Simple I/O using iostream library

Use of iostream (cin/cout) instead of cstdlib (scanf/printf)

- syntax is very different, but behaviour is similar
- iostream and the std namespace
- basic use of cout
- endl vs \n for newlines
- basic use of cin
- formatting cout for field widths
- formatting cout floats for precision

using namespace std

- C++ allows definitions to be grouped into “namespaces”
- allows us to specify what sets of definitions to use
- std is a widely used namespace in the C++ libraries
- for now, to utilize this namespace we'll use:

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

- this has drawbacks, but we'll discuss those with more info on namespaces later in the term

output using cout

- cout is part of iostream library, defined in std namespace
- syntax is very different than printf
- examples of printing with cout

```
int x;
float y
cout << “Here is x: “ << x << “, and y: “ << y;
```

- the << are placed before each item to be displayed

endl instead of \n

- we can still use \n (inside double quotes) for newlines
- a special keyword, endl, defined by iostream within std
- to print a newline we can use

```
cout << endl;
cout << “the value is “ << x << endl;
```

- can put multiple newlines to get blank lines of output, e.g.

```
cout << endl << endl << endl;
```

input using cin

- the input counterpart to cout
- uses >> rather than <<
- reads and stores user input into a variable

```
int x;
float f;
cin >> x;
cin >> f;
```

- handles reading/whitespace much like scanf

formatted width in cout

- iomanip library needs to be included for formatting

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

- provides a setw routine to specify width of next field

```
cout << setw(6) << x; // pads x to 6 chars width
```

- actual padding behaves much like in printf

floating point precision in cout

- iomanip also contains routine to turn on fixed-precision formatting of floats

```
cout << setiosflags(ios::fixed);
```

- with fixed precision turned on, we use setprecision to specify precision for an output value

```
float f = 123.456;
cout << setiosflags(ios::fixed);
cout << setprecision(2) << f;  // 2 digits after .
```