

ADVANCED PROGRAMMING (BETC 1353)

WEEK 7: FILE PROCESSING (PART 1)

AIMAN ZAKWAN BIN JIDIN

aimanzakwan@utem.edu.my

NORFADZLIA BINTI MOHD YUSOF

norfadzlia@utem.edu.my

Learning Outcomes:

At the end of this session, you should be able:

- To explain the data hierarchy, files and streams
- To create a sequential-access file

A File

- is collection of records
- Is stored in a folder physically
- A File extension is usually included to indicate a kind of file
 - .txt -> a text file
 - .jpg -> an image file
 - .cpp -> a CPP source code

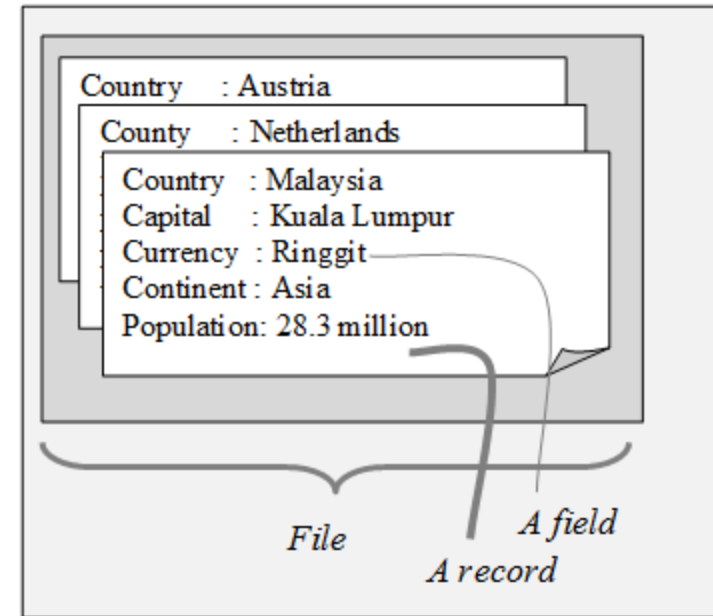
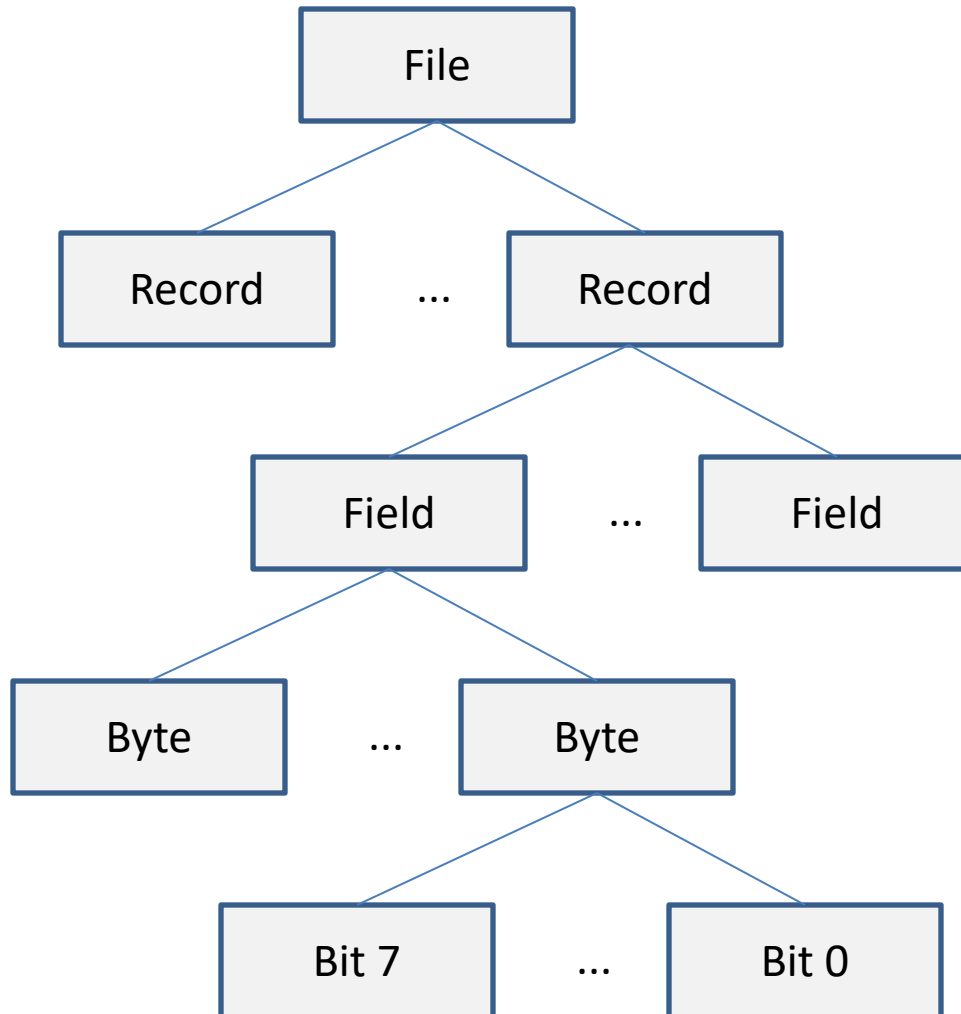
File Classifications

- Files can be classified based on how to access data in its memory as:
 - **Sequential file:** All records are stored sequentially as there are entered. It is suitable if all of the contents are processed
 - **Random access file:** A record can be accessed directly by using its index. Indices for all records must be maintained.
 - **Direct access file:** A record can be accessed based on its relative position to the first record. Each record must be have same length.

File Classifications

- A File can be classified as a text file or a binary file
 - **A text file:** Data are stored in ASCII character code. Its contents can be viewed by using **type** command in DOS mode (Command prompt)
 - **A binary file:** Data are stored same as its format in the computer memory spaces.
- Suppose, you will save data: 4567801.92. In a text file, that data need 10 bytes (10 characters). In a binary file, that data consume 4 bytes (float type).

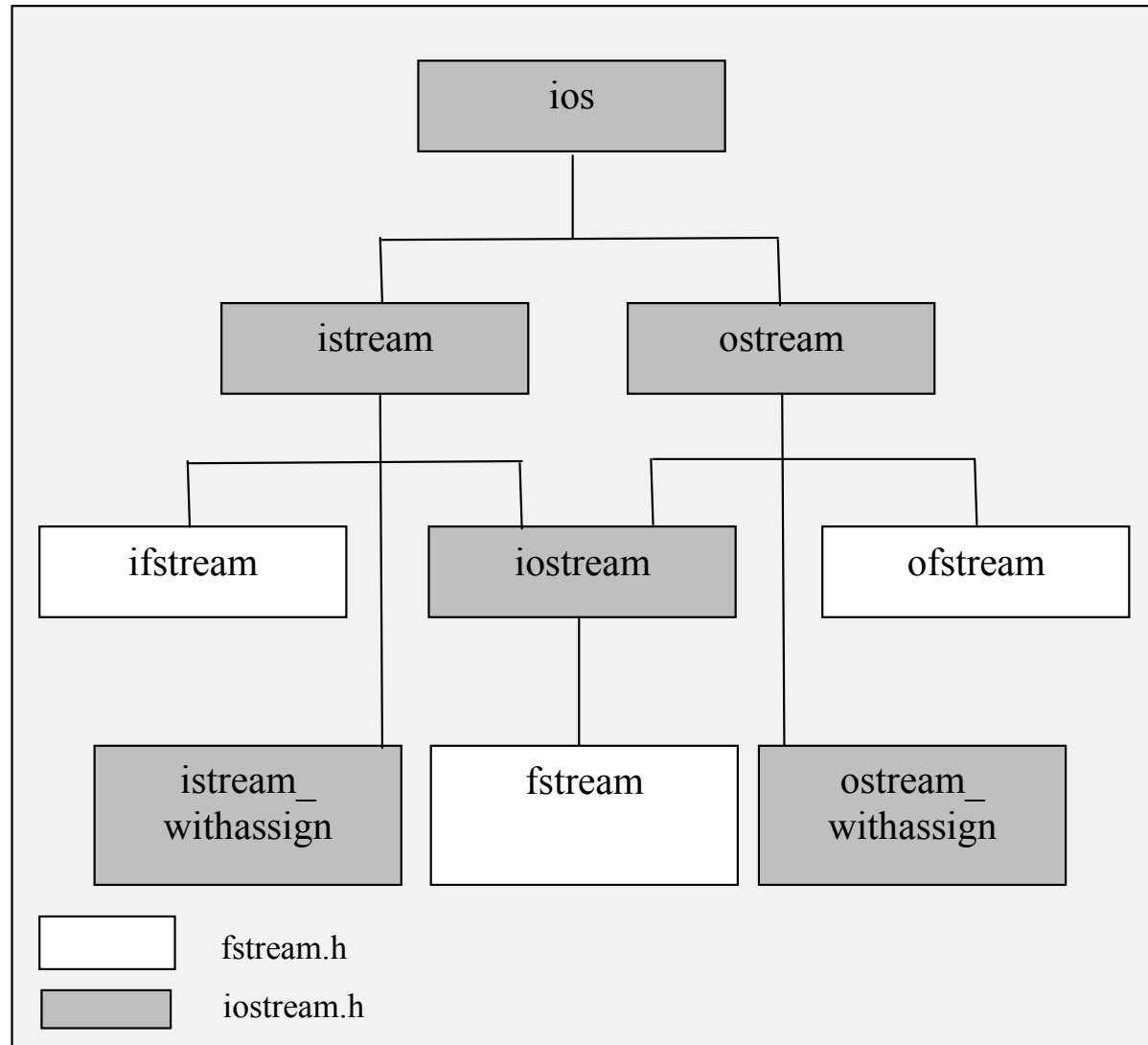
Data Hierarchy in A File



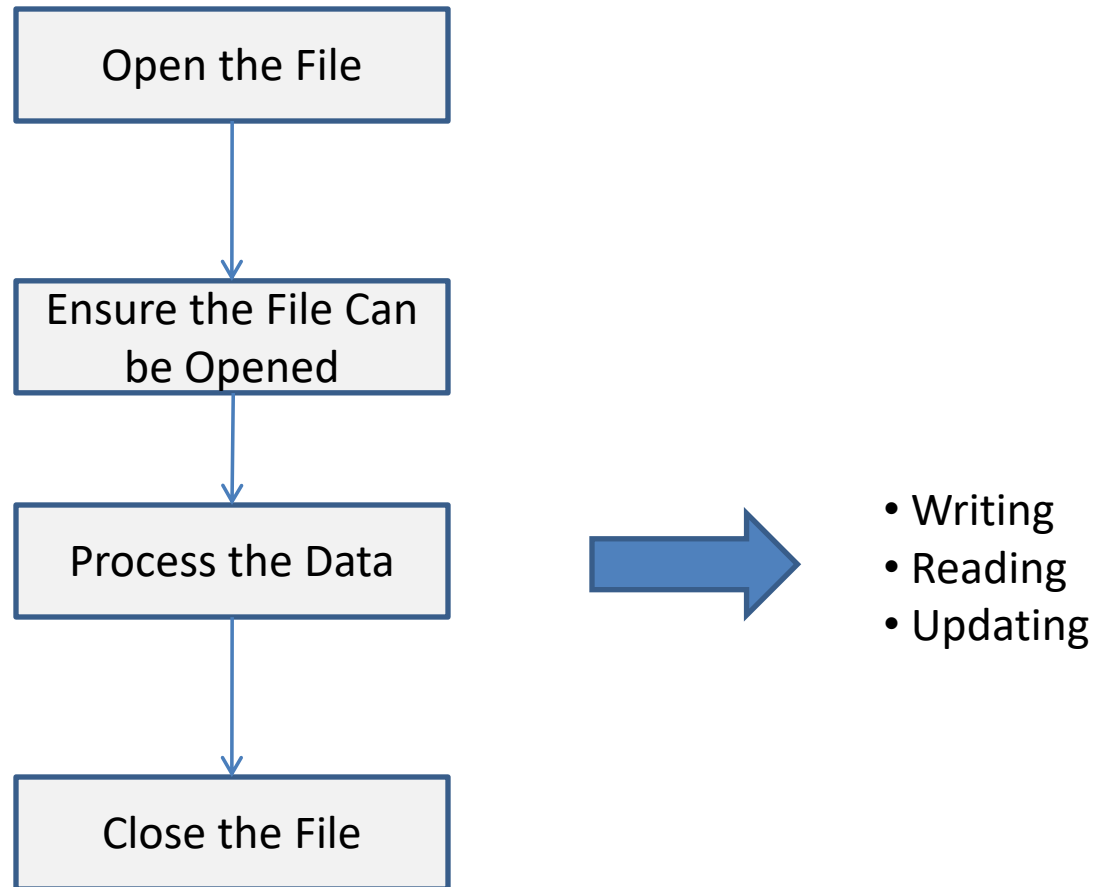
Stream

- *Stream* is a general term to indicate data flow from one place to another one.
- For example:
 - **cin** is an object that handles data flow from a standard input (i.e. keyboard) into a variable
 - **cout** is an object that handles data flow from any expressions into standard output (i.e. consol)
- File processing also uses the stream mechanism like on cin or cout. Therefore, operators such as << and >> that are used in cin or cout can be used in file processing

Class Hierarchy for Stream Operations



Procedure in Accessing a File



How to Store Data in a Sequential-Access File (Part 1)

- First, create object by using **ofstream** class (its prototype is in `fstream.h`). For example:

```
ofstream country_file;
```

Previously, you must include: `#include <fstream>`

- To create and open the file, use the **open()** member function of the object. For example:
`country_file.open('country.txt');`

How to Store Data in a Sequential-Access File (Part 2)

- To ensure that file can be opened, check by calling the **fail()** member function of the object. This function returns **true** if the file can not be created/opened. For example:

```
if (country_file.fail())
{
    cout << "File country.txt can't be created" << endl;
    exit(1);
}
```

- The alternative way is using the **is_open()** member function. This function returns **true** if the file can be opened. For example:

```
if (!country_file.is_open())
{
    cout << "File country.txt can't be created" << endl;
    exit(1);
}
```

How to Store Data in a Sequential-Access File (Part 3)

- Suppose your object is **country_file**, the following command will write data to the file:

```
country_file << setw(20) << "Austria"  
              << setw(20) << "Vienna"  
              << endl;
```

- To close the file, use the **close()** member function of the object. For example:

```
country_file.close();
```

Try Yourself!

savecountry.cpp

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <iomanip>

using namespace std;

int main()
{
    ofstream country_file; // object declaration

    // Create and open country.txt file
    country_file.open("country.txt");
    if (country_file.fail())
    {
        cout << "File country.txt can't be created" << endl;
        exit(1);
    }

    // Save data
    country_file << setiosflags(ios::left);
    country_file << setw(20) << "Austria"
                << setw(20) << "Vienna" << endl;
    country_file << setw(20) << "Netherlands"
                << setw(20) << "Amsterdam" << endl;
    country_file << setw(20) << "Indonesia"
                << setw(20) << "Jakarta" << endl;
    country_file << resetiosflags(ios::left);
```

*Please continue
to the next
page.*



savecountry.cpp

```
// Close the file
country_file.close();

cout << "Data have been stored in the country.txt file "
      << endl;
cout << "Please check it in the same folder with this program"
      << endl;
}
```

Yes, here is the
end of program.
Lets try now to
compile and run it.



How to Check The File



The screenshot shows a Windows Command Prompt window with an orange title bar. The text inside the window is as follows:

```
C:\>
D:\Teaching\program>dir country.txt
Volume in drive D is New Volume
Volume Serial Number is E036-3716

Directory of D:\Teaching\program

11/14/2013  08:57 AM                126 country.txt
               1 File(s)                126 bytes
               0 Dir(s) 106,249,302,016 bytes free

D:\Teaching\program>type country.txt
Austria      Vienna
Netherlands  Amsterdam
Indonesia    Jakarta

D:\Teaching\program>
```

How To Create a Binary File

- Add the second argument of the **open()** with **ios::binary**
- **The ios::binary** is called as a **file mode flag**, a flag that determines the operation will be done in the file.
- Write data using the **write()** member function

savebin.cpp

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

using namespace std;

int main()
{
    ofstream output_file;

    // Create and open the file
    output_file.open("number.bin", ios::binary);
    if (output_file.fail())
    {
        cout << "The number.bin file can't be created"
              << endl;
        exit(1);
    }
}
```

*Please continue
to the next
page.*



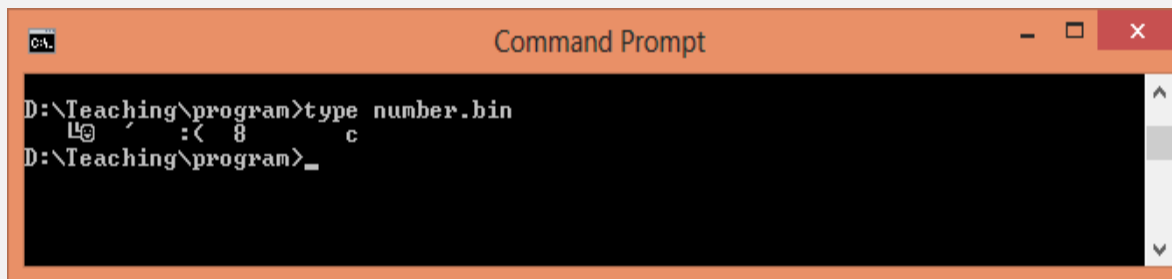
savebin.cpp

```
// Store the data
int data[] = {7, 456, 239, 10298, 56, -1, 99};
for (int i = 0; i < sizeof(data) / sizeof(int); i++)
    output_file.write((char *) &data[i], sizeof(int));

// Close the file
output_file.close();

cout << "Data have been stored in the number.bin file"
      << endl;
cout << "Please check it in the same folder with this program"
      << endl;
}
```

You can see the result by using the type command. And, of course, you can't read a binary file with your eyes!



```
Command Prompt
D:\Teaching\program>type number.bin
U@ ' :< 8 c
D:\Teaching\program>_
```



Try Yourself!

savetext.cpp

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

using namespace std;

int main()
{
    ofstream output_file;

    // Create and open the file
    output_file.open("number.txt");
    if (output_file.fail())
    {
        cout << "The number.txt file can't be created"
              << endl;
        exit(1);
    }
}
```

*Please continue
to the next
page.*



Try Yourself!

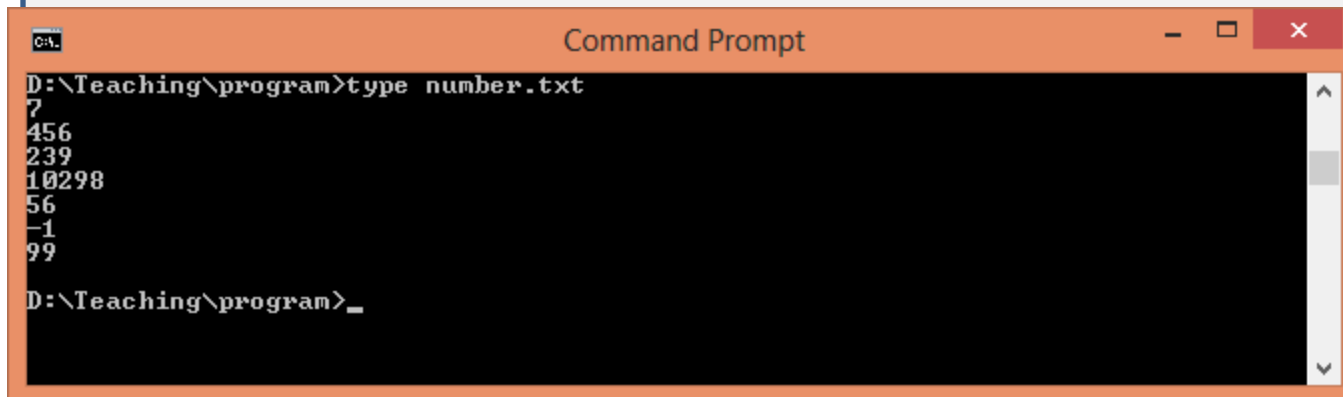
savetext.cpp

```
// Store the data
int data[] = {7, 456, 239, 10298, 56, -1, 99};
for (int i = 0; i < sizeof(data) / sizeof(int); i++)
    output_file << data[i] << endl;

// Close the file
output_file.close();

cout << "Data have been stored in the number.txt file"
    << endl;
cout << "Please check it in the same folder with this program"
    << endl;
}
```

*Notice the result! Yes, you
can read the data if you
store the data in a text file!*



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command prompt is at the directory "D:\Teaching\program". The user has entered the command "type number.txt", and the output displays the contents of the file: 7, 456, 239, 10298, 56, -1, and 99, each on a new line. The prompt is currently at "D:\Teaching\program>_".

```
Command Prompt
D:\Teaching\program>type number.txt
7
456
239
10298
56
-1
99
D:\Teaching\program>_
```



How to Add Data to the Existing File

- Add the second argument of the **open()** with **ios::app**
- For example:

```
country_file.open("country.txt", ios::app);
```

The existing data will not be deleted. The new output will be appended into the last record.



addcountry.cpp

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <iomanip>

using namespace std;

int main()
{
    ofstream country_file; // object declaration

    // open country.txt file for adding data
    country_file.open("country.txt", ios::app);
    if (country_file.fail())
    {
        cout << "File country.txt can't be created"
              << endl;
        exit(1);
    }
}
```

*Please
continue to
next page.*



Try Yourself!

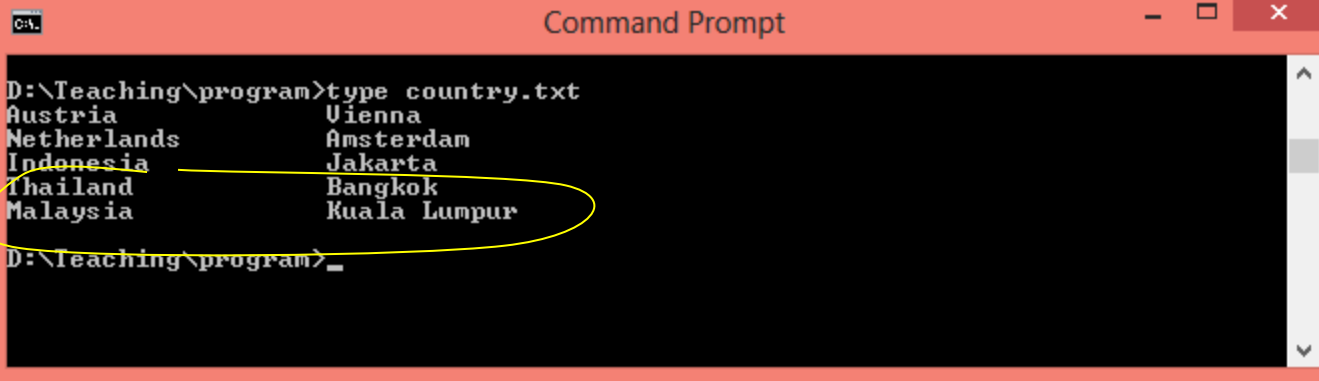
addcountry.cpp

```
// Save data
country_file << setiosflags(ios::left);
country_file << setw(20) << "Thailand"
               << setw(20) << "Bangkok" << endl;
country_file << setw(20) << "Malaysia"
               << setw(20) << "Kuala Lumpur" << endl;
country_file << resetiosflags(ios::left);

// Close the file
country_file.close();

cout << "Data have been added into the country.txt file "
      << endl;
}
```

You can see that Thailand and Malaysia have been appended into the country.txt file.



```
CA. Command Prompt
D:\Teaching\program>type country.txt
Austria      Vienna
Netherlands  Amsterdam
Indonesia    Jakarta
Thailand      Bangkok
Malaysia     Kuala Lumpur
D:\Teaching\program>
```



Useful File Flag Modes

- **ios::app** : Append mode. If file exists, the contents are preserved. The new output will be appended to the last records
- **ios::binary**: Binary mode
- **ios::in**: Read mode. File is intended for reading
- **ios::out**: Write mode. File is intended for writing

Member Functions for Detecting Errors

- **good()**: The function returns **true** if the last operation on the file is successful
- **eof()**: The function returns **true** if the last operation that read or read data make the file pointer reaches the end of file. Useful for reading data
- **fail()**: The function returns **true** if the last operation on the file is not successful
- **bad()**: The function returns **true** if the last operation is not valid

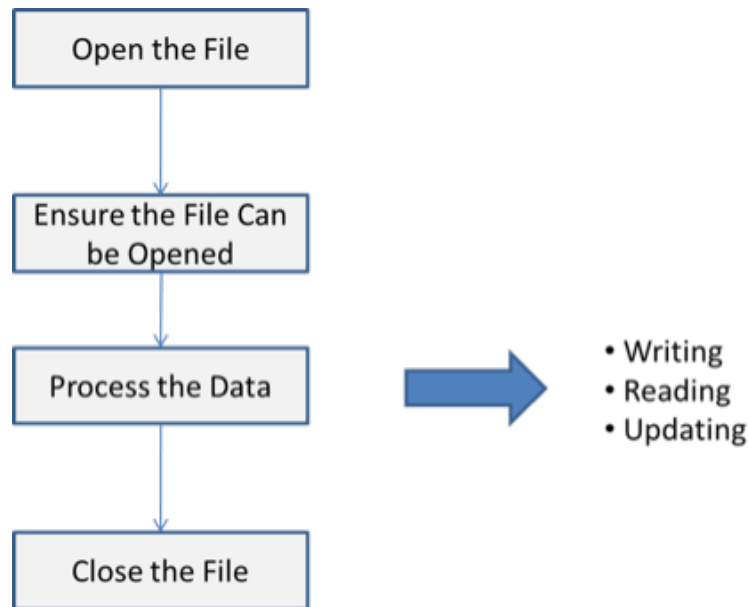
Self-Review Questions

1. Write the 4 fundamental steps in the file processing in C++.
2. Write a C++ program which writes the text “My favorite food is chicken burger” to a text file named `mytext.txt`.

Self-Review Questions

Answers:

1. 4 fundamental steps in file processing:



Self-Review Questions

Answers:

2.

```
#include <iostream>
#include <fstream>           //file processing functions
using namespace std;

int main(){
    ofstream outfile;
    outfile.open("mytext.txt");    //open the file
    if (outfile.fail()){          //ensure the file is successfully opened
        cout << "File opening Error!\n";
        exit(1);
    }
    outfile << "My favorite food is chicken burger"; //write to file
    outfile.close();              //close the file
    return 0;
}
```