C++ I/O refresher

`iostream` is the main C++ library for input and output

```
#include <iostream>

using std::cin;    // default input stream
using std::cout;   // default output stream
using std::endl;   // end of line, flushes buffer
```

also

```
using std::cerr;   // default error output stream
```

`<<` is the stream insertion operator; used for output

`>>` is the stream extraction operator; used for input

C++ File I/O

- In C, printf wrote to stdout and scanf read from stdin
- `fprintf` and `fscanf` were their counterparts for files
- In C++, we have `std::cout` and `std::cin`
- `std::ofstream` and `std::ifstream` are their counterparts for files
- These are defined in the file-stream header: `#include <fstream>`
  - `ofstream`: for writing to a file
  - `ifstream`: for reading from a file
  - `fstream`: for reading and writing to/from a file
- we still use `<<` and `>>` operators for file I/O

C++ osftream usage

io1.cpp:

```
#include <iostream>
#include <fstream>
int main(){
        std::ofstream ofile( "hello.txt" );
        ofile << "Hello, World!" << std::endl;
        return 0;
}
```

```
$ g++ -std=c++11 -pedantic -Wall -Wextra -c io1.cpp
$ g++ -o io1 io1.o
$ ./io1
$ cat hello.txt
Hello, World!
```

C++ istream usage

io2.cpp:

```
#include <iostream>
#include <fstream>
#include <string>
int main(){
        std::ifstream ifile( "hello.txt" );
        std::string word;
        while( ifile >> word )
                std::cout << word << std::endl;
        return 0;
}
```

```
$ g++ -std=c++11 -pedantic -Wall -Wextra -c io2.cpp
$ g++ -o io2 io2.o
$ ./io2
Hello,
World!
```

C++ I/O from/to strings

`std::stringstream`

Instead of reading or writing to console or file, it reads and writes to a temporary string ("buffer") stored inside

io3.cpp:

```
#include <iostream>
#include <sstream>
int main(){
        std::stringstream ss;
        ss << "Hello, world!" << std::endl;
        std::cout << ss.str();
        return 0;
}
```

```
$ g++ -std=c++11 -pedantic -Wall -Wextra -c io3.cpp
$ g++ -o io3 io3.o
$ ./io3
Hello, world!
```

C++ stringstream details

- a string buffer that contains a sequence of characters
- `str()` function can be used to get the content of the buffer
- `str(string)` sets the content of the buffer to the string argument
- `<<` and `>>` operators can be used with `stringstream` to insert/extract content

```
std::stringstream ss("ali");
```

C++ another stringstream example

io4.cpp:

```
#include <string>
#include <iostream>
#include <sstream>
int main(){
        std::stringstream ss;
        ss << "Hello" << ' ' << 2019 << " world";
        std::cout << ss.str() << std::endl;
        std::string word1, word2;
        int num;
        ss >> word1 >> num >> word2;
        std::cout << word1 << ", " << word2 << " " << num << '!' << std::endl;
        return 0;
}
```

```
$ g++ -std=c++11 -pedantic -Wall -Wextra -c io4.cpp
$ g++ -o io4 io4.o
$ ./io4
Hello 2019 world
Hello, world 2019!
```

C++ stringstream differentiation

- Like the filestream, the stringstream also comes in flavors that only do reading or writing:
  - istringstream <–> ifstream
  - ostringstream <–> ostream