Streams

Rupesh Nasre.

OOAIA
January 2025

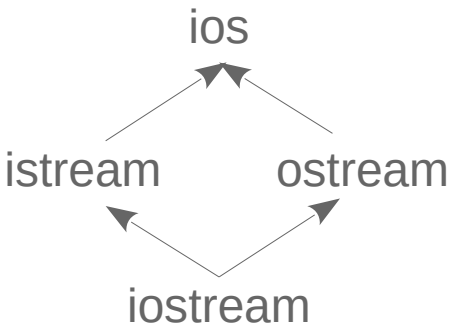
Streams

- Linear queue maintaining data-flow between the source (for input) or the destination (for output) and your program.
- Typically, a sequence of bytes
- Used in
  - Standard input, output
  - Files
  - Strings
  - Custom (e.g., sockets, pipes, etc.)

I/O

- Input stream
  - istream cin
  - Defaults to keyboard / stdin
- Output stream
  - ostream cout
  - Defaults to console / stdout
- Uses header <iostream>
- cin and cout are objects.
  - >> and << are overloaded.

```
#include <iostream>

int main() {
    std::string name;
    std::cout << "Enter your name: ";
    std::cin >> name;
    std::cout << "Hello " << name << "!\n";

    return 0;
}
```



cin

- By default, reads whitespace-separated formatted tokens.
  - But be careful with char and string.
- Example stream: 12 17.3 -19

```
int A, B;
double X;
cin >> A;   12
cin >> X;   17.3
cin >> B;   -19
```

```
int A, B;
char X;
cin >> A;  12
cin >> B;  17
cin >> X;  '.'
cin >> A;  3
```

```
int A;
char B, C, D;
cin >> A;  12
cin >> B;  '1'
cin >> C;  '7'
```

```
string A, B, C;
cin >> A;  "12"
cin >> B;  "17.3"
cin >> C;  "-19"
```

ignore

- cin ignores the leading whitespace.
- One can ignore input until a character of relevance is seen.
  - cin.**ignore**(N, ch); // skip upto N chars or till ch is seen.
  - Useful for ignoring a line (ch = '\n')

```
cout << "Enter your name: ";
cin.ignore(2, 'A');

cin >> name;          // ABCD
cout << name;         // BCD
```

```
cout << "Enter your name: ";
cin.ignore(2, 'Z');

cin >> name;          // ABCD
cout << name;         // CD
```

cout

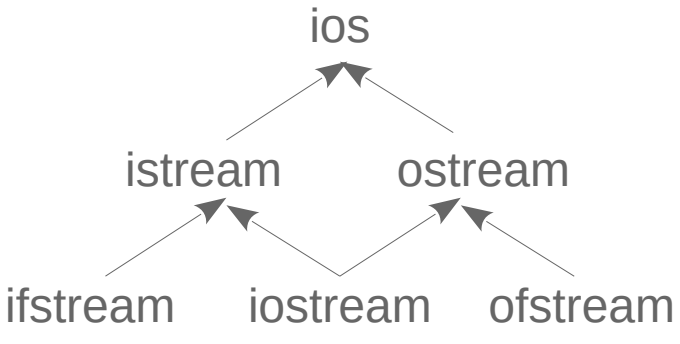
- Uses buffered output
  - \n can be printed with endl.
  - May not print to screen.
  - To force, use cout << flush.
  - endl involves flush.
  - cin involves cout's flush.

```
#include <iostream>

int main() {
    std::string name;
    std::cout << "Enter your name: ";
    std::cin >> name;
    std::cout << "Hello " << name << endl;

    return 0;
}
```

  - *cin* maps to *stdin*.
  - *cout* maps to *stdout*.
  - *cerr*, *clog* map to *stderr*.

File streams

- #include <fstream>
  - ifstream and ofstream



What is the output?

What is the issue?

stream4.cpp

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

int main() {
        ifstream file("stream4.cpp");
        string word;
        while (!file.eof()) {
                file >> word;
                cout << word;
        }

        file.close();   // optional
        return 0;
}
```

#include<iostream>#include<fstream>usingnamespacestd;intmain()
{ifstreamfile("stream4.cpp");stringword;while(!file.eof()){file>>word;cout<<word;}file.close();//
optionalreturn0;}}

ifstream

- **Issue**: Last line is read twice.
- **Solution**: read precedes eof.

stream4.cpp

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

int main() {
        ifstream file("stream4.cpp");
        string word;
        file >> word;
        while (!file.eof()) {
                cout << word;
                file >> word;
        }

        file.close();   // optional
        return 0;
}
```

#include<iostream>#include<fstream>usingnamespacestd;intmain()
{ifstreamfile("stream4.cpp");stringword;while(!file.eof()){file>>word;cout<<word;}file.close();//
optionalreturn0;}

cat

- What is the output?

```
while (getline(file, line)) {
        cout << line;
}
```

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

int main() {
        ifstream file("stream4.cpp");
        string line;
        getline(file, line);
        while (!file.eof()) {
                cout << line;
                getline(file, line);
        }

        file.close();   // optional
        return 0;
}
```

#include <iostream>#include <fstream>using namespace std;int main() { ifstream file("stream4.cpp"); string word; while (!file.eof()) { file >> word; cout << word; } file.close(); // optional return 0;}

cat

- What is the output?

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

int main() {
	ifstream file("stream4.cpp");
	string word;
	while (!file.eof()) {
		file >> word;
		cout << word;
	}

	file.close();	// optional
	return 0;
}
```

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

int main() {
	ifstream file("stream4.cpp");
	string line;
	getline(file, line);
	while (!file.eof()) {
		cout << line << endl;
		getline(file, line);
	}

	file.close();	// optional
	return 0;
}
```

cp

- a.out <inputfile> <outputfile>

```
#include <iostream>
#include <fstream>
#include <stdlib.h>
using namespace std;

int main(int argc, char *argv[]) {
        ifstream iifile(argv[1]);
        ofstream oofile(argv[2]);
        string line;
        getline(iifile, line);
        while (!iifile.eof()) {
                oofile << line << endl;
                getline(iifile, line);
        }

        return 0;
}
```

cp

- a.out <inputfile> <outputfile>

```
#include <iostream>
#include <fstream>

using namespace std;

int main(int argc, char *argv[]) {
	if (argc != 3) {
		cerr << "Usage: " << argv[0] << " <inputfile> <outputfile>" << endl;
		exit(1);
	}
	ifstream iifile(argv[1]);
	ofstream oofile(argv[2]);
	string line;
	getline(iifile, line);
	while (!iifile.eof()) {
		oofile << line << endl;
		getline(iifile, line);
	}

	return 0;
}
```

```
#include <iostream>
#include <fstream>

using namespace std;

int main(int argc, char *argv[]) {
	if (argc != 3) {
		cerr << "Usage: " << argv[0] << " <inputfile> <outputfile>" << endl;
		exit(1);
	}
	ifstream iifile(argv[1]);
	if (iifile.fail()) {
		cerr << "File " << argv[1] << " could not be opened." << endl;
		exit(2);
	}
	ofstream oofile(argv[2]);
	string line;
	getline(iifile, line);
	while (!iifile.eof()) {
		oofile << line << endl;
		getline(iifile, line);
	}

	return 0;
}
```

Appending to a file

- ios::app
  - Creates file if it does not exist.

```
ofstream out("y", ios::app);
out << "This is appended\n";
out.close();
```

```
$ cat y
No such file or directory
$ a.out; cat y
This is appended
$ a.out; cat y
This is appended
This is appended
$
```

Write a program to replace *appended* with *defended*.

Both read and write

- ios::in | ios::out

```
fstream rwfile("y", ios::in | ios::out);
string word;

while (rwfile >> word) {
        if (word == "appended")
                rwfile << "defended";
}
rwfile.close();
```

```
$ cat y
This is appended
This is appended
$ a.out; cat y
This is appendeddefended appendeddefended
```

Write a program to replace *appended* with *defended*.

Both read and write

- ios::in | ios::out

```
fstream rwfile("y", ios::in | ios::out);
string word;

while (rwfile >> word) {
        if (word == "appended") {
                rwfile.seekg(-word.length(), ios::cur);
                rwfile << "defended";
        }
}
rwfile.close();
```

```
$ cat y
This is appended
This is appended
$ a.out; cat y
This is defended
This is defended
```

Formatted I/O

data.txt

```
Name: Roll Number: Marks
John Augustine: CS12D001: 88
Madhu Mutyam: CS11D111: 89
Rupesh Nasre: CS13B000: 25
```

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

int main() {
        ifstream file("data.txt");
        string name, roll, marks;

        getline(file, name, ':');
        while (!file.eof()) {
                getline(file, roll, ':');
                getline(file, marks, ':');

                cout << name << ":::" << roll << ":::" << marks << endl
                getline(file, name, ':');
        }
        return 0;
}
```

```
Name::: Roll Number::: Marks
John Augustine
 CS12D001::: 88
Madhu Mutyam::: CS11D111
 89
Rupesh Nasre::: CS13B000::: 25
```

Formatted I/O

data.txt

```
Name: Roll Number: Marks
John Augustine: CS12D001: 88
Madhu Mutyam: CS11D111: 89
Rupesh Nasre: CS13B000: 25
```

Note: getline does not ignore leading whitespace.

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

int main() {
        ifstream file("data.txt");
        string name, roll, marks;

        getline(file, name, ':');
        while (!file.eof()) {
                getline(file, roll, ':');
                getline(file, marks);

                cout << name << ":::" << roll << ":::" << marks << endl
                getline(file, name, ':');
        }
        return 0;
}
```

```
Name::: Roll Number::: Marks
John Augustine::: CS12D001::: 88
Madhu Mutyam::: CS11D111::: 89
Rupesh Nasre::: CS13B000::: 25
```

Formatted I/O

```
           Name Roll Number     Marks
John Augustine  CS12D001        88
  Madhu Mutyam  CS11D111        89
  Rupesh Nasre  CS13B000        25
```

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

int main() {
        ifstream file("data.txt");
        string name, roll, marks;

        getline(file, name, ':');
        while (!file.eof()) {
                getline(file, roll, ':');
                getline(file, marks);

                cout << setw(20) << name;
                cout << setw(10) << roll;
                cout << setw(10) << marks;
                cout << endl;
                getline(file, name, ':');
        }
        return 0;
}
```

Other functions

- **setprecision**: for floating point values.
- **get**: to read a single character.
- **peek**: to examine a character without removing it from the stream.
- **putback**: add to the stream.

String streams

- #include <sstream>
  - istringstream and ostringstream

```
std::string source = "CS2810 101\n34.5 50";
std::istringstream ss(source);
std::string course;
int nstuds;
float average;
int marks;

std::cout << "ss.str = " << ss.str() << std::endl;
ss >> course >> nstuds >> average >> marks;
std::cout << "Course = " << course << std::endl;
std::cout << "Class size = " << nstuds << std::endl;
std::cout << "Average = " << average << " / " << marks << std::endl;
```

```
ss.str = CS2810 101
34.5 50
Course = CS2810
Class size = 101
Average = 34.5 / 50
```

Acknowledgments

- https://courses.cs.vt.edu/cs1044/Notes/C04.IO.pdf
- cppreference.com