#include<iostream>

#include<Windows.h>

#include<iomanip>

#include<ios>

#include<fstream>

#include<string>

using namespace std;

struct grades {

string subject, grade\_obtained;

double credit\_hours;

};

void display\_data(int);

void Enter\_data();

void Delete\_data();

void Edit\_data();

void Key\_checker();

void Roll\_checker();

void IDpos\_searchengine();

void file\_printer(int);

void ID\_EXISTANCE(int);

void Data\_entry();

void encrypt\_data();

string Decrypt\_line();

void Store\_grades();

double grade\_point\_checker(string);

void find\_transcript();

void print\_transcript();

void encrypt\_transcript(grades);

string Decrypt\_transcript();

char ch, opt[6], E\_address[24], E\_name[15], E\_fname[15], E\_dob[9], E\_roll[7],gender, E\_subject[5], E\_grade[3];

bool search, save;

string roll, address, name, fname, dob;

int c, x, id, n = 0, y, key; double cgpa,p;

fstream studentrec\_file,Grade\_rec;

int main()

{

cout << "\n\n\n\n\n\t#" << setfill('#') << setw(95) << "#\n\t#" << setfill(' ') << setw(95) << "#\n\t#" << setfill(' ') << setw(50) << "WELCOME!" << setfill(' ') << setw(45) << "#\n\t#" << setfill(' ') << setw(95) << "#\n\t#" << setfill('#') << setw(95) << "#\n\n\n";

Sleep(3000);

system("cls");

do {

cout << "\t\t\t-----------------------\n\t\t\tWhat do you want to do?\n\t\t\t-----------------------\n\t\t\t1) Display the students record\n\t\t\t2) Add a new student to record\n\t\t\t3) Edit the records\n\t\t\t4) Erase all data\n\t\t\t5) Quit this program\n\nEnter your choice: ";

cin >> c;

switch (c) {

case 1:

do {

cout << "\t\t\t1) To display all the record\n\t\t\t2) To display the record of a specific student\n\t\t\t3) To return tomain menu\nEnter your choice: " << endl;

cin >> x;

display\_data(x);

} while (x != 3);

system("cls");

break;

case 2:

Enter\_data();

system("cls");

break;

case 3:

system("cls");

Edit\_data();

system("pause");

system("cls");

break;

case 4:

Delete\_data();

system("cls");

break;

case 5:

break;

default:

system("cls");

cout << "\t\t\tInvalid Input\n" << endl;

break;

}

} while (c != 5);

system("cls");

cout << "\n\n\n\n\n\t#" << setfill('#') << setw(95) << "#\n\t#" << setfill(' ') << setw(95) << "#\n\t#" << setfill(' ') << setw(60) << "THIS PROGRAM IS CREATED BY Zakriya, Faizan & Abdullah." << setfill(' ') << setw(35) << "#\n\t#" << setfill(' ') << setw(95) << "#\n\t#" << setfill('#') << setw(95) << "#\n\n\n";

Sleep(5000);

system("cls");

return 0;

}

void display\_data(int x) {

switch (x) {

case 1:

system("cls");

Key\_checker();

if (key == 1234) {

cout << "|" << setfill('-') << setw(159) << "|\n" << setfill(' ') << setw(10) << "Roll" << setw(13) << "Name" << setw(25) << "Gender" << setw(15) << "Father" << setw(25) << "CGPA" << setw(20) << "DOB" << setw(25) << "Address\n|" << setfill('-') << setw(158) << "|" << setfill(' ') << "\n" << setw(5);

studentrec\_file.open("new\_file\_write", ios::in);

if (!studentrec\_file)

cout << "\n\nfile not found \n";

else

file\_printer(y = 0);

cout << endl;

system("pause");

system("cls");

studentrec\_file.close();

}system("cls");

break;

case 2:

system("cls");

search = false;

Roll\_checker();

if (roll.size() == 6) {

studentrec\_file.open("new\_file\_write", ios::in);

if (studentrec\_file) {

IDpos\_searchengine();

if (search == true) {

studentrec\_file.close();

studentrec\_file.open("new\_file\_write", ios::in);

studentrec\_file.seekg(c, ios::beg);

cout << "\t\tID FOUND\n|" << setfill('-') << setw(159) << "|\n" << setfill(' ') << setw(10) << "Roll" << setw(13) << "Name" << setw(25) << "Gender" << setw(15) << "Father" << setw(25) << "CGPA" << setw(20) << "DOB" << setw(25) << "Address\n|" << setfill('-') << setw(158) << "|" << setfill(' ') << "\n" << setw(5);

file\_printer(y = 2);

cout << "\n#" << setfill('#') << setw(150) <<"#\n"<<setfill(' ')<<setw(45)<< "Transcript of student\n" << endl;

find\_transcript();

Grade\_rec.open("Transcript\_file", ios::in);

Grade\_rec.seekg(p+6, ios::beg);

print\_transcript();

}

else

cout << "ID NOT FOUND" << endl;

studentrec\_file.close();

}

else

cout << "File not found" << endl;

cout << endl;

system("pause");

}

system("cls");

break;

case 3:

break;

default:

cout << "\t\t\tInvalid Input\n" << endl;

break;

}

}

void Enter\_data() {

int j = 0;

save = false;

system("cls");

Key\_checker();

if (key == 1234) {

roll = "";

while (roll != "0" && save != true) {

cout << "\nPLEASE ENTER the following required data (use , for spaces)\n";

ID\_EXISTANCE(y = 2);

if (roll != "0" && search == false)

Data\_entry();

}

}

if (save == true) {

cout << "Data saved successfully" << endl;

Sleep(1000);

}

}

void Delete\_data() {

system("cls");

Key\_checker();

if (key == 1234) {

studentrec\_file.open("new\_file\_write");

if (!studentrec\_file)

cout << "File not found" << endl;

else {

ofstream studentrec\_file("new\_file\_write", ios::trunc);

ofstream Grade\_rec("Transcript\_file", ios::trunc);

cout << "Data successfully removed!" << endl;

}

}

studentrec\_file.close();

Sleep(2000);

}

void Edit\_data() {

string line3, replacement = "";

int j,q;

Key\_checker();

if (key = 1234) {

save = false, search = false;

while (roll != "0" && save != true) {

ID\_EXISTANCE(y = 1);

system("cls");

if (search == true) {

line3 = Decrypt\_line();

j = line3.find(':');

studentrec\_file.seekg(0);

studentrec\_file >> line3;

studentrec\_file.close();

line3.replace(c, j + 1, replacement);

ofstream studentrec\_file("new\_file\_write", ios::out);

studentrec\_file << line3;

studentrec\_file.close();

cout << "\t\tEnter the following data and use , for spaces\n";

find\_transcript();

line3 = Decrypt\_transcript();

q = line3.find(':');

line3.replace(p, q + 1, replacement);

ofstream Grade\_rec("Transcript\_file", ios::out);

Grade\_rec << line3;

Grade\_rec.close();

cout << line3; system("pause");

Data\_entry();

}

if (save == true) {

cout << "Data has been edited" << endl;

Sleep(1000);

}

}

}

}

void Key\_checker() {

::key = 0;

cout << setfill(' ') << setw(35) << "Admin key required : ";

while (key != 1234 && key != 2) {

cin >> key;

if (key != 1234 || key == 2)

cout << "\n\t\tWrong key entered\n\t\tAdmin key required or enter 2 to exit to main menu : ";

}

}

void Roll\_checker() {

roll = "";

while (roll.size() != 6 && roll != "0") {

cout << setfill(' ') << setw(35) << "Enter student's roll number : ";

cin >> roll;

if (roll.size() == 6)

break;

system("cls");

cout << "\n\t\tIncorrect ID entered\n\t\tEnter 0 to exit to menu or\n";

}

}

void IDpos\_searchengine() {

string line2; c = 0;

line2 = Decrypt\_line();

::c = line2.find(roll);

if (::c != string::npos)

search = true;

}

void ID\_EXISTANCE(int y) {

studentrec\_file.open("new\_file\_write");

if (studentrec\_file) {

search = false;

while (search != true && roll != "0") {

Roll\_checker();

if (roll.size() == 6) {

IDpos\_searchengine();

if (search == false || roll == "0") {

if (y == 1 && roll != "0") {

cout << "\nID doesn't exist in data" << endl;

system("pause");

}

system("cls");

break;

}

if (y == 2)

cout << "\nID already exist in data" << endl;

system("pause");

system("cls");

studentrec\_file.close();

}

}

}

}

string Decrypt\_line() {

string Q = "", line;

studentrec\_file.close();

studentrec\_file.seekg(c);

studentrec\_file.open("new\_file\_write", ios::in);

studentrec\_file >> line;

for (int i = 0; line[i] != '\0'; i++) {

int val = int(line[i]);

if (val == 59 || val == 58 || val == 45 || val == 44)

Q += char(val);

else

Q += char(val - 100);

}

studentrec\_file.close();

return Q;

}

void file\_printer(int a) {

while (!studentrec\_file.eof()) {

studentrec\_file >> ch;

if (ch == ';') {

cout << setw(14);

continue;

}

if (ch == ':') {

if (y == 2) {

cout << "\n";

break;

}

else {

cout << "\n" << setw(5);

continue;

}

}

if (ch == '-') {

cout << setw(7);

continue;

}

ch -= 100;

if (ch == ',')

cout << " ";

else

cout << ch;

}

}

void Data\_entry() {

std::cin.clear();

cout << "\n" << setfill(' ') << setw(35) << "Student name (Middle,Last) : ";

cin >> name;

std::cin.clear();

if(name.size()>14)

name.replace(14, name.size(), "");

if (name.size() < 14)

name.insert(name.size(), 14, ',');

cout << "\n" << setfill(' ') << setw(35) << "Student Gender (F,M,O) : ";

cin >> gender;

std::cin.clear();

cout << "\n" << setfill(' ') << setw(35) << "Father name (Middle,Last) : ";

cin >> fname;

if (fname.size() > 14)

fname.replace(14, fname.size(), "");

if (fname.size() < 14)

fname.insert(fname.size(), 14, ',');

std::cin.clear();

cout << "\n" << setfill(' ') << setw(35) << "Student's Address : ";

cin >> address;

std::cin.clear();

if (address.size() > 23)

address.replace(23, address.size(), "");

if (address.size() < 23)

address.insert(address.size(), 14, ',');

cout << "\n" << setfill(' ') << setw(35) << "Student's DOB (DD/MM/YY): ";

cin >> dob;

std::cin.clear();

encrypt\_data();

Store\_grades();

string c;

c = to\_string(cgpa);

for (int i = 0; c[i]!='\0'; i++)

c[i] += 100; c.replace(4, c.size(), "");

studentrec\_file.open("new\_file\_write");

if (!studentrec\_file) {

studentrec\_file.open("new\_file\_write", ios::out);

studentrec\_file << E\_roll << "-" << E\_name << ";" << char(gender+100) << ";" << E\_fname << ";" << c << ";" << E\_dob << ";" << E\_address << ":";

}

else {

ofstream studentrec\_file("new\_file\_write", ios::app);

studentrec\_file << E\_roll << "-" << E\_name << ";" << char(gender + 100) << ";" << E\_fname << ";" << c << ";" << E\_dob << ";" << E\_address << ":";

}

studentrec\_file.close();

cout << endl;

save = true;

}

void encrypt\_data() {

strcpy\_s(E\_roll, roll.c\_str());

strcpy\_s(E\_name, name.c\_str());

strcpy\_s(E\_fname, fname.c\_str());

strcpy\_s(E\_dob, dob.c\_str());

strcpy\_s(E\_address, address.c\_str());

for (int i = 0; E\_roll[i] != '\0'; i++)

E\_roll[i] += 100;

for (int i = 0; E\_name[i] != '\0'; i++)

E\_name[i] += 100;

for (int i = 0; E\_fname[i] != '\0'; i++)

E\_fname[i] += 100;

for (int i = 0; E\_dob[i] != '\0'; i++)

E\_dob[i] += 100;

for (int i = 0; E\_address[i] != '\0'; i++)

E\_address[i] += 100;

}

void Store\_grades() {

int totalcredithours = 0; cgpa = 0; double grade\_point, totalpoints = 0; bool verify; grades\* subject; string grade;

cout << "\n" << setw(35) << "Number of subjects : ";

cin >> n;

subject = new (nothrow) grades[n];

Grade\_rec.open("Transcrip\_file");

if (!Grade\_rec) {

Grade\_rec.open("Transcript\_file", ios::out);

Grade\_rec << ":" << E\_roll;

Grade\_rec.close();

}

else {

Grade\_rec.open("Transcript\_file", ios::app);

Grade\_rec << E\_roll;

Grade\_rec.close();

}

std::cin.clear();

for (int i = 0; i < n; i++) {

system("cls");

cout << "\n" << setw(65) << "Enter subject " << i + 1 << "name : ";

cin >> subject[i].subject;

std::cin.clear();

if (subject[i].subject.size() > 4)

subject[i].subject.replace(4, subject[i].subject.size(), "");

if (subject[i].subject.size() < 4)

subject[i].subject.insert(subject[i].subject.size(), 4, ',');

verify = false;

while (verify != true) {

cout << "\n" << setw(65) << "please enter grades from A - F, (may include +/-) : " ;

cin >> grade;

std::cin.clear();

grade[0] = toupper(grade[0]);

if (grade == "A-" || grade == "A+" || grade == "A" || grade == "B-" || grade == "B+" || grade == "B" || grade == "C-" || grade == "C+" || grade == "C" || grade == "D-" || grade == "D+" || grade == "D" || grade == "F")

verify = true;

else

cout << "\n" << setw(65) << "wrong grade entered!";

}

subject[i].grade\_obtained = grade;

cout << "\n" << setw(65) << "Enter credit hour : ";

cin >> subject[i].credit\_hours;

std::cin.clear();

grade\_point=grade\_point\_checker(subject[i].grade\_obtained);

totalpoints += (grade\_point\*subject[i].credit\_hours);

totalcredithours += subject[i].credit\_hours;

encrypt\_transcript(subject[i]);

Grade\_rec.open("Transcript\_file", ios::app);

Grade\_rec << ";" << E\_subject << "-" << E\_grade;

Grade\_rec.close();

}

Grade\_rec.open("Transcript\_file", ios::app);

Grade\_rec << ":";

Grade\_rec.close();

cgpa = totalpoints / totalcredithours;

}

double grade\_point\_checker(string x) {

int y;

if (x == "A+" || x == "A")

y = 4;

if (x == "A-")

y = 3.7;

if (x == "B+")

y = 3.3;

if (x == "B" )

y = 3.0;

if (x == "B-")

y = 2.7;

if (x == "C+")

y = 2.3;

if (x == "C")

y = 2.0;

if (x == "C-")

y = 1.7;

if (x == "D+")

y = 1.3;

if (x == "D")

y = 1.0;

if (x == "F")

y = 4;

return y;

}

void find\_transcript() {

string line4; p = 0;

line4 = Decrypt\_transcript();

::p = line4.find(roll);

}

void encrypt\_transcript(grades x) {

strcpy\_s(E\_subject, x.subject.c\_str());

strcpy\_s(E\_grade, x.grade\_obtained.c\_str());

for (int i = 0; E\_subject[i] != '\0'; i++)

E\_subject[i] += 100;

for (int i = 0; E\_grade[i] != '\0'; i++)

E\_grade[i] += 100;

}

void print\_transcript() {

while (!Grade\_rec.eof()) {

Grade\_rec>> ch;

if (ch == ';') {

cout <<"\n"<<setw(15);

continue;

}

if (ch == ':')

break;

if (ch == ',') {

cout << " ";

continue;

}

if (ch == '-') {

cout << setw(16);

continue;

}

cout << char(ch-100);

}

}

string Decrypt\_transcript() {

string Q,line5; Q = "";

Grade\_rec.close();

Grade\_rec.seekg(p);

Grade\_rec.open("Transcript\_file", ios::in);

Grade\_rec >> line5;

for (int i = 0; line5[i] != '\0'; i++) {

int val = int(line5[i]);

if (val == 59 || val == 58 || val == 45)

Q += char(val);

else

Q += char(val - 100);

}

Grade\_rec.close();

return Q;

}