common Stream Flags and Manipulators

Here is a chart of common stream flags and corresponding stream manipulators (non-parameterized, and all from namespace std).

Flag Name	Corresponding	Description
	Stream Manipulator	
ios::fixed	fixed	if this is set, floating point numbers are printed in fixed
		point notation. When this flag is set, ios::scientific is
		automatically unset
ios::scientific	scientific	if this is set, floating point numbers are printed in scientific
		(exponential) notation. When this flag is set, ios::fixed is automatically unset
iosshowpoint	showpoint	if this is set, the decimal point is always shown, even if
lospoint		there is no precision after the decimal. Can be unset with
		the manipulator noshowpoint
ios::showpos	showpos	if set, positive values will be preceded by a plus sign $+$.
-	-	Can be unset with the manipulator noshowpos .
ios::right	right	if this is set, output items will be right-justified within the
		field (when using width() or setw()), and the unused spaces
		filled with the fill character (the space, by default).
ios::left	left	if this is set, output items will be left-justified within the
		field (when using width() or setw()), and the unused spaces
		filled with the fill character (the space, by default).
ios::showbase	showbase	Specifies that the base of an integer be indicated on the
		output. Decimal numbers have no prefix. Octal numbers
		(base 8) are prefixed with a leading 0. Hexadecimal numbers
		(base 16) are prefixed with a leading 0x. This setting can be
		reset with the manipulator noshowbase .
ios::uppercase	uppercase	specifies that the letters in hex outputs (a-f) and the letter 'e'
		in scientific notation will be output in uppercase. This can be
		reset with the manipulator nouppercase .

Here is a table of other common stream manipulators, all from namespace std

Manipulator	Description		
flush	causes the output buffer to be flushed to the output device before processing proceeds		
endl	prints a newline and flushes the output buffer		
dec	causes integers to be printed in decimal (base 10)		
oct	causes integers from this point to be printed in octal (base 8)		
hex	causes integers from this point to be printed in hexadecimal (base 16)		
setbase()	a parameterized manipulator that takes either 10, 8, or 16 as a parameter, and causes integers to be printed in that base. setbase(16) would do the same thing as hex, for example		
internal	if this is set, a number's sign will be left-justified and the number's magnitude will be right-justified in a field (and the fill character pads the space in between). Only one of right, left, and internal can be set at a time.		
boolalpha	causes values of type bool to be displayed as words (true or false)		
noboolalpha	causes values of type bool to be displayed as the integer values 0 (for false) or 1 (for true)		