Reading & Writing Data Files

1) What are streams, objects, and classes?
2) How can a program read and write a plain text data file?

About Files

- Volatile Storage
  - Data in main memory (RAM) is "volatile":
    It does..
- Non-volatile Storage
  - Computers store data files on "non-volatile storage".
  - Ex: On the..
- Reading/Writing
  - Programs write to files to save/store data.
  - Programs read from files to load data.

File streams

- C++ treats a file as as a stream:
  - An input stream reads from a file: ifstream
    Use..
  - An output stream writes to a file: ofstream
    Use..
- Console Streams
  - cin is an input stream; cout is an output stream
- File Streams
  - Create an input or output file stream to read or write a file.

File streams

- Streams are Objects
  - Objects store some data and have functions you can call on them.
  - Call member functions on an object with..
  - Example:

```
if (!myFile.is_open()) {
    println("ERROR: Unable to open the file.");
    exit(EXIT_FAILURE);
}
```

is_open() is a member function: it is a function which belongs to the object myFile.

Writing To a File

- ..

```
ofstream myOutStream("test.txt");

if (!myOutStream.is_open()) {
    println("ERROR: Unable to open the file.");
    exit(EXIT_FAILURE);
}
```

- ..

```
println(myOutStream, "Hello world!");
println(myOutStream, "{} = {:.2f}", "pi", 3.1415);
```

- ..

```
myOutStream.close();
```

Write Example: User's values to file

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

int main() {
  // Create the file for output (erasing existing content)
  ofstream fileOut("numbers.txt");

  if (!fileOut.is_open()) {
    println("ERROR opening file.");
    exit(EXIT_FAILURE);
  }

  println("Enter values (-1 to end).");
  bool done = false;
  while (!done) {
    double value;
    print(": ");
    cin >> value;

    if (value == -1) {
      done = true;
    } else {
      println(fileOut, "{:.2f}", value);
    }
  }
}
```

Read full lines

- Often a program reads a whole line of text at once
  - Ex: csv file (comma separated)
- We will read in a file line-by-line using ..

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

int main()
{
  // Open file for reading
  ifstream myStream("test.txt");

  if (!myStream.is_open()) {
    println("Error opening file");
    exit(EXIT_FAILURE);
  }

  // Read each line of text into
  // my `line` variable.
  string line;

  while (getline(myStream, line)) {

    println("Read '{}'", line);
  }
}
```

Read Example: Sum numbers in file

```cpp
// Read in some numbers from a file
#include <print>
#include <fstream>
using namespace std;

int main() {
    // Open file for reading
    ifstream myStream("numbers.txt");
    if (!myStream.is_open()) {
        println("ERROR: Unable to open file.");
        exit(EXIT_FAILURE);
    }

    // Sum up all values in a file
    double sum = 0;
    string line;
    while (getline(myStream, line)) {
        println("Read line '{}'", line);

        // Skip blank lines
        if (line.empty()) {
            continue;
        }

        // String-to-double: stod
        double value = stod(line);
        sum += value;
    }
    println("Sum of values in file: {:.2f}", sum);
}
```

Classes and Objects

- Classes and Objects
  - Class: a data type in C++ which allows you to..
  - Objects are..
    - Example with strings:
      string myName;
      - string is a class (the type)
      - myName is an object of type string.
  - When you create variables of type string
    it.. an object of type string.

3 ways of leaving: Exit, Return, Break

- exit(EXIT_FAILURE);..
  - Argument:
    0: Success (EXIT_SUCCESS)
    1: Failure (EXIT_FAILURE)
  - Defined in cstdlib:
    #include <cstdlib>
- return;
- break;

Summary

- Data files
  - Store information between program executions.
- Objects are instances of classes.
  - They have both data and member functions.
- Streams used to read/write files:
  - ifstream: input file stream
  - ofstream: output file stream
  - Use println() and getline() to work with files.
- Use exit() to immediately exit the program.