Students:
Section 2.22 is a part of 1 assignment: **CSC108 CH02.11-2.24 C2B**
This assignment's due date has passed. Activity will still be recorded, but will not count towards this assignment (unless the due date is changed). See this article for more info.

Includes: CA
Due: 02/06/2025, 11:59 PM EST

2.22 Output formatting

Floating-point manipulators

A programmer can adjust the way that a program's output appears, a task known as output formatting. The main formatting approach uses manipulators. A ***manipulator*** is a function that overloads the insertion operator << or extraction operator >> to adjust the way output appears. Manipulators are defined in the iomanip and ios libraries in namespace std.

PARTICIPATION ACTIVITY | 2.22.1: Floating-point manipulators.

Start 2x speed

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

int main() {
   double miles = 765.4261;

   cout << "setprecision(p) sets # digits" << endl;
   cout << miles << " (default p is 6)" << endl;
   cout << setprecision(8) << miles << " (p = 8)" << endl;
   cout << setprecision(5) << miles << " (p = 5)" << endl;
   cout << setprecision(2) << miles << " (p = 2)" << endl;
   cout << miles << endl << endl;

   // fixed uses fixed point notation
   cout << fixed;
   cout << "fixed: " << miles << endl;

   // scientific uses scientific notation
   cout << scientific;
   cout << "scientific: " << miles << endl;

   return 0;
}
```

```
setprecision(p) sets # digits
765.426 (default p is 6)
765.4261 (p = 8)
765.43 (p = 5)
7.7e+02 (p = 2)
7.7e+02

fixed: 765.43
scientific: 7.65e+02
```

Captions ^

1. The iostream and iomanip libraries define several floating-point manipulators.
2. When a floating-point number is output, the default number of digits to display is 6.
3. The setprecision() manipulator changes the total number of digits to display. Scientific notation (e with a power of 10) may result for smaller precision values.
4. setprecision() affects all subsequent floating-point number output, not just the next output.
5. The fixed manipulator uses fixed-point notation, and the precision (2) now applies to the number of decimal places.
6. The scientific manipulator turns scientific notation on.

Feedback?

Table 2.22.1: Floating-point manipulators.

Manipulator	Description	Example
fixed	Use fixed-point notation. From <iostream>	`// 12.340000` `cout << fixed << 12.34;`
scientific	Use scientific notation. From <iostream>	`// 1.234000e+01` `cout << scientific << 12.34;`

setprecision(p)	If stream has not been manipulated to fixed or scientific:
	Sets max number of digits in number
		`// 12.3`
		`cout << setprecision(3) << 12.34;`
		`// 12.34`
		`cout << setprecision(5) << 12.34;`

	If stream has been manipulated to fixed or scientific:
	Sets max number of digits in fraction only (after the decimal point).
	From <iomanip>
		`// 12.3`
		`cout << fixed << setprecision(1) << 12.34;`
		`// 1.2e+01`
		`cout << scientific << setprecision(1) << 12.34;`

showpoint	Even if fraction is 0, show decimal point and trailing 0s.
	Opposite is noshowpoint.
	From <iostream>
		`// 99`
		`cout << setprecision(3) << 99.0;`
		`// 99.0`
		`cout << setprecision(3) << showpoint << 99.0;`

Feedback?

Manipulators are always meant to be used with the << and >> operators. *A common error is to have a statement like* `setprecision(2);` *rather than* `cout << setprecision(2);`*, which compiles fine but does not impact cout.* Details on operator overloading are presented elsewhere in this material.

PARTICIPATION ACTIVITY | 2.22.2: Output formatting for floating-point manipulators.

Use the code below to answer each question, and assume no manipulators have previously been applied.

```
double temp;
temp = 98.63;
// a floating-point manipulator
cout << temp;
```

1) Which causes the output to appear as "98.6"?
- `cout << fixed;`
- `cout << setprecision(3);`
- `cout << setprecision(2);`
- `cout << scientific << setprecision(2);`

2) Which causes the output to appear as "9.86e+01"?
- `cout << fixed;`
- `cout << setprecision(3);`
- `cout << setprecision(2);`
- `cout << scientific << setprecision(2);`

3) Which causes the output to appear as "99"?
- `cout << fixed;`
- `cout << setprecision(3);`
- `cout << setprecision(2);`
- `cout << scientific << setprecision(2);`

Feedback?

Text-alignment manipulators

Some manipulators help align output.

PARTICIPATION ACTIVITY | 2.22.3: Text-alignment manipulators.

Start 2x speed

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

int main() {
   // Dog age in human years (dogyears.com)
   cout << setw(10) << left  << "Dog age" << "|";
   cout << setw(12) << right << "Human age" << endl;

   // Produce long line
   cout << setfill('-') << setw(23) << "" << endl;

   // Reset fill character back to space
   cout << setfill(' ');

   cout << setw(10) << left  << "2 months" << "|";
   cout << setw(12) << right << "14 months" << endl;
   cout << setw(10) << left  << "6 months" << "|";
   cout << setw(12) << right << "5 years" << endl;
   cout << setw(10) << left  << "8 months" << "|";
   cout << setw(12) << right << "9 years" << endl;
   cout << setw(10) << left  << "1 year" << "|";
   cout << setw(12) << right << "15 years" << endl;

   // Produce long line
   cout << setfill('-') << setw(23) << "" << endl;

   return 0;
}
```

10 chars 12 chars

```
Dog age   |   Human age
-----------------------
2 months  |  14 months
6 months  |    5 years
8 months  |    9 years
1 year    |   15 years
-----------------------
```

Captions ^

1. The setw() manipulator sets the number of characters for displaying the following item to 10. The left manipulator outputs the following item "Dog age" left-aligned within 10 characters.
2. The vertical bar | is output at the end of the 10 characters. setw() only affects the "Dog age" output.
3. The setw() and right manipulators output "Human age" right-aligned within 12 characters.
4. The setfill() manipulator sets the dash character to fill the empty space created when outputting an empty string in the next 23 characters.
5. setfill(' ') sets the empty space character back to the default space character.
6. The dog age values are output left-aligned within 10 spaces, and the human age values are output right-aligned within 12 spaces.
7. One final line of 23 dashes is output.

Feedback?

Table 2.22.2: Text-alignment manipulators.

Manipulator	Description	Example

setw(n)	Sets the number of characters for the next output item only (does not persist, in contrast to other manipulators).
	By default, the item will be right-aligned, and filled with spaces.
	From <iomanip>
		`// "    Amy"`
		`// " George"`
		`cout << setw(7) << "Amy" << endl;`
		`cout << setw(7) << "George" << endl;`

setfill(c)	Sets the fill to character c. From <iomanip>	`// "****Amy"` `cout << setfill('*') << setw(7) << "Amy";`
left	Changes to left alignment. From <iostream>	`// "Amy    "` `cout << left << setw(7) << "Amy";`
right	Changes back to right alignment. From <iostream>	`// "    Amy"` `cout << right << setw(7) << "Amy";`

Feedback?

PARTICIPATION ACTIVITY | 2.22.4: Output formatting for text manipulators.

Use the code below to answer each question, and assume no manipulators have previously been applied.

```
string str = "Amy";
```

1) Which statement prints `"...Amy"`?
- `cout << setfill('.') << str;`
- `cout << setw(6) << setfill('.') << str;`
- `cout << setw(6) << "." << str;`
- `cout << right << setw(6) << str;`

2) Which statement prints `"Amy"`?
- `cout << setfill('.') << str;`
- `cout << setw(6) << setfill('.') << str;`
- `cout << setw(6) << "." << str;`
- `cout << right << setw(6) << str;`

3) Which statement prints `"   Amy"`?
- `cout << setfill('.') << str;`
- `cout << setw(6) << setfill('.') << str;`
- `cout << setw(6) << "." << str;`
- `cout << right << setw(6) << str;`

Feedback?

Buffer manipulators

Printing characters from the buffer to the output device (e.g., screen) requires a time-consuming reservation of processor resources. Once the resources are reserved, moving characters is fast, whether there is 1 character or 50 characters to print.

To preserve resources, the system may wait until the output buffer is full, or at least has a certain number of characters, before moving the characters to the output device. Or, with fewer characters in the buffer, the system may wait until the resources are not busy. Sometimes a programmer does not want the system to wait. Ex: In a very processor-intensive program, waiting could cause delayed and/or jittery output.

Two manipulators exist to send all buffer contents to the output device without waiting: endl and flush.

Table 2.22.3: Buffer manipulators.

Manipulator	Description	Example
endl	Inserts a newline character '\n' into the output buffer and informs the system to flush the buffer. From <iostream>	`// Insert newline and flush` `cout << endl;`
flush	Informs the system to flush the buffer. From <iostream>	`// Flush buffer` `cout << flush;`

Feedback?

A common error is to assume that a cout statement is never reached because the output had not been flushed when the program crashed. Ex: If the program crashes on the statement `cout << "value is " << someFunction();`, the words "value is" will not output because the buffer was not flushed.

Exploring further:

- More manipulators exist. See cplusplus.com

PARTICIPATION ACTIVITY | 2.22.5: Buffer manipulators.

1) The text "test" is likely to immediately display on the screen.

```
cout << "test";
```

- True
- False

2) The text "test" and "this" are likely to immediately display on the screen.

```
cout << "test" << endl << "this\n";
```

- True
- False

3) The text "test" is likely to immediately display on the screen.

```
cout << "test" << flush;
```

- True
- False

Feedback?

CHALLENGE ACTIVITY | 2.22.1: Enter the formatted output.

620890.5010016.qx3zqy7

Start

Type the program's output

```
#include <iostream>
#include <iomanip>

using namespace std;

int main() {
   float myFloat;

   myFloat = 50.8423;

   cout << setprecision(5) << myFloat << endl;
   cout << setprecision(6) << myFloat << endl;

   return 0;
}
```

```
50.842
50.8423
```

1 2 3 4 5 6

Check Next

Feedback?

CHALLENGE ACTIVITY | 2.22.2: Output formatting.

620890.5010016.qx3zqy7

Start

Double areaMeasured is read from input. Output areaMeasured with a maximum of eight digits, showing the decimal point and any trailing 0s. End with a newline.

Ex: If the input is `39.75`, then the output is:

```
39.750000
```

Note: `cout << showpoint` shows the decimal point and trailing 0s.

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

int main() {
   double areaMeasured;

   cin >> areaMeasured;

   /* Your code goes here */

   return 0;
}
```

1 2 3

Check Next level

Feedback?

How was this section? Provide section feedback

Activity summary for assignment: CSC108 CH02.11-2.24 C2B

51 / 51 points
51 / 51 points submitted to BlackboardLearn

Due: 02/06/2025, 11:59 PM EST
This assignment's due date has passed. Activity will still be recorded, but will not count towards this assignment (unless the due date is changed). See this article for more info.

Completion details