Section #1 – File Organization

Sequential Files Implementation

Note: At the beginning of the section, a quick review of (Member Function, Class, Object, Constructor).

1) Sequential Files:

1. Records can only be accessed sequentially, one after another.
2. Use loops to read and process records one by one.
3. Used in applications that need to access all records from beginning to end.

2) Write a program used to list contents of file using C++ stream.

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

void main() {
	char ch;
	fstream file;
	char filename[20];
	cout << "Enter the name of the file: " << flush;
	cin >> filename;
	file.open(filename, ios::in);
	file.unsetf(ios::skipws);
	while (1)
	{
		file >> ch;
		if (file.fail())
			break;
		cout << ch;
	}
	file.close();
	_getch();
}
```

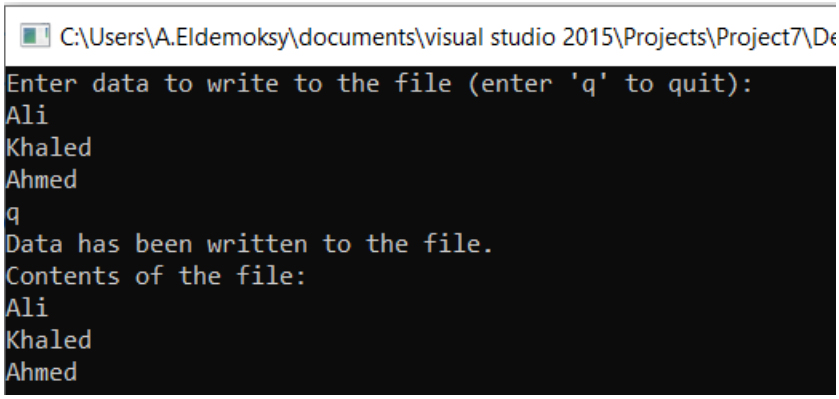
```
C:\Users\A.Eldemoksy\documents\visual studio 20
Enter the name of the file: D:\data.txt
ahmed
ali
khaled
```

Write a program that contains two functions: the first is used to write data into a sequential file, and the second is used to read data from a file using a C++ stream.

```
#include <iostream>
#include <fstream>
#include <string>
#include <conio.h>
using namespace std;
void writeToFile(const string& filepath)
{
	ofstream outputFile(filepath); // Open file for writing
	if (!outputFile.is_open()) {
		cerr << "Error opening file " << filepath << " for writing." << endl;
		return;
	}
	string data;
	cout << "Enter data to write to the file (enter 'q' to quit):" << endl;
	while (true) {
		getline(cin, data);
		if (data == "q")
			break;
		outputFile << data << endl; // Write data to file
	}
	outputFile.close(); // Close the file
	cout << "Data has been written to the file." << endl;
}
void readFromFile(const string& filepath)
{
	ifstream inputFile(filepath); // Open file for reading
	if (!inputFile.is_open()) {
		cerr << "Error opening file " << filepath << " for reading." << endl;
		return;
	}
	string line;
	cout << "Contents of the file:" << endl;
	while (getline(inputFile, line)) {
		cout << line << endl; // Print each line of the file
	}
	inputFile.close(); // Close the file
}
```

```
void main()
{
	string path = "D:/data.txt"; // Specify the path to the file
	writeToFile(path); // Write data to file
	readFromFile(path); // Read data from file

	_getch();
}
```



Write the necessary code for creating a class called Person that is used to save person data, then create a Person object, get its data from the user, and finally save it into a sequential text file using a stream at a specific location.

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

using namespace std;

class Person
{
public:
	string lastName;
	string firstName;
	string address;
	string city;
	string state;
	string zipCode;
	Person(); // Constructor declaration
	void getDataFromUser(); // Method to get data from user
	void saveToFile(const string& filePath); // Method to save data to file
};

// Constructor Definition
Person::Person()
{
	lastName = "0";
	firstName = "0";
	address = "0";
	city = "0";
	state = "0";
	zipCode = "0";
}
```

```
// Method to get data from user
void Person::getDataFromUser()
{
	cout << "Enter Last Name: ";
	getline(cin, lastName);

	cout << "Enter First Name: ";
	getline(cin, firstName);

	cout << "Enter Address: ";
	getline(cin, address);

	cout << "Enter City: ";
	getline(cin, city);

	cout << "Enter State: ";
	getline(cin, state);

	cout << "Enter Zip Code: ";
	getline(cin, zipCode);
}

// Method to save data to file
void Person::saveToFile(const string& filePath)
{
	ofstream outFile(filePath); // Open file
	// Check if the file is opened successfully
	if (!outFile.is_open())
	{
		cerr << "Error: Unable to open the file.\n";
		return;
	}
	// Write data to file
	outFile << "Last Name: " << lastName << endl;
	outFile << "First Name: " << firstName << endl;
	outFile << "Address: " << address << endl;
	outFile << "City: " << city << endl;
	outFile << "State: " << state << endl;
	outFile << "Zip Code: " << zipCode << endl;
	outFile.close();
}
```

```
int main()
{
	// Create a Person object
	Person person1;

	// Get data from the user
	person1.getDataFromUser();

	// Save the Person object's data to a text file at specific location
	person1.saveToFile("D:\\data.txt");

	cout << "Person data saved to file.\n";

	return 0;
}
```