

SCHOOL OF COMPUTER SCIENCES

ACADEMIC SESSION 2016 / 2017, SEMESTER 1

CPT 111 – PRINCIPLES OF PROGRAMMING

ASSIGNMENT 3

NAME : CHONG KAR HIE

MATRIC NUMBER : 132893

GROUP : D

LECTURER’S NAME : ENCIK MOHD AZAM OSMAN

DUE DATE : 05 DECEMBER 2016

**Content**

|  |  |
| --- | --- |
| Content | Page number |
| Specification Requirements   * Inputs * Outputs * Process * Test data | 1  1  2-3  4 |
| Design of the program   * Program Structure diagram * Function description | 5  6-7 |
| C++ code (source code) | 8-41 |
| Print screen of output | 42-51 |

**Specification Requirements**

**Input:**

- Main program:-

1. Selection for the main menu
2. Selection for the display a list of students’ menu
3. Input to start over the program

-Hostel management officer:-

1. Password to allow officer to access their functions
2. Matric number of students
3. MyCSD score of students
4. Name of students
5. Course code of students
6. Contact number of students
7. Gender of students
8. Minimum score to find students who MyCSD score is higher than minimum score
9. Weighted percentage

-Students:-

1. Matric number of specific student in order to search for information and qualification

**Output:-**

1. Matric number of students
2. MyCSD score of students
3. Course of students
4. Name of students
5. Contact number of students
6. Gender of the students
7. Weighted MyCSD score
8. Number of students

**Process:-**

The question wants us to store and manage students’ information and to find whether students are qualified to get USM hostels or not. The program requires the user to have a text file named “StudentMyCSD.txt” in order to read existing students’ information from the text file to the array which is already prepared inside the program. Hostel managing officers need to enter the weighted percentage in order to calculate for the weighted score which students are required to achieve at least the minimum weighted score that is set by the officer in order to qualify to get USM hostels. Students can check their qualification status by just entering their matric numbers but for hostel managing officers’ features, they need to key in the password in order to grant access to change or modify students’ information such as MyCSD score.

For the menu screen, there is a total of 8 selections, 6 for officers, 1 for students, and 1 for the exit of the program. For every selection for hostel managing officers, they are required to enter the password in order to use the function that the program allows. For the first selection of hostel managing officers, they are allowed to add students’ information. The program will prompt the officers to enter the amount of students that will be added. Then, officers are required to enter the information of the students by following the instruction prompted by the program such as matric numbers, names, course codes, genders, contact numbers and also MyCSD scores of the students.

For the second selection, officers are allowed to check the information of all of the students such as matric numbers, names, course codes, genders, contact numbers, and also MyCSD scores of all of the students.

For the third selection of hostel managing officers, they are allowed to find a list of students by using minimum score that is needed to input, gender, qualifications, and also qualifications by course codes. For minimum score, officers are required to input the minimum score, so that they can see the information of the students which their MyCSD scores are higher than the minimum score. For gender, officers can choose either they want to show the information of male students or female students by inputting ‘M’ or ‘F” in the program. For qualifications, officers can check whether all of the students are qualified or not qualified to get USM hostels based on the weighted MyCSD score. For qualifications by course codes, officers need to input the course code in order to check the qualification of the student in the course typed, and the program will show the information of students which is qualified and not qualified to get USM hostels based on the weighted MyCSD score.

For the fourth selection of hostel managing officers, officers can search for the information of the specific student by entering their matric number. Then the program will also show the qualification status of the student by referring to the weighted MyCSD score.

For the fifth selection of hostel managing officers, officers are allowed to change the weighted score by entering the percentage of the weightage. Weighted score is important because it is used to determine whether students are qualified to get USM hostels or not. Formulae are used to calculate the weighted score. First, initialize the total MyCSD score to 0.Then the following formula is used.

total = total + score[j]

This formula should be in a for loop in order to keep adding the MyCSD score of each student to the total MyCSD score collected. Then,

average =total / number

is used to find the average MyCSD score gained by the students. After that,

weightedScore = average + ((weighted/100) \* average)

is used to find the weighted MyCSD score that students need to achieve in order to be qualified to get USM hostels.

For the sixth selection of hostel managing officers, officers are allowed to change the MyCSD score of students.

For the students, the only selection is the seventh selection. For the seventh selection for students, students can check their information and also their qualification which is based on the weighted MyCSD score to get USM hostels by entering their matric number, no password is required in this selection.

**Test data:-**

279173 MeenHo M CS 0126782662 80

279183 Tan F BIO 012678922 100

279126 Lee F TI 012678922 45

279173 Peter M TI 012678922 65

279153 Minah F EE 012678922 95

132893 Aaron M CS 0165221452 100

123896 Axun M BIO 0161155221 130

132789 Cheryll F CS 0162222335 130

132659 Phoong M BIO 01659995563 50

147852 Leong M CS 0162233545 100

132697 Leon M CS 0162354789 50

132996 Chong F CS 0162354974 100

132654 Chung M PHY 0166326975 90

**Design of the program**

**Program Structure Diagram:-**

menu

showCourse

showQualified

showGender

showScore

Display a list of students

setWeighted

showSpecific

updateScore

checkPoints

display

add

Exit program

Password page for officers

Read text file

Main function

START

**Function description:-**

1. numOfStudents

* void function, passing with arrays and reference
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int& number
* Purpose of the function: To count the number of students that exists in the text file by counting the number of lines.

1. menu

* void function
* No parameter is received.
* Purpose of the function: To output the lists available in the menu function in order to let user to select the function of the program.

1. add

* void function, passing by arrays and value
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number
* Purpose of the function: To add new students’ information into the array, then output the information to the text file.

1. display

* void function, passing by arrays and value
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number
* Purpose of the function: To display all students’ information from the text file.

1. checkPoints

* void function, passing by arrays and values
* List of parameter received: int matric[], string name[], char gender[], string course[]. String contact[], int score[], int number, double weightedScore
* Purpose of the function: To let students to check their information and also their qualification to get USM hostels.

1. setWeighted

* value returning function (double), passing by arrays and value
* List of parameter received: int score[], int number
* Purpose of the function: To set the weighted score by entering the percentage, and thus setting a minimum score that students must exceed the minimum MyCSD score in order to qualify to get USM hostels.

1. showGender

* void function, passing by arrays and value
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number
* Purpose of the function: To show the information of male students or female students.

1. showScore

* void function, passing by arrays and value
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number
* Purpose of the function: To check the information of students which exceeds the typed minimum MyCSD score which is prompted inside the function.

1. showQualified

* void function, passing by arrays and values
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore
* Purpose of the function: To show the information of students who have higher MyCSD score than the weighted score that has been set and qualified to get USM hostels.

1. showCourse

* void function, passing by arrays and values
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore
* Purpose of the function: To show the information of students who are qualified to get USM hostels based on the course code inputted.

1. showSpecific

* void function, passing by arrays and values
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore
* Purpose of the function: To show a particular student’s information and their qualification status.

1. updateScore

* void function, passing by arrays and value
* List of parameter received: int matric[], string name[], char gender[], string course[], string contact[], int score[], int number
* Purpose of the function: To update a particular student’s MyCSD score

**C++ source code:-**

#include <iostream>

#include <fstream>

#include <cstdlib>

#include <iomanip>

#include <windows.h>

#include <conio.h>

using namespace std;

void numOfStudents(int[], string[], char[], string[], string[], int[], int&);

void menu();

void add(int[], string[], char[], string[], string[], int[], int);

void display(int[], string[], char[], string[], string[], int[], int);

void checkPoints(int[], string[], char[], string[], string[], int[], int, double);

double setWeighted(int[], int);

void showGender(int[], string[], char[], string[], string[], int[], int);

void showScore(int[], string[], char[], string[], string[], int[], int);

void showQualified(int[], string[], char[], string[], string[], int[], int, double);

void showCourse(int[], string[], char[], string[], string[], int[], int, double);

void showSpecific(int[], string[], char[], string[], string[], int[], int, double);

void updateScore(int[], string[], char[], string[], string[], int[], int);

int main()

{

int n = 1000;

int matric[n], score[n], selection, listselect;

string course[n], name[n], contact[n];

char gender[n];

char proceed = 'Y';

double weightedScore = -1;

string password;

char welcome[8] = {'W','E','L','C','O','M','E','!'};

char ch;

cout << fixed << showpoint << setprecision(2);

system("color 3F");

cout << endl << endl << endl << endl << endl << endl;

cout << setw(50);

for(int t = 0; t < 8; t++)

{

cout << welcome[t];

Sleep(300); // Delay the time of showing the next thing for 300 milliseconds

}

while(proceed == 'Y' || proceed == 'y')

{

system("cls");

system("color 8E");

numOfStudents(matric, name, gender, course, contact, score, n);

cout << "\t\t\t----------Total number of students in database: " << n << "----------" << endl << endl; // Shows the total number of students available in text file

menu();

cin >> selection;

cout << endl;

while(!cin || selection < 1 || selection > 8)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input. Please try again: ";

cin >> selection;

cout << endl;

}

if(selection == 1 || selection == 2 || selection == 3 || selection == 4 || selection == 5 || selection == 6)

{

system("cls");

system("color 0A");

cout << endl << endl << endl;

cout << "\t\t\tFor hostel management officer, please enter the password: "; // Password to access to officer's functions is admin

ch = getch();

while(ch != 13) // Show typed keys as asterisks if enter key is not entered

{

password.push\_back(ch);

cout << "\*";

ch = getch();

}

cout << endl << endl;

cout << "\t\t\tValidating.";

Sleep(500);

cout << ".";

Sleep(500);

cout << ".";

Sleep(500);

cout << endl << endl;

while(password != "admin") // Password to access to officer's functions is admin

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tPassword incorrect! Please try again.";

Sleep(500);

system("cls");

cout << endl << endl << endl;

cout << "\t\t\tFor hostel management officer, please enter the password: ";

password = "";

ch = getch();

while(ch != 13)

{

password.push\_back(ch);

cout << "\*";

ch = getch();

}

cout << endl << endl;

cout << "\t\t\tValidating.";

Sleep(500);

cout << ".";

Sleep(500);

cout << ".";

Sleep(500);

cout << endl;

}

cout << endl;

cout << "\t\t\tPassword match!" << endl;

Sleep(500);

}

switch(selection)

{

case 1: add(matric, name, gender, course, contact, score, n); // Function to add students' information

continue;

case 2: display(matric, name, gender, course, contact, score, n); // Function to display all students' information

system("pause");

system("cls");

continue;

case 3: system("cls");

system("color 8F");

cout << endl << endl << endl << endl;

cout << "\t\t\tSearch a list of students by using..." << endl;

cout << "\t\t\t1. - Minimum score" << endl;

cout << "\t\t\t2. - Gender" << endl;

cout << "\t\t\t3. - Qualifications" << endl;

cout << "\t\t\t4. - Qualifications by course code" << endl;

cout << "\t\t\tPlease enter your selection: ";

cin >> listselect;

while(!cin || listselect < 1 || listselect > 4)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input. Please try again: ";

cin >> listselect;

}

switch(listselect)

{

case 1: showScore(matric, name, gender, course, contact, score, n); // Function to display information of students who have more than the minimum score typed

system("pause");

break;

case 2: showGender(matric, name, gender, course, contact, score, n); // Function to display information of students by gender

system("pause");

break;

case 3: showQualified(matric, name, gender, course, contact, score, n, weightedScore); // Function to display information of qualified students

system("pause");

break;

case 4: showCourse(matric, name, gender, course, contact, score, n, weightedScore); // Function to display information of students by course code

system("pause");

break;

default: cout << "\t\t\tInvalid input!";

}

continue;

case 4: showSpecific(matric, name, gender, course, contact, score, n, weightedScore); // Function to display information of specific students by matric number

continue;

case 5: weightedScore = setWeighted(score, n); // Function to set weighted score

continue;

case 6: updateScore(matric, name, gender, course, contact, score, n); // Function to update student's MyCSD score

continue;

case 7: checkPoints(matric, name, gender, course, contact, score, n, weightedScore); // Function to let students to check their MyCSD score and qualifications

break;

case 8: return 0;

default: cout << "\t\t\tInvalid input!" << endl << endl;

}

cout << "\t\tDo you want to start again? Y/y for yes, other keys to exit.";

cin >> proceed;

}

system("pause");

return 0;

}

void numOfStudents(int matric[], string name[], char gender[], string course[], string contact[], int score[], int& number) // Function to count the number of students available in text file

{

int i = 0;

ifstream inData;

inData.open("StudentMyCSD.txt");

while(!inData.eof()) //Read the text file

{

inData >> matric[i] >> name[i] >> gender[i] >> course[i] >> contact[i] >> score[i];

i++; // Increment of i for one time for each line of text file that has read

}

inData.close();

number = i; // Assign i to number of students and pass back to main function by reference

}

void menu()

{

cout << "\t\t\t------------------------------------------------------------" << endl;

cout << " \t\t\t\t\t 1-Hostel USM (1Hostel)"<< endl;

cout << "\t\t\t------------------------------------------------------------" << endl << endl;

cout << "\t\t\t------------------------------------------------------------" << endl;

cout << "\t\t\tFor Hostel Managing Officer..." << endl << endl;

cout << "\t\t\t1. - Add student info..." << endl;

cout << "\t\t\t2. - Display all student info..." << endl;

cout << "\t\t\t3. - Display a list of students..." << endl;

cout << "\t\t\t4. - Search for specific student..." << endl;

cout << "\t\t\t5. - Set weighted percentage..." << endl;

cout << "\t\t\t6. - Update specific student's score points..." << endl << endl;

cout << "\t\t\t------------------------------------------------------------" << endl;

cout << "\t\t\tFor students..." << endl << endl;

cout << "\t\t\t7. - Check collected points and qualification status..." << endl;

cout << endl;

cout << "\t\t\t------------------------------------------------------------" << endl;

cout << "\t\t\t8. - Exit program..." << endl;

cout << "\t\t\t------------------------------------------------------------" << endl;

cout << "\t\t\tPlease enter your choice: ";

}

void add(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number) //Function for add student's information

{

system("cls");

system("color 1F");

int num;

int i = 0;

int addmatric, addscore;

string addname, addcourse, addcontact;

char addgender;

cout << endl << endl << endl;

cout << "\t\t\tPlease enter the amount of students that you want to input: ";

cin >> num;

while(!cin || num < 0)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input. The amount of students must be in positive number." << endl;

cout << "\t\t\tPlease try again: ";

cin >> num;

}

cout << endl << endl << endl;

ofstream outData;

outData.open("StudentMyCSD.txt", ios::app);

for(int j = 0; j < num; j++)

{

cout << "\t\t\tFor student " << j+1 << " :" << endl;

cout << "\t\t\tPlease enter the matric number of student: ";

cin >> addmatric;

while(!cin || addmatric < 100000 || addmatric > 999999)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Matric number must be in 6 digits integer!" << endl;

cout << "\t\t\tPlease try again: ";

cin >> addmatric;

}

for(i = 0; i < number; i++)

{

while(addmatric == matric[i])

{

cout << "\t\t\tMatric number is already existed! Please try with another matric number: ";

cin >> addmatric;

i = 0;

while(!cin || addmatric < 100000 || addmatric > 999999)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Matric number must be in 6 digits integer!" << endl;

cout << "\t\t\tPlease try again: ";

cin >> addmatric;

}

}

}

cout << endl << endl;

cout << "\t\t\tPlease enter the name of student: ";

cin >> addname;

while(addname.find\_first\_of("`1234567890-=/\*+~!@#$%^&\*()[]{}|;'" ":/.,<>?") != string::npos) // Validation loop for non-alphabets input for name

{

cout << "\t\t\tInvalid input! Please enter again: ";

cin >> addname;

}

cout << endl << endl;

cout << "\t\t\tPlease enter the gender of student: ";

cin >> addgender;

while(addgender != 'M' && addgender != 'F')

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tGender must be in 'M'- Male and 'F'- Female only" << endl;

cout << "\t\t\tPlease try again: ";

cin >> addgender;

}

cout << endl << endl;

cout << "\t\t\tPlease enter the abbreviation of the course for student in capital letter: ";

cin >> addcourse;

while(addcourse.find\_first\_of("`1234567890-=/\*+~!abcdefghijklmnopqrstuvwxyz@#$%^&\*()[]{}|;'" ":/.,<>?") != string::npos) // Validation loop for non-alphabets inpyt for course code

{

cin.clear();

cin.ignore(100,'\n');

cout << "\t\t\tInvalid input! Please try again: ";

cin >> addcourse;

}

cout << endl << endl;

cout << "\t\t\tPlease enter the contact number of student: ";

cin >> addcontact;

while(addcontact.length() < 10 || addcontact.length() > 11 || addcontact[0] != '0' || (addcontact.find\_first\_of("abcdefghijklmnopqrstuvwxyz`-=/\*+~!@#$%^&\*()[]{}|;'" ":/.,<>?") != string::npos)) // Validation loop for non-digit input for contact number

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input. Please try again: ";

cin >> addcontact;

}

cout << endl << endl;

cout << "\t\t\tPlease enter the score obtained by student: ";

cin >> addscore;

while(!cin || addscore < 0)

{

cin.clear();

cin.ignore(100,'\n');

cout << "\t\t\tInvalid input. The score must be in positive number only." << endl;

cout << "\t\t\tPlease try again: ";

cin >> addscore;

}

cout << endl << endl;

outData << endl << addmatric << " " << addname << " " << addgender << " " << addcourse << " " << addcontact << " " << addscore;

}

outData.close();

cout << "\t\t\tProcessing.";

Sleep(500);

cout << ".";

Sleep(500);

cout << ".";

Sleep(500);

cout << endl;

cout << "\t\t\tDone!";

Sleep(500);

}

void display(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number) // Function to display all students' information

{

for(int z = 0; z < 2; z++)

{

system("cls");

cout << endl << endl << endl << endl << endl << endl << endl << endl;

cout << setw(50) << "LOADING.";

Sleep(400);

system("cls");

cout << endl << endl << endl << endl << endl << endl << endl << endl;;

cout << setw(51) << "LOADING..";

Sleep(400);

system("cls");

cout << endl << endl << endl << endl << endl << endl << endl << endl;

cout << setw(52) << "LOADING...";

Sleep(400);

}

system("cls");

system("color 8F");

int i = 0;

cout << endl;

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

}

cout << endl;

cout << "\t\t\t\t\t\tStudents found: " << i << endl << endl;

cout << "\t\t\t";

}

void showScore(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number) // Function to display students' information which is above the score inputted

{

system("cls");

system("color 8F");

int findscore;

int exist = 0;

int i = 0;

cout << endl << endl << endl;

cout << "\t\tPlease enter the minimum score that you want to find: ";

cin >> findscore;

while(!cin || findscore < 0)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Please try again: ";

cin >> findscore;

}

system("cls");

cout << endl << endl << endl;

cout << "\t\tThe students who have the score higher than " << findscore << " is: " << endl;

cout << "\t\t" <<"Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(findscore < score[i])

{

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

exist++;

}

}

if (exist == 0)

cout << "\t\t\tNo students found!" << endl;

cout << endl << "\t\t\t";

}

void showGender(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number) // Function to show students' information by gender

{

system("cls");

system("color 8F");

int i = 0;

char findgender;

cout << endl << endl << endl;

cout << "\t\tPlease enter the gender of students that you want to search" << endl;

cout << "\t\t'M' for Male, 'F' for Female: ";

cin >> findgender;

while(findgender != 'M' && findgender !='m' && findgender != 'F' && findgender != 'f')

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Please try again: ";

cin >> findgender;

}

system("cls");

cout << endl << endl << endl;

cout << "\t\t" <<"Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(findgender == gender[i])

{

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

}

}

cout << endl << "\t\t\t";

}

void showQualified(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore) // Function to show qualified students' information

{

system("cls");

system("color 8F");

int i = 0;

system("cls");

if(weightedScore == -1)

{

cout << endl << endl << endl;

cout << "\t\t\tYou have not set the weighted score yet!" << endl << endl;

cout << "\t\t\tPlease set it before proceed." << endl << endl;

cout << "\t\t\t";

return;

}

system("cls");

cout << endl << endl << endl;

cout << "\t\tThe students who are qualified to stay in USM hostel: " << endl << endl;

cout << "\t\t" <<"Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(score[i] > weightedScore)

{

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

}

}

cout << endl << endl << endl;

cout << "\t\tThe students who are not qualified to stay in USM hostel: " << endl << endl;

cout << "\t\t" <<"Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(score[i] < weightedScore)

{

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

}

}

cout << endl << "\t\t\t";

}

void showCourse(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore) // Function to show qualified students' information by course code

{

system("cls");

system("color 8F");

int i = 0;

string findcourse;

if(weightedScore == -1)

{

cout << endl << endl << endl;

cout << "\t\t\tThe weighted score has not set by hostel officer!" << endl << endl;

cout << "\t\t\tPlease contact hostel officer for more information." << endl << endl;

cout << "\t\t\t";

return;

}

cout << endl << endl << endl;

cout << "\t\tPlease enter the course code of the students that you want to find in capital letter: ";

cin >> findcourse;

while(findcourse.find\_first\_of("`1234567890-=/\*+~!abcdefghijklmnopqrstuvwxyz@#$%^&\*()[]{}|;'" ":/.,<>?") != string::npos)

{

cin.clear();

cin.ignore(100,'\n');

cout << "\t\tInvalid input! Please try again: ";

cin >> findcourse;

}

system("cls");

cout << endl << endl << endl;

cout << "\t\tThe students in specific school who are qualified to stay in USM hostel: " << endl << endl;

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(findcourse == course[i])

{

if(score[i] > weightedScore)

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

}

}

cout << endl << endl << endl;

cout << "\t\tThe students in specific school who are not qualified to stay in USM hostel: " << endl << endl;

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(findcourse == course[i])

{

if(score[i] < weightedScore)

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

}

}

cout << endl << "\t\t\t";

}

void showSpecific(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore) // Function to search specific student's information by matric number

{

system("cls");

system("color 8F");

int i = 0;

int findmatric;

int exist = 0;

if(weightedScore == -1)

{

cout << endl << endl << endl;

cout << "\t\t\tThe weighted score has not set by hostel officer!" << endl << endl;

cout << "\t\t\tPlease contact hostel officer for more information." << endl << endl;

cout << "\t\t\t";

system("pause");

return;

}

cout << endl << endl << endl;

cout << "\t\tPlease enter the matric number of the student that you want to find: ";

cin >> findmatric;

while(!cin || findmatric < 100000 || findmatric > 999999)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\tInvalid input! Please try again: ";

cin >> findmatric;

}

system("cls");

cout << endl << endl << endl;

cout << "\t\t" <<"Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(findmatric == matric[i])

{

cout << "\t\t" <<setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl << endl;

exist++;

if(score[i] > weightedScore)

cout << "\t\t\tThis student is qualified to get USM hostel." << endl;

else if(score[i] < weightedScore)

cout << "\t\t\tThis student is not qualified to get USM hostel." << endl;

}

}

if(exist == 0)

cout << endl << "\t\t\t Student not exist! Please try again." << endl;

cout << endl << "\t\t\t";

system("pause");

}

double setWeighted(int score[], int number) // Function to set weighted score

{

system("cls");

system("color 1F");

double weighted;

double total = 0;

double average;

double weightedScore;

cout << endl << endl << endl;

cout << "\t\t\tPlease enter the weightage percentage (%): ";

cin >> weighted;

while(!cin || weighted < 0 || weighted > 100)

{

cin.clear();

cin.ignore(10, '\n');

cout << "\t\t\tInvalid input. Weighted percentage should be in between (0-100)%" << endl;

cout << "\t\t\tPlease try again: ";

cin >> weighted;

}

for(int j = 0; j < number; j++)

{

total = total + score[j];

}

average = total / number;

weightedScore = average + ((weighted/100) \* average);

cout << endl << endl;

cout << "\t\t\tThe qualified status is " << weightedScore << ". " << endl;

cout << "\t\t\tStudents must score at least " << weightedScore << " or above." << endl << endl;

cout << "\t\t\t";

system("pause");

return weightedScore;

}

void updateScore(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number) // Function to update specific student's MyCSD score

{

system("cls");

system("color 1F");

int i = 0;

int exist=0;

int findmatric;

cout << endl << endl << endl;

cout << "\t\tPlease enter the matric number of the student that you want to update: ";

cin >> findmatric;

while(!cin || findmatric < 100000 || findmatric > 999999)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input. Please try again: ";

cin >> findmatric;

}

system("cls");

cout << endl << endl << endl;

for(i = 0; i < number; i++)

{

if(i == 0)

{

ofstream outData;

outData.open("StudentMyCSD.txt");

if(matric[i] == findmatric)

{

cout << "\t\tThe student's details are as follow: " << endl << endl;

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl << endl;

ofstream outData;

outData.open("StudentMyCSD.txt", ios::app);

outData << matric[i] << " " << name[i] << " " << gender[i] << " " << course[i] << " " << contact[i] << " ";

cout << "\t\t\tPlease enter the new score that you want to update: ";

cin >> score[i];

while(!cin || score[i] < 0)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Please try again: ";

cin >> score[i];

}

outData << score[i] << endl;

outData.close();

exist++;

}

else

outData << matric[i] << " " << name[i] << " " << gender[i] << " " << course[i] << " " << contact[i] << " " << score[i] << endl;

outData.close();

}

else if(i == number-1)

{

ofstream outData;

outData.open("StudentMyCSD.txt", ios::app);

if(matric[i] == findmatric)

{

cout << "\t\tThe student's details are as follow: " << endl << endl;

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl << endl;

ofstream outData;

outData.open("StudentMyCSD.txt", ios::app);

outData << matric[i] << " " << name[i] << " " << gender[i] << " " << course[i] << " " << contact[i] << " ";

cout << "\t\tPlease enter the new score that you want to update: ";

cin >> score[i];

while(!cin || score[i] < 0)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Please try again: ";

cin >> score[i];

}

outData << score[i];

outData.close();

exist++;

}

else

outData << matric[i] << " " << name[i] << " " << gender[i] << " " << course[i] << " " << contact[i] << " " << score[i];

outData.close();

}

else

{

ofstream outData;

outData.open("StudentMyCSD.txt", ios::app);

if(matric[i] == findmatric)

{

cout << "\t\tThe student's details are as follow: " << endl << endl;

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl << endl;

ofstream outData;

outData.open("StudentMyCSD.txt", ios::app);

outData << matric[i] << " " << name[i] << " " << gender[i] << " " << course[i] << " " << contact[i] << " ";

cout << "\t\tPlease enter the new score that you want to update: ";

cin >> score[i];

while(!cin || score[i] < 0)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\t\tInvalid input! Please try again: ";

cin >> score[i];

}

outData << score[i] << endl;

outData.close();

exist++;

}

else

outData << matric[i] << " " << name[i] << " " << gender[i] << " " << course[i] << " " << contact[i] << " " << score[i] << endl;

outData.close();

}

}

if(exist == 0)

cout << "\t\t\tMatric number not found! Please try again." << endl << endl;

cout << "\t\t\t";

system("pause");

}

void checkPoints(int matric[], string name[], char gender[], string course[], string contact[], int score[], int number, double weightedScore) // Function to let students to check their MyCSD score and qualifications

{

system("cls");

system("color F1");

int i = 0;

int exist = 0;

int findmatric;

char fetching[16] = {'F','e','t','c','h','i','n','g',' ','d','a','t','a','.','.','.'};

if(weightedScore == -1)

{

cout << endl << endl << endl;

cout << "\t\t\tThe weighted score has not set by hostel officer!" << endl << endl;

cout << "\t\t\tPlease contact hostel officer for more information." << endl << endl;

cout << "\t\t\t";

system("pause");

return;

}

cout << endl << endl << endl;

cout << "\t\t\tYou can use your matric number to search for your accumulated points! " << endl;

cout << "\t\t\tAnd also your qualification!" << endl;

cout << "\t\t\tPlease enter your matric number to proceed: ";

cin >> findmatric;

while(!cin || findmatric < 100000 || findmatric > 999999)

{

cin.clear();

cin.ignore(100, '\n');

cout << "\t\tInvalid input! Matric number should be in 6 digit numbers!" << endl;

cout << "\t\tPlease try again: ";

cin >> findmatric;

}

system("cls");

cout << endl << endl << endl << endl << endl << endl << endl << endl;

cout << setw(50);

for(int z = 0; z < 16; z++)

{

cout << fetching[z];

Sleep(200);

}

system("cls");

cout << "\t\t" << "Matric no." << setw (14) << "Name" << '\t' << "Gender" << setw(12) << "Course" << setw(17) << "Contact no." << setw(10) << "MyCSD" << endl;

for(i = 0; i < number; i++)

{

if(matric[i] == findmatric)

{

cout << "\t\t" << setw(9) << matric[i] << setw(15) << name[i] << setw(14) << gender[i] << setw(12) << course[i] << setw(16) << contact[i] << setw(11) << score[i] << endl;

exist++;

if(score[i] > weightedScore)

{

cout << endl << "\t\t\tGood news for you!" << endl;

cout << "\t\t\tYou're qualified to get USM hostel!" << endl << endl;

}

else

{

cout << endl << "\t\t\tBad news for you." << endl;

cout << "\t\t\tYou're not qualified to get USM hostel." << endl;

cout << "\t\t\tPlease do not give up!" << endl << endl;

}

}

}

if(exist == 0)

{

cout << endl << endl << endl;

cout << "\t\tMatric number not found! " << endl << endl;

}

cout << "\t\t";

}

**Print screen of output:-**







































