FEU Institute of Technology
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

CCS0007
Computer Programming 2 for IT

EXERCISE

6

File Handling

CAPILI, MARK ANGELO	Joie Ann Maghanoy
18/02/2020	18/02/2020

I. OBJECTIVES

At the end of this exercise, students must be able to:

- Create a program that open a given text file, process, and write the output to another file.
- Create a record management system that stores basic information on a given text file and load the data when opened.

II. BACKGROUND INFORMATION

To perform file processing in C++, header files <iostream> and <fstream> must be included in your C++ source file.

This requires another standard C++ library called fstream, which defines three new data types:

Data Type	Description
ofstream	This data type represents the output file stream and is used to create files and to write information to files.
ifstream	This data type represents the input file stream and is used to read information from files.
fstream	This data type represents the file stream generally, and has the capabilities of both ofstream and ifstream which means it can create files, write information to files, and read information from files.

III. EXPERIMENTAL PROCEDURE

INSTRUCTIONS:

Copy your source codes to be pasted in this document as well as a screen shot of your running output.

Upload your document using the link provided in your Canvas.

ACTIVITY 6.1: Player's record with file operations

Redo your program in Laboratory Activity 5, this time you have to save the records to a text file and will be retrieved for display. Your program must be able to keep records and compute for the scores of 5 players. The information of each player contains: Nickname, Age and two best played scores.

The program will prompt the user to choose the operation of records from a menu as shown below:

```
==============================================
                    MENU
```

==========

1. Add record
2. View players records
3. Compute for the average
4. Show the player(s) who gets the max average.
5. Show the player(s) who gets the min average.
6. Open the file.
7. Close the File
8. Exit

```
#include <iostream>
#include <cstring>
#include <fstream>
using namespace std;

ifstream inFile;
ofstream outFile;


int choice;
// structure for storing all the players data
struct Person
{
    char name[50];
    int age;
    int score1;
    int score2;
    float ave;
};

//for IU
void menu(){
     cout << "**********************************\n";
     cout << "            MENU            \n";
     cout << "**********************************\n";
     cout << "1. Add Record\n";
     cout << "2. View Players Records\n";
     cout << "3. Compute For The Average\n";
     cout << "4. Show the player(s) who get the max average\n";
     cout << "5. Show the player(s) who get the min average\n";
     cout << "6. Open the File\n";
```

```
**********************************
               MENU
**********************************
1. Add Record
2. View Players Records
3. Compute For The Average
4. Show the player(s) who get the max average
5. Show the player(s) who get the min average
6. Open the File
7. Close the File
8. Exit
Enter the # of choice: 1
PLAYER 1 RECORDS
Enter Name:
```

```
Enter the # of choice: 2
PLAYER 1 RECORDS:
NICKNAME: Mark1
AGE: 1
SCORE1: 1
SCORE2: 11

PLAYER 2 RECORDS:
NICKNAME: Mark2
AGE: 2
SCORE1: 22
SCORE2: 22

PLAYER 3 RECORDS:
NICKNAME: Mark3
AGE: 3
SCORE1: 33
SCORE2: 33

PLAYER 4 RECORDS:
NICKNAME: Mark4
AGE: 4
SCORE1: 44
SCORE2: 44

PLAYER 5 RECORDS:
NICKNAME: Mark5
AGE: 55
SCORE1: 55
SCORE2: 55

Press any key to continue . . .
```

```
    cout << "7. Close the File\n";
    cout << "8. Exit\n";
    cout << "Enter the # of choice: ";
    cin >> choice;

}



// ALGORITHM FOR ALL THE REQUIREMENTS in the MENU

int main() {
    int temp1,temp2;
    int aveT = 0;
    menu();
    Person p[5];
    // setting all the members to default values
    for(int i = 0 ; i < 5 ; i++) {
        p[i].age = 0;
        p[i].score1 = 0;
        p[i].score2 = 0;
    }
    while(true){
        switch(choice) {
            //FOR STORING THE DATA INSIDE tHE STRUCT
            case 1:
                for(int i = 0 ; i < 5 ; i++) {
                    cout << "PLAYER " << i + 1 << " RECORDS\n";
                    cout << "Enter Name: ";
                    cin >> p[i].name;
                    cout << "Enter Age: ";
                    cin >> p[i].age;
                    cout << "Enter 1st score: ";
                    cin >> p[i].score1;
                    cout << "Enter 2nd score: ";
                    cin >> p[i].score2;
                    system ("cls");
                }
                // FILE HANDLING
                outFile.open("data.txt");
                for(int i = 0 ; i < 5 ; i++) {
```

```
Enter the # of choice: 3
PLAYER Mark1 SCORE AVERAGE:
6

PLAYER Mark2 SCORE AVERAGE:
22

PLAYER Mark3 SCORE AVERAGE:
33

PLAYER Mark4 SCORE AVERAGE:
44

PLAYER Mark5 SCORE AVERAGE:
55

TOTAL 5 PLAYERS AVERAGE: 32
Press any key to continue . . .
```

```
Enter the # of choice: 4
THE HIGHEST AVERAGE AMONG ALL PLAYER IS : Mark5
Press any key to continue . . .
```

```
Enter the # of choice: 5
THE LOWER AVERAGE AMONG ALL PLAYER IS : Mark1
Press any key to continue . . .
```

```
Enter the # of choice: 6
outfile.open command was executed
```

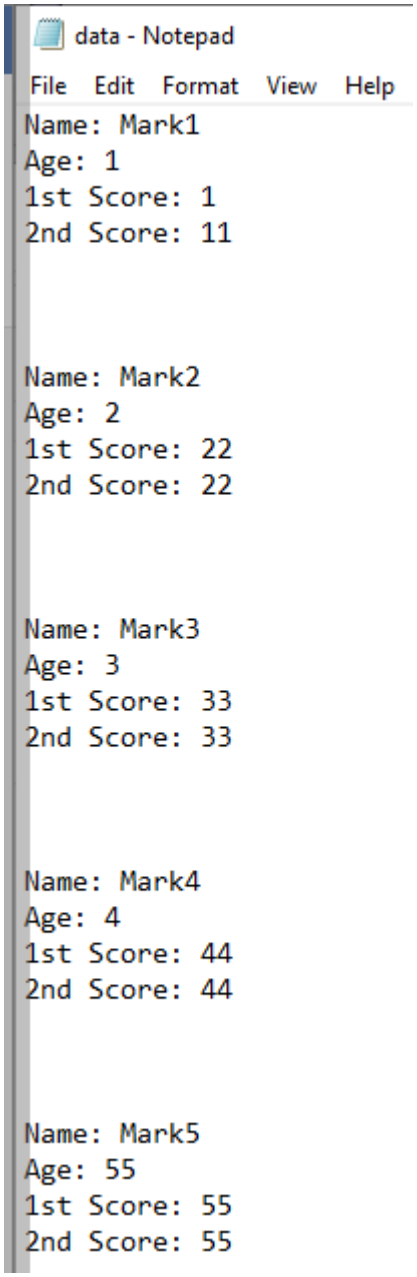
```
Enter the # of choice: 7
outfile.close command was executed
```

```
                        //cout << "Enter Name: ";
                        outFile << "Name: " << p[i].name << endl;
                        //cout << "Enter Age: ";
                        outFile << "Age: " << p[i].age << endl;
                        //cout << "Enter 1st score: ";
                        outFile <<"1st Score: " << p[i].score1 << endl;
                    // cout << "Enter 2nd score: ";
                        outFile << "2nd Score: " << p[i].score2 << endl;
                        outFile << endl << endl << endl;
                    }

                    menu();
                break;
                // FOR OUTPUTING THE AVERAGES
            case 2:
                //if age is STILL 0 then there's still no inputs
                if (p[0].age == 0) {
                    cout << "NO INPUT" << endl;
                    system ("pause");
                    menu();

                    }
                    for(int i = 0 ; i < 5 ; i++) {

                        cout << "PLAYER " << i+1 << " RECORDS: \n";
                        cout << "NICKNAME: " << p[i].name << endl;
                        cout << "AGE: " << p[i].age << endl;
                        cout << "SCORE1: " << p[i].score1 << endl;
                        cout << "SCORE2: " << p[i].score2 << endl << endl;
```



```
                    p[i].ave = (p[i].score1 +
p[i].score2) / 2;
                }
                system ("pause");
                system ("cls");
                menu();
                break;
                // TOTALLING ALL THE
AVERAGES
            case 3:
                //if age is STILL 0 then there's
still no inputs
                if (p[0].age == 0) {
                    cout << "NO INPUT" <<
endl;
                    system ("pause");
                    menu();
                }
                for(int i = 0 ; i < 5 ; i++) {
                    cout << "PLAYER " <<
p[i].name << " SCORE AVERAGE: " << endl;
                    cout << p[i].ave << endl <<
endl;
                }
                // FILE HANDLING
                outFile << endl << endl << endl
<< endl;
                for(int i = 0 ; i < 5 ; i++) {
                    outFile << "PLAYER " <<
p[i].name << " SCORE AVERAGE: " << endl;
                    outFile << p[i].ave << endl
<< endl;
                }
                // outFile.close();
                cout << "TOTAL 5 PLAYERS
AVERAGE: ";
                for(int i = 0 ; i < 5 ; i++) {
                    aveT = p[i].ave + aveT ;
                }
                cout << aveT / 5 << endl;
                system ("pause");
                system("cls");
                menu();
                break;
                // ALGO FOR PRODUCING
THE MAX AVERAGE
```

```
PLAYER Mark1 SCORE AVERAGE:
6

PLAYER Mark2 SCORE AVERAGE:
22

PLAYER Mark3 SCORE AVERAGE:
33

PLAYER Mark4 SCORE AVERAGE:
44

PLAYER Mark5 SCORE AVERAGE:
55

THE HIGHEST AVERAGE AMONG ALL PLAYER IS : Mark5
THE LOWER AVERAGE AMONG ALL PLAYER IS : Mark1
```

```
            case 4:
                //if age is STILL 0 then there's
still no inputs
                if (p[0].age == 0) {
                    cout << "NO INPUT" <<
endl;
                    system ("pause");
                    menu();
                }
                temp1 = p[0].ave;
                for(int i = 0 ; i < 5 ; i++) {
                    if(temp1 < p[i].ave){
                        temp1 = p[i].ave;
                    }
                }
                cout << "THE HIGHEST
AVERAGE AMONG ALL PLAYER IS : ";
                for(int i = 0 ; i < 5 ; i++) {
                    if(temp1 == p[i].ave){
                        cout << p[i].name <<
endl;
                        break;
                    }
                }
                // FILE HANDLING
                outFile << "THE HIGHEST
AVERAGE AMONG ALL PLAYER IS : ";
                for(int i = 0 ; i < 5 ; i++) {
                    if(temp1 == p[i].ave){
                        outFile << p[i].name <<
endl;
                        break;
                    }
                }
                system ("pause");
                system("cls");
                menu();
                break;
                //ALGO FOR PRODUCING
THE MIN AVERAGE
            case 5:
                //if age is STILL 0 then there's
still no inputs
                if (p[0].age == 0) {
                    cout << "NO INPUT" <<
endl;
```

```
                    system ("pause");
                    menu();
                }
                temp2 = p[0].ave;
                for(int i = 0 ; i < 5 ; i++) {
                    if(temp2 > p[i].ave){
                        temp2 = p[i].ave;
                    }
                }
                cout << "THE LOWER
AVERAGE AMONG ALL PLAYER IS : ";
                for(int i = 0 ; i < 5 ; i++) {
                    if(temp2 == p[i].ave){
                        cout << p[i].name <<
endl;
                        break;
                    }
                }


                outFile << "THE LOWER
AVERAGE AMONG ALL PLAYER IS : ";
                for(int i = 0 ; i < 5 ; i++) {
                    if(temp2 == p[i].ave){
                        outFile << p[i].name <<
endl;
                        break;
                    }
                }
                system ("pause");
                system("cls");
                menu();
                break;
            case 6:
                cout << "outfile.open command
was executed";
                system ("pause");
                system("cls");
                menu();
                break;
            case 7:
                cout << "outfile.close
command was executed";
                outFile.close();
                system ("pause");
                system("cls");
```

```
                    menu();
                    break;
                case 8:
                    return 0;
                    break;
            }
        }
//outFile.close();
        }
```

IV. ASSESSMENT

Department	Information Technology
Subject Code	CCS0007
Description	COMPUTER PROGRAMMING 2 FOR IT
Term/Academic Year	

Topic	File Handling
Lab Activity No	6
Lab Activity	**File Handling**
CLO	**3**

Note: The following rubrics/metrics will be used to grade students' output in the lab exercise.

Trait	**(Excellent)**	**(Good)**	**(Fair)**	**(Poor)**
Requirement Specification(30pts)	Able to identify correctly all input and output and provide alternative. **(28-20pts)**	Able to identify correctly all input and output **(25-17pts)**	Able to identify only one input or output **(22-14pts)**	Unable to identify any input and output **(20-11pts)**
Data type(20pts)	Able to apply required data type or data structure and produce correct results **(18-20pts)**	Able to apply required data type or data structure and produce partially correct results **(15-17pts)**	Able to identify required data type or data structure but does apply correctly **(12-14pts)**	Unable to identify required data type **(9-11pts)**
Input Validation(20pts)	The program works and meets all specifications. Does exception al checking for errors and out-	The program works and meets all specifications. Does some checking for errors and outof-	The program produces correct results but does not display correctly Does not check for	The program produce s incorrect results **(9-11pts)**

	of- range data **(18-20pts)**	range data **(15-17pts)**	errors and outof- range data **(12-14pts)**	
Free from syntax, logic, and runtime errors (10pts)	Unable to run program **(10pts)**	Able to run program but have logic error **(8-9pts)**	Able to run program correctly without any logic error and display inappropri ate output **(6-7pts)**	Able to run program correctly without any logic error and display appropriate output **(5pts)**
Delivery (10pts)	The program was delivered on time **(10pts)**	The program was delivered after 5 minutes from the time required. **(8-9pts)**	The program was delivered after 10 minutes from the time required. **(6-7pts)**	The program was delivered after 15 (or more) minutes from the time required. **(5pts)**
Use of Comments (10pts)	Specific purpose is noted for each function, control structure, input requirements, and output results. **(10pts)**	Specific purpose is noted for each function and control structure. **(8-9pts)**	Purpose is noted for each function. **(6-7pts)**	No comments included. **(5pts)**

Topic	File Handling
Lab Activity No	6.1
Lab Activity	File Handling
CLO	3
Requirement Specification (30pts)	
Data type (20pts)	
Input Validation (20pts)	
Free from syntax, logic, and runtime errors (10pts)	
Delivery (10pts)	
Use of Comments (10pts)	
TOTAL	