Lab 14
File Handling

What is file handling in C++?

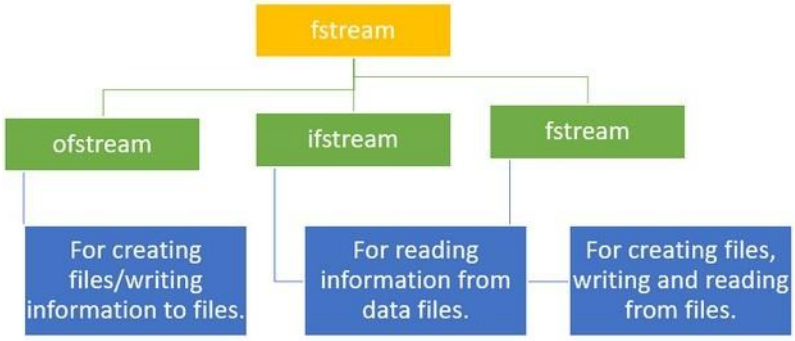
Files store data permanently in a storage device. With file handling, the output from a program can be stored in a file. Various operations can be performed on the data while in the file. A stream is an abstraction of a device where input/output operations are performed. You can represent a stream as either a destination or a source of characters of indefinite length. This will be determined by their usage. C++ provides you with a library that comes with methods for file handling.

The fstream Library

The fstream library provides C++ programmers with three classes for working with files. These classes include:

- **ofstream**- This class represents an output stream. It's used for creating files and writing information to files.
- **ifstream**- This class represents an input stream. It's used for reading information from data files.
- **fstream**- This class generally represents a file stream. It comes with ofstream/ifstream capabilities. This means it's capable of creating files, writing to files, reading from data files.

The following image makes it simple to understand:



To use the above classes of the fstream library, you must include it in your program as a header file.

How to Open Files

Before performing any operation on a file, you must first open it. If you need to write to the file, open it using fstream or ofstream objects. If you only need to read from the file, open it using the ifstream object. The three objects, that is, fstream, ofstream, and ifstream, have the open() function defined in them. The function takes this syntax:

```
open (file_name, mode);
```

- The file_name parameter denotes the name of the file to open.
- The mode parameter is optional. It can take any of the following values:

Value	Description
ios:: app	The Append mode. The output sent to the file is appended to it.
ios::ate	It opens the file for the output then moves the read and write control to file's end.
ios::in	It opens the file for a read.
ios::out	It opens the file for a write.
ios::trunk	If a file exists, the file elements should be truncated prior to its opening.

How to Close Files

Once a C++ program terminates, it automatically

- flushes the streams
- releases the allocated memory
- closes opened files.

However, as a programmer, you should learn to close open files before the program terminates. The fstream, ofstream, and ifstream objects have the close() function for closing files. The function takes this syntax:

```
void close();
```

How to Write to Files

You can write to file right from your C++ program. You use stream insertion operator (<<) for this. The text to be written to the file should be enclosed within double-quotes.

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
        fstream my_file;
        my_file.open("my_file.txt", ios::out);
        if (!my_file) {
                cout << "File not created!";
        }
        else {
                cout << "File created successfully!";
                my_file << "Guru99";
                my_file.close();
        }
        return 0;
}
```

How to Read from Files

You can read information from files into your C++ program using stream extraction operator (>>). You use the operator in the same way you use it to read user input from the keyboard. However, instead of using the cin object, you use the ifstream/ fstream object.

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
        fstream my_file;
        my_file.open("my_file.txt", ios::in);
        if (!my_file) {
                cout << "No such file";
        }
        else {
                char ch;

                while (1) {
                        my_file >> ch;
                        if (my_file.eof())
                                break;

                        cout << ch;
                }

        }
        my_file.close();
        return 0;
}
```

Special operations in a File

There are few important functions to be used with file streams like:

- tellp()- It tells the current position of the put pointer.
  Syntax: filepointer.tellp()
- tellg()- It tells the current position of the get pointer.
  Syntax: filepointer.tellg()
- seekp()- It moves the put pointer to mentioned location.
  Syntax: filepointer.seekp(no of bytes,reference mode)
- seekg()- It moves get pointer(input) to a specified location.
  Syntax: filepointer.seekg((no of bytes,reference point)
- put()- It writes a single character to file.
- get()- It reads a single character from file.

Tasks

Problem No: 1

Q1: Write a program to implement Employee Directory. The main purpose of the class is to get the data, store it into a file, perform some operations and display data.
For the purpose mentioned above, you should write a template class **Directory** for storing the data empID(**template**), empFirstName(**string**), empLastName(**string**), empContactNumber(**int**) and empAddress(**string**) of each Employee in a file **EmployeeDirectory.txt**.

1. Write a function **Add** to write the above mentioned contact details of the employee into EmployeeDirectory.txt. The record of each employee should be written in one line having space among the attributes. Consider the file below as example for EmployeeDirectory.txt

9743 Ali Ahmad 03005543287 H#34, sector H11, Islamabad 1234 Ayesha Ahmad 03215573987 H#45, sector G11,

2. Write a template function **UpdateAddress** that receives two parameters **empID** of type template and **AddressToBeUpdated** of type string. This function will update the address of the given employee in the file EmployeeDirectory.txt.
3. Write a template function **SearchById** that receives empID of type template as parameter. Your function should be capable of searching the employee record based on the provided employee id. The return type of the function should be template, which should be able to either return true/false or the string of the employee record.
4. Write a function **Delete** of type void which accepts a parameter empContactNumber of type string and performs the delete operation according to the user's request. This function should be able to update the file after deletion of employee record.
5. Also write a void function **printDetails** which must read the file completely and print the record stored in EmployeeDirectory.txt

Problem No: 2

Q2: Create a program for writing records of 10 students in a text fie Students.txt. Store roll no, first name, last name, department and section of student.
Output should be like this in the file.

Student.txt

123 Asad Ali CS A.
321 Abbas Khan EE C.
342 Bilal Haider FSM A.

Problem No: 3

Q3: To automate bill of a restaurant you have to take input from file. The file contains name of item, price, Ingredients then you have to calculate the bill and write down the bill in the file.
FunctiongetData: This function loads the data into the array menuListfrom a file.
FunctionshowMenu: This function shows the different items offered by the restaurant and tells the user how to select the items.
FunctionprintCheck: This function calculates and prints the check.
Use Character Array for Ingredients.

Input.txt (menu.txt)

Name of item	*price*	*ingredients*
Plain Egg	10	A,B,C
Mutton	30	A,C,D
Tea	5	G,J,K

Output.txt

Name of item	*Quantity*	*Price*	*Ingredients*
Plain Egg	2	20	A,B,C
Mutton	1	30	A,C,D
Tea	5	25	G,J,K
Total		75	