Input / Output Basics

Last updated 12/3/19

Input / Output

- Stream
  - A stream is a flow of information
  - An io stream allows:
    - Information to be entered into a program
    - Information to be displayed by a program
    - Information to be read/written to a file by the program
    - Errors to be displayed by a program

  - In C – we used stdin, stdout, and stderr
    - printf() and scanf() managed these for us so we never really used them

Input / Output

- Stream
  - In C++ we will use cout, cin, cerr, and clog
    - These are more basic than printf and scanf, but also more versatile
    - Require inclusion of iostream
    - Require use of a name space (std)
      - A name space defines the interpretation of names in a specific manner
      - cout, cin, …

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

  - No error checking of the stream
    - See the third example

Input / Output

- Stream
  - 4 stream channels
    - Input channel – cin – typically a keyboard/file
    - Output channel – cout – typically a display/file
    - Error output channel – cerr – typically a display/file
    - Log output channel – clog – typically a display/file
  - 2 stream operators
    - Stream extraction operator >>
      - Extracts data from the stream proceeding it and places it into the entity following it
      - Any "white" space is considered a terminator (sp, tab, line feed, ...)
    - Stream insertion operator <<
      - Inserts the data that follows it into the stream that precedes it
      - No implicit white space added

Input / Output

- Stream
  - Output to a terminal
    cout << {string, variable, object, …}
  - Input from a keyboard
    cin >> {string, variable, object}
  - endl
    - Similar to \n
      - Creates a new line
      - Flushes the buffer
      - cout << … << endl;

Input / Output

- Escape sequences
  - Sequences to include special characters into the stream

\a	Alert (Beep, Bell)
\b	Backspace
\f	Formfeed Page Break
\n	Newline (Line Feed)
\r	Carriage Return
\t	Horizontal Tab
\v	Vertical Tab
\\	Backslash
\'	Single quotation mark
\"	Double quotation mark

note: \n still here

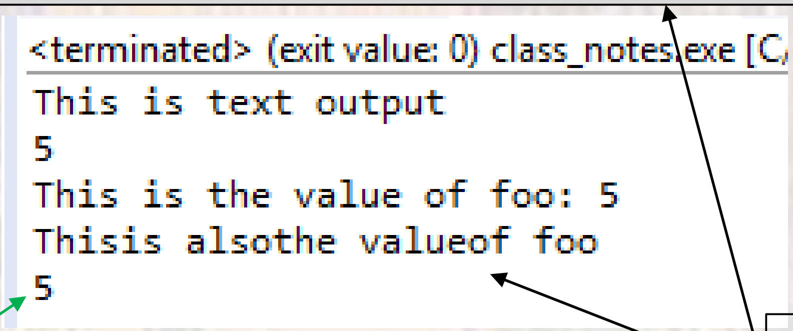
Input / Output

- Stream Example 1 - output

cout_cin.cpp

```
int foo;
foo = 5;

cout << "This is text output\n";
cout << foo;
cout << endl;
cout << "This is the value of foo: " << foo << endl;
cout << "This" << "is also" << "the value" << "of foo\n" << foo << endl;
```



```
<terminated> (exit value: 0) class_notes.exe [C,
This is text output
5
This is the value of foo: 5
Thisis alsothe valueof foo
5
```

Why is this on a new line?

Note – the insertion operator does not provide spaces

Input / Output

- Stream Example 2 – input/output

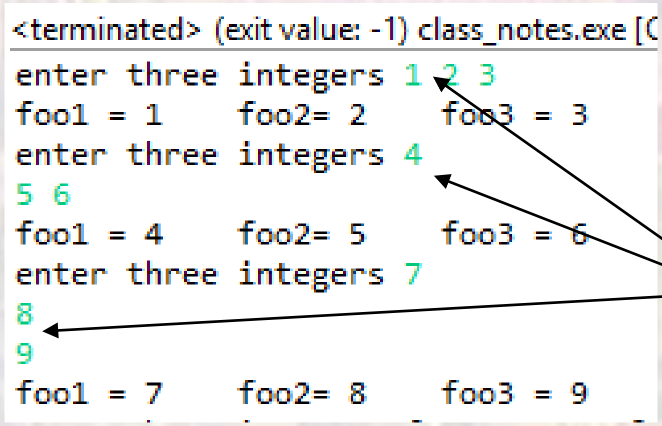
cout_cin.cpp

```
int foo1;
int foo2;
int foo3;

while(1){
   cout << "enter three integers ";

   cin >> foo1;
   cin >> foo2 >> foo3;

   cout << "foo1 = " << foo1 << "    foo2= " << foo2 << "    foo3 = " << foo3;
   cout << endl;
} // end while
```



```
<terminated> (exit value: -1) class_notes.exe [C
enter three integers 1 2 3
foo1 = 1     foo2= 2     foo3 = 3
enter three integers 4
5 6
foo1 = 4     foo2= 5     foo3 = 6
enter three integers 7
8
9
foo1 = 7     foo2= 8     foo3 = 9
```

Note – whitespace used as delimiter

Input / Output

- Stream Example 3 – input/output

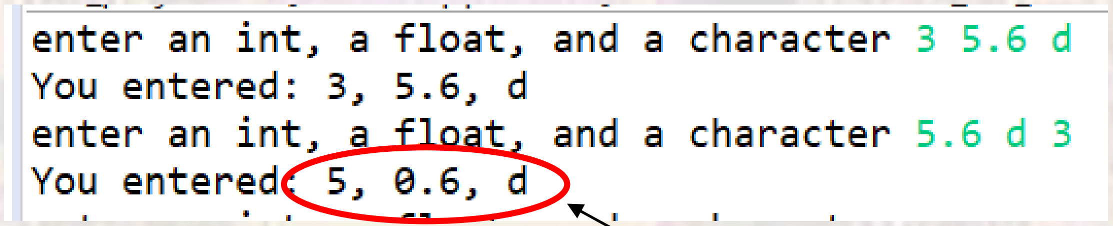
cout_cin.cpp

```
int foo1;
float foo2;
char foo3;

while(1){
   cout << "enter an int, a float, and a character ";

   cin >> foo1 >> foo2 >> foo3;

   cout << "You entered: " << foo1 << ", " << foo2 << ", " << foo3;
   cout << endl;
} // end while
```



```
enter an int, a float, and a character 3 5.6 d
You entered: 3, 5.6, d
enter an int, a float, and a character 5.6 d 3
You entered: 5, 0.6, d
```

Looking for an int – stops reading at the .
The .6 is still in the buffer so it reads in as the float