**LAB 10**

**QUESTION 1:**

**This task involves developing a student information system for a school. The program will begin by defining a Student class, which includes fields to store a student's ID (int), name (string), and GPA (float). The program will then prompt the user to enter data for 5 students. These student records will be written to a file named "students.txt" using ofstream in ios::out mode. Subsequently, the program will allow the user to append the record of one additional student to the same file, utilizing ios::app mode to add the new record without overwriting the existing ones. Finally, the program will read and display all student records from the file using ifstream.**

**PROGRAM**:

#include<iostream>

#include<fstream>

using namespace std;

class student {

private:

    string name;

    int id;

    float gpa;

public:

    void writeDataToFile(ofstream &*studentFile*) {

        cout << "Enter name: ";

        cin >> name;

        cout << "Enter ID: ";

        cin >> id;

        cout << "Enter GPA: ";

        cin >> gpa;

        studentFile << "Name: " << name << ", ID: " << id << ", GPA: " << gpa << endl;

    }

    void readFromFile(ifstream &*displayFile*) {

        string line;

        while (getline(displayFile, line)) {

            cout << line << endl;

        }

    }

};

int main() {

    student s[5];

    ofstream studentFile("student.txt", ios::out);

    if (!studentFile) {

        cout << "Error opening file for writing!" << endl;

        return 1;

    }

    for (int i = 0; i < 5; i++) {

        s[i].writeDataToFile(studentFile);

    }

    studentFile.close();

    ofstream studentFileapp("student.txt", ios::app);

    if (!studentFileapp) {

        cout << "Error opening file for appending!" << endl;

        return 1;

    }

    student appendstudent;

    appendstudent.writeDataToFile(studentFileapp);

    studentFileapp.close();

    ifstream DisplayFile("student.txt");

    if (!DisplayFile) {

        cout << "Error opening file for reading!" << endl;

        return 1;

    }

    student displaystudent;

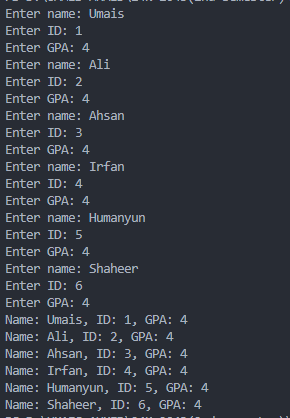
    displaystudent.readFromFile(DisplayFile);

    DisplayFile.close();

    return 0;

}

**RESULT:**

****

**QUESTION#2**

**This task focuses on creating a profile management feature for a social media application. The program will prompt the user to enter their profile details, including their name (string), email (string), years of experience (int), and a brief summary (string). This data will be written to a file named "resume.txt". To ensure that only the latest profile information is stored, ofstream will be used with ios::trunc to clear any previous data in the file before writing the new information.After writing the data, the program will read and display the file's content to confirm that the file contains only the most recent entry.**

**PROGRAM**:

#include<iostream>

#include <fstream>

using namespace std;

class user{

    private:

    string name;

    string email;

    int yearOfexperience;

    string summary;

    public:

    void WriteToFile(ofstream &*resumeFile*){

      cout<<"Enter name :";

      cin>>name;

      cout<<"Enter email :";

      cin>>email;

      cout<<"Enter year of experience :";

      cin>>yearOfexperience;

      cout<<"Enter summary :";

      cin>>summary;

      resumeFile<<"name :"<<name<<", email :"<<email<<", year of experience :"<<yearOfexperience<<", summary :"<<summary<<endl;

    }

   void readFromFile(istream &*displayFile*){

    string line;

    while(getline(displayFile, line)){

        cout << line << endl;

    }

   }

};

int main(){

    user u;

    ofstream resumeFile("resume.txt",ios::trunc);

    if(!resumeFile){

        cout<<"Error to opening file"<<endl;

        return 1;

    }

    u.WriteToFile(resumeFile);

    user u2;

    u2.WriteToFile(resumeFile);

    resumeFile.close();

    ifstream displayFile("resume.txt");

    if(!displayFile){

        cout<<"Error to opening file"<<endl;

        return 1;

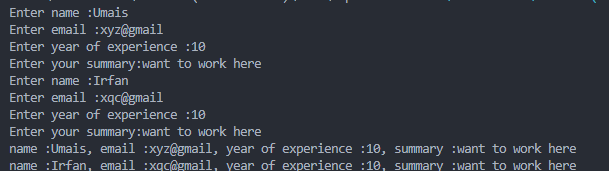
    }

    user u3;

    u3.readFromFile(displayFile);

    displayFile.close();

}



**QUESTION#3**

**This task involves building a tool that assembles a final document by merging the contents of two separate chapter files. The program will start by reading the contents of two existing files, "chapter1.txt" and "chapter2.txt", using ifstream. It is assumed that these chapter files are already populated with text. The program will then open a file named "book.txt" using ofstream in ios::app mode. The contents read from the chapter files will be appended sequentially to "book.txt". To maintain clear separation between the chapters' content, a newline character will be inserted between the appended content of each chapter. Finally, the program will open and display the contents of "book.txt" to verify that the chapter contents have been merged correctly.**

#include<iostream>

#include <fstream>

using namespace std;

class book{

    private:

    ofstream bookFile;

    ifstream displayFile;

  public:

  book():bookFile("book.txt",ios::app),displayFile("book.txt"){

  }

  void WriteTotheFile(ifstream &*chapter1File*, ifstream &*chapter2File*){

    string line;

    bookFile << "Chapter 1" << endl;

    while(getline(chapter1File, line)){

        bookFile << line << endl;

    }

    bookFile << endl;

    bookFile << "Chapter 2" << endl;

    while(getline(chapter2File, line)){

        bookFile << line << endl;

    }

    bookFile.close();

}

void readTotheFile() {

    string line;

    while (getline(displayFile, line)) {

        if (line.empty()) {

            cout << "\n";

        } else {

            cout << line << endl;

        }

    }

    displayFile.close();

}

};

int main(){

    ifstream chapter1("chapter 1.txt");

    ifstream chapter2("chapter 2.txt");

    if (!chapter1 || !chapter2) {

        cerr << "Error: One or both chapter files could not be opened." << endl;

        return 1;

    }

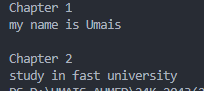
    book b;

    b.WriteTotheFile(chapter1,chapter2);

   book b2;

   b2.readTotheFile();

}



**QUESTION#4**

#include <iostream>

#include <fstream>

using namespace std;

class TextEditor {

private:

    fstream file;

public:

    TextEditor(const string& *filename*) {

        file.open(*filename*, ios::in | ios::out);

        if (!file) {

            cerr << "Error opening file!" << endl;

            exit(1);

        }

    }

    void writeInitialContent() {

        ofstream outFile("info.txt", ios::out);

        outFile << "C++ is a powerful programming language.";

        outFile.close();

    }

    void displayInitialPositions() {

        cout << "Initial get pointer: " << file.tellg() << endl;

        cout << "Initial put pointer: " << file.tellp() << endl;

    }

    void modifyContent() {

        string word;

        file.seekg(9);

        file.read(&word[0], 8);

        cout << "Word read at position 9: " << word << endl;

        file.seekp(9);

        file << "dynamic";

    }

    void displayUpdatedPositions() {

        cout << "Updated get pointer: " << file.tellg() << endl;

        cout << "Updated put pointer: " << file.tellp() << endl;

    }

    void displayModifiedContent() {

        file.close();

        ifstream inFile("info.txt");

        string line;

        getline(inFile, line);

        cout << "Modified content: " << line << endl;

        inFile.close();

    }

};

int main() {

    TextEditor editor("info.txt");

    editor.writeInitialContent();

    editor.displayInitialPositions();

    editor.modifyContent();

    editor.displayUpdatedPositions();

    editor.displayModifiedContent();

    return 0;

}

A screenshot of a computer program

AI-generated content may be incorrect.