CS 32 – Summer 2021 E. Ambrosio

Stringstreams

Sometimes you have information that you would like to present in a way that would be trivial if it were delivered to the screen using cout. For example, suppose we want to write a double in the conventional way of representing a U.S. monetary amount:

```
double amt;
... suppose some code here gives amt the value 123.4
cout.setf(ios::fixed);
cout.precision(2);
cout << "$" << amt;</pre>
```

(The middle two statements set the state of cout so that all doubles written from now on will be in fixed instead of scientific notation, and will display two digits to the right of the decimal point.)

But suppose instead that we didn't want to write the sequence of characters "\$123.40" to the screen, but instead wanted to capture it into a string that we could do something with later. Here's an easy way to do this:

The type name ostringstream denotes an output stringstream. We created one that we chose to call oss. Just as an output stream like cout uses the screen as destination for characters, and an ofstream (an output file stream) uses a file as a destination for characters, an ostringstream uses its own internal storage as a destination for characters. You can later retrieve those characters as a string.

When we created oss, its internal storage is empty. The operations we can perform on an ostringstream are the same ones we can perform on cout or an ofstream. To retrieve the characters later as a string, use the ostringstream's member function str(). Here's another example:

CS 32 – Summer 2021 E. Ambrosio

The string s now contains "----\n 123\n". (The setw(5) says that the next thing written (a number or a string) will be written in a field 5 characters wide, so there are two spaces before the 123.) Notice that we don't have to write everything to the ostringstream at one time; we can write something, then later on something additional, etc. This can make it easier to use than the C library's sprintf function.

Suppose we're writing a value in a field wider than we need to represent the value (e.g., writing 123 in a field more than 3 characters wide). By default, the extra characters will be blanks, but we can change that by setting the stream's fill character:

```
ostringstream oss;
int k;
... suppose some code here gives k the value 123
oss << setw(5) << k << endl;
oss.fill('*');
oss << setw(5) << k << endl;
oss << setw(4) << k << endl;
oss.fill('0');
oss << setw(5) << k << endl;
string s = oss.str();</pre>
```

The string s now contains " $123\n**123\n*123\n"$.

There is also the concept of an input stringstream, an istringstream, but this isn't useful for Project 4.