Input / Output Stream Object

Last updated 12/5/18

Stream Objects

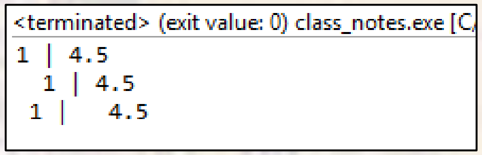
- cout and cin are actually C++ objects (stream Class)
  - Include a series of manipulators and methods (functions)
  - Manipulators modify the content of the stream
    - Require #include <iomanip>
  - Methods operate on the contents of the stream

Stream Objects

- cout manipulators
  - setw()
    - Sets the width used to print out the next value
    - When necessary – pads with leading spaces
    - Parameter must be an int

stream_objects.cpp

```
int foo;
double boo;
foo = 1;
boo = 4.5;
cout << foo << " | " << boo << endl;
cout << setw(3);
cout << foo << " | " << boo << endl;
cout << setw(2) << foo << " | " << setw(5) << boo << endl;
```



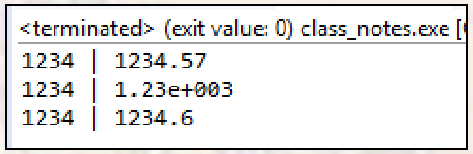
```
<terminated> (exit value: 0) class_notes.exe [C
1 | 4.5
  1 | 4.5
 1 |   4.5
```

Stream Objects

- cout manipulators
  - setprecision()
    - Sets the precision used to print out floating point values
    - # of digits printed (excluding the decimal point)
    - Parameter must be an int
    - Remains in effect until changed

stream_objects.cpp

```
int foo;
double boo;
foo = 1234;
boo = 1234.5678;
cout << foo << " | " << boo << endl;
cout << setprecision(3);
cout << foo << " | " << boo << endl;
cout << setprecision(5) << foo << " | "  << boo << endl;
```



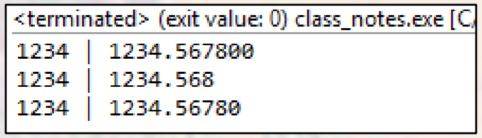
```
<terminated> (exit value: 0) class_notes.exe [
1234 | 1234.57
1234 | 1.23e+003
1234 | 1234.6
```

Stream Objects

- cout manipulators
  - fixed
    - Forces numbers to be printed in fixed point form vs. scientific notation

stream_objects.cpp

```
int foo;
double boo;
foo = 1234;
boo = 1234.5678;
cout << fixed;
cout << foo << " | " << boo << endl;
cout << setprecision(3);
cout << foo << " | " << boo << endl;
cout << setprecision(5) << foo << " | "  << boo << endl;
```



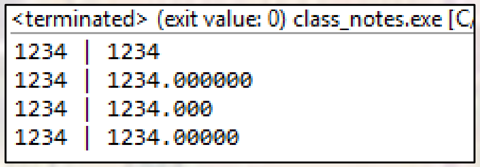
```
<terminated> (exit value: 0) class_notes.exe [C
1234 | 1234.567800
1234 | 1234.568
1234 | 1234.56780
```

Stream Objects

- cout manipulators
  - showpoint
    - Forces numbers to be printed with the decimal point showing

stream_objects.cpp

```cpp
int foo;
double boo;
foo = 1234;
boo = 1234;
cout << foo << " | " << boo << endl;
cout << fixed << showpoint;
cout << foo << " | " << boo << endl;
cout << setprecision(3);
cout << foo << " | " << boo << endl;
cout << setprecision(5) << foo << " | "  << boo << endl;
```



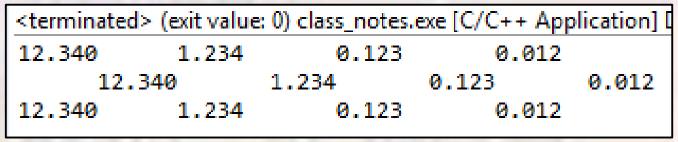
```
<terminated> (exit value: 0) class_notes.exe [C
1234 | 1234
1234 | 1234.000000
1234 | 1234.000
1234 | 1234.00000
```

Stream Objects

- cout manipulators
  - left and right
    - Forces left or right alignment
    - Maintains sense until changed
    - Default is right

stream_objects.cpp

```
double foo;
foo = 12.34;
cout << fixed << showpoint << setprecision(3);
cout << left;
cout << setw(12) << foo << setw(12) << foo/10 << setw(12) << foo/100 << setw(12) << foo/1000 << endl;
cout << right;
cout << setw(12) << foo << setw(12) << foo/10 << setw(12) << foo/100 << setw(12) << foo/1000 << endl;
cout << left;
cout << setw(12) << foo << setw(12) << foo/10 << setw(12) << foo/100 << setw(12) << foo/1000 << endl;
```



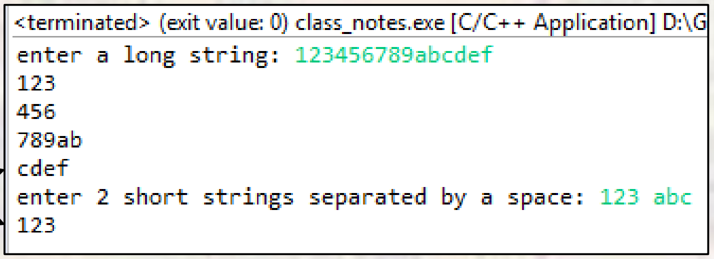
```
<terminated> (exit value: 0) class_notes.exe [C/C++ Application]
12.340      1.234       0.123       0.012
      12.340       1.234       0.123       0.012
12.340      1.234       0.123       0.012
```

Stream Objects

- cin manipulator
  - setw()
    - Read in only the specified number of characters
    - If too few characters entered, cin stops reading at any white space

stream_objects.cpp

```
string foo;
cout << "enter a long string: ";
cin >> setw(3) >> foo;
cout << foo << endl;
cin >> setw(3) >> foo;
cout << foo << endl;
cin >> setw(5) >> foo;
cout << foo << endl;
cin >> setw(5) >> foo;
cout << foo << endl;
cout << "enter 2 short strings separated by a space: ";
cin >> setw(8) >> foo;
cout << foo << endl;
```



```
<terminated> (exit value: 0) class_notes.exe [C/C++ Application] D:\G
enter a long string: 123456789abcdef
123
456
789ab
cdef
enter 2 short strings separated by a space: 123 abc
123
```

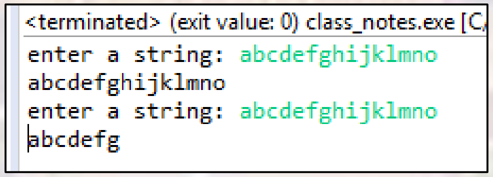
terminated by the white space

Stream Objects

- cin member functions
  - cin.getline(name of array to store line, #of characters to read + 1)

stream_objects.cpp

```
char foo[20];
cout << "enter a string: ";
cin.getline(foo, 20);
cout << foo << endl;
cout << "enter a string: ";
cin.getline(foo, 8);
cout << foo;
```



<terminated> (exit value: 0) class_notes.exe [C

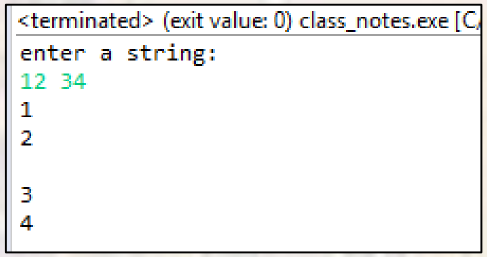
```
enter a string: abcdefghijklmno
abcdefghijklmno
enter a string: abcdefghijklmno
abcdefg
```

Stream Objects

- cin member functions
  - cin.get(name of variable to store character)
  - Reads in a single character
    - Includes whitespace

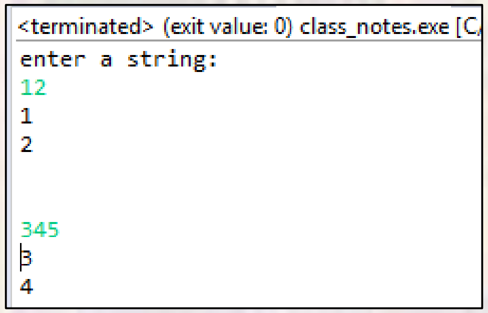
stream_objects.cpp

```
char foo;
cout << "enter a string: " << endl;
cin.get(foo);
cout << foo << endl;
cin.get(foo);
cout << foo << endl;
cin.get(foo);
cout << foo << endl;
cin.get(foo);
cout << foo << endl;
cin.get(foo);
cout << foo << endl ;
```



```
<terminated> (exit value: 0) class_notes.exe [C
enter a string:
12 34
1
2

3
4
```



```
<terminated> (exit value: 0) class_notes.exe [C
enter a string:
12
1
2


345
3
4
```

Note: the return causes 2 lines
1 for the next output and 1 for
the return (LF)

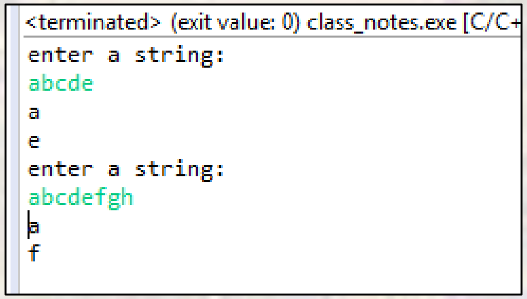
Stream Objects

- cin member functions
  - cin.ignore (number of characters to ignore)
  - cin.ignore(number of characters to ignore, character to stop reading at (if reached))

stream_objects.cpp

```
char foo;
cout << "enter a string: " << endl;
cin.get(foo);
cout << foo << endl;
cin.ignore(3);
cin.get(foo);
cout << foo << endl;
cin.ignore();  //ignore the CR

cout << "enter a string: " << endl;
cin.get(foo);
cout << foo << endl;
cin.ignore(20,'e');
cin.get(foo);
cout << foo;
```



<terminated> (exit value: 0) class_notes.exe [C/C+

```
enter a string:
abcde
a
e
enter a string:
abcdefgh
a
f
```