



Computer Programming (b) - E1124

(Spring 2021-2022)

Lecture 5

File Input/output

INSTRUCTOR

Dr / Ayman Soliman



➤ Contents

- Introduction to I/O files
- Programming example: student grade



File input/output

➤ File input/output

- Getting input from the **keyboard** and sending output to the **screen** have several limitations.
- Inputting data in a program from the keyboard is comfortable if the amount of input is **very small**.
- Sending output to the screen works well if the amount of data is **small** (no larger than the size of the screen) and you do not want to distribute the output in a printed format to others.

Types of Files

```
graph TD; A[Types of Files] --> B[Text files]; A --> C[Binary files];
```

Text files

Text files are the normal **.txt** files. You can easily create text files using any simple text editors such as Notepad.

When you open those files, you'll see all the contents within the file as plain text. You can easily edit or delete the contents.

They take minimum effort to maintain, are easily readable, and provide the least security and takes bigger storage space

Binary files

Binary files are mostly the **.bin** files in your computer. Instead of storing data in plain text, they store it in the binary form (0's and 1's).

They can hold a higher amount of data, are not readable easily, and provides better security than text files.

➤ **File input/output**

- The standard I/O header file, `iostream`, contains data types and variables that are used only for input from the standard input device and output to the standard output device.
- C++ provides a header file called `fstream`, which is used for **file I/O**.
- Among other things, the `fstream` header file contains the definitions of two data types:
 - ❑ `ifstream`, which means input file stream.
 - ❑ `ofstream` which means output file stream.

➤ **File input/output (cont.)**

➤ File: area in secondary storage to hold info

➤ File I/O is a five-step process

Include fstream header

Declare file stream variables

Associate the file stream variables with the input/output sources

Use the file stream variables with `>>`, `<<`, or other input/output functions

Close the files

➤ File input/output (cont.)

- Step 1 requires that the header file `fstream` be included in the program.
- The following statement accomplishes this task:

```
#include <fstream>
```

- Step 2 requires you to declare file stream variables:

```
ifstream inData;
```

```
ofstream outData;
```


➤ File input/output (cont.)

- Step 3 requires you to associate file stream variables with the input/output sources.
- This step is called **opening the files**. The stream member function `open` is used to open files.
- The syntax for opening a file is:

`fileStreamVariable.open(sourceName);`

- Here, **`fileStreamVariable`** is a file stream variable
- and **`source Name`** is the name of the input/output file.

➤ File input/output (cont.)

- Suppose that the input data is stored in a file called `prog.dat`.
- The following statements associate `inData` with `prog.dat` and `outData` with `prog.out`.
- That is, the file `prog.dat` is opened for inputting data, and the file `prog.out` is opened for outputting data.

```
inData.open("prog.dat");
```

```
outData.open("prog.out");
```

➤ **File input/output (cont.)**

- Step 4 you use the file stream variables with `>>`, `<<`, or other input/output functions.
- The syntax for using `>>` or `<<` with file stream variables is the same as the syntax for using `cin` and `cout`.
- Instead of using `cin` and `cout`, however, you use the file stream variable names that were declared.

➤ File input/output (cont.)

```
inData >> payRate;
```

- reads the data from the file prog.dat and stores it in the variable payRate.

```
outData << "The paycheck is: $" << pay << endl;
```

- stores the output:

```
The paycheck is: $ 565.78
```

- in the file prog.out. This statement assumes that the pay was calculated as 565.78.

➤ **File input/output (cont.)**

- Once the I/O is complete, Step 5 requires closing the files.
- Closing a file means that the file stream variables are disassociated from the storage area and are freed. Once these variables are freed, they can be reused for another file I/O.

```
inData.close();
```

```
outData.close();
```

➤ File input/output (cont.)

```
#include <fstream>

//Add additional header files you use

using namespace std;

int main()
{
    //Declare file stream variables such as the following
    ifstream inData;
    ofstream outData;
    .
    .
    .

    //Open the files
    inData.open("prog.dat"); //open the input file
    outData.open("prog.out"); //open the output file

    //Code for data manipulation

    //Close files
    inData.close();
    outData.close();

    return 0;
}
```

➤ **PROGRAMMING EXAMPLE: Student Grade**

- Write a program that reads a student name followed by five test scores.
 - The program should output the student's name, the five test scores, and the average test score.
 - The data to be read is stored in a file called test.txt. The output should be stored in a file called testavg.txt.
-
- **Input:** A file containing the student's name and the five test scores. A sample input is:

Ayman Soliman 2 3 4 5 6
 - **Output:** The student's name, the five test scores, and the average of the five test scores, saved to a file.

➤ File input/output (cont.)

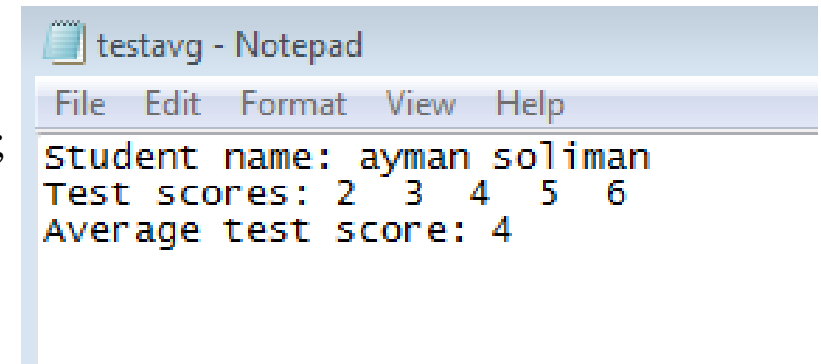
```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main ()
{
    ifstream inFile;
    ofstream outFile;
    double test1, test2, test3, test4, test5;
    double average;
    string firstName;
    string lastName;

    inFile.open("test.txt");
    outFile.open("testavg.txt");
```

```
    cout << "Processing data" << endl;
    inFile >> firstName >> lastName;
    outFile << "Student name: " << firstName << " " << lastName
    << endl;
```

```
    inFile >> test1 >> test2 >> test3 >> test4 >> test5;
    outFile << "Test scores: " << test1 << " " << test2 << " " <<
        test3 << " " << test4 << " " << test5 << endl;
    average = (test1 + test2 + test3 + test4 + test5) / 5.0;
    outFile << "Average test score: " << average << endl;
```

```
    inFile.close();
    outFile.close();
    return 0;
}
```



```
testavg - Notepad
File Edit Format View Help
Student name: ayman soliman
Test scores: 2 3 4 5 6
Average test score: 4
```


Thank
you

