**NAME:MUHAMMAD AHAD SIDDIQUE**

**OOP PROJECT**

**SUBMITTED TO:SIR IRTAZA SALEEM**

**TASK 1:**

#include <iostream>

#include <string>

#include<fstream>

using namespace std;

struct StudentData {

string name;

string address;

string father;

int roll;

int id;

int marks;

string attendance;

};

struct Course {

string courseCode;

string facultyName;

int numStudents;

StudentData students[10];

};

Course courses[3];

const int MAX\_STUDENTS = 50;

class User {

protected:

string username;

string password;

public:

User(string username, string password) {

this->username = username;

this->password = password;

}

bool authenticate(string enteredPassword) {

return (enteredPassword == password);

}

};

class Admin : public User {

public:

Admin(string username, string password) : User(username, password) {}

void offerSemester() {

string semester;

cout << "WHICH SEMESTERS? FALL OR SPRING:" << endl;

cin >> semester;

if (semester == "fall" || semester == "FALL") {

cout << "semester available are 1, 3, 5, 7" << endl;

}

else if (semester == "SPRING" || semester == "spring") {

cout << "semester available are 2, 4, 6, 8" << endl;

}

else {

cout << "Invalid semester" << endl;

}

}

void offercourse(){

for(int i=0;i<10;i++)

{

string offer;

int ce;

cout<<"SELECT ONE:\n1.OFFER A COURSE\n2.EXIT."<<endl;

cin>>ce;

if(ce==1)

{

cout<<"Enter course name:"<<endl;cin>>offer;

cout<<"Enter course code:"<<endl;cin>>courses[i].courseCode;

courses[i].numStudents = 1;

}

else if(ce==2)

{

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

cout<<"COURSE OFFERED IS:\n"<<courses[j].courseCode<<endl;

break;

}

}

}

void assignFaculty(Course\* courses, int numCourses, string courseCode, string facultyName) {

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

if (courses[i].courseCode == courseCode) {

courses[i].facultyName = facultyName;

cout << "Faculty assigned to course " << courseCode << ": " << facultyName << endl;

return;

}

}

cout << "Invalid course code" << endl;

}

void registerStudent(Course\* courses, int& numCourses) {

StudentData newStudent;

cout << endl << "REGISTER THE NEW STUDENT:" << endl;

cout << endl << "NAME:" << endl;

cin >> newStudent.name;

cout << endl << "ADDRESS:" << endl;

cin >> newStudent.address;

cout << endl << "FATHER NAME" << endl;

cin >> newStudent.father;

cout << "ROLL NUMBER TO BE ASSIGNED:" << endl;

cin >> newStudent.roll;

cout << endl << "UNIVERSITY ID:" << endl;

cin >> newStudent.id;

newStudent.marks = 0;

newStudent.attendance = "L";

string courseCode;

cout << "Enter the course code to enroll the student: ";

cin >> courseCode;

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

if (courses[i].courseCode == courseCode) {

courses[i].students[courses[i].numStudents] = newStudent;

courses[i].numStudents++;

cout << "Student enrolled in course " << courseCode << endl;

return;

}

}

cout << "Invalid course code" << endl;

}

};

void saveCoursesToFile(Course\* courses, int numCourses) {

fstream file;

file.open("courses.txt",ios::out);

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

file << courses[i].courseCode << endl;

file << courses[i].facultyName << endl;

file << courses[i].numStudents << endl;

for (int j = 0; j < courses[i].numStudents; j++) {

file<<courses[i].students[j].name<<endl;

file<<courses[i].students[j].address << endl;

file<<courses[i].students[j].father << endl;

file<<courses[i].students[j].roll << endl;

file<<courses[i].students[j].id << endl;

file<<courses[i].students[j].marks << endl;

file<<courses[i].students[j].attendance << endl;

}

file.close();

cout << "Courses data saved to file." << endl;

}

}

void loadcourses(Course\* courses, int numCourses) {

fstream file;

file.open("courses.txt",ios::in);

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

file >> courses[i].courseCode;

file >> courses[i].facultyName;

file >> courses[i].numStudents;

for (int j = 0; j < courses[i].numStudents; j++) {

string temp;

file>>courses[i].students[j].name>>temp;

file>>courses[i].students[j].address>>temp;

file>>courses[i].students[j].father>>temp;

file>>courses[i].students[j].roll>>temp;

file>>courses[i].students[j].id>>temp;

file>>courses[i].students[j].marks>>temp;

file>>courses[i].students[j].attendance>>temp;

}

file.close();

cout << "Courses data loaded from file." << endl;

}

}

int cour=0;

class Student : public User {

private:

string name;

int semester;

public:

Student(string username, string password)

: User(username, password) {}

void enrollCourses(Course\* courses, int numCourses) {

string courseCode;

cout << "Enter the course code to enroll in: ";

cin >> courseCode;

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

if (courses[i].courseCode == courseCode) {

if (courses[i].numStudents < MAX\_STUDENTS) {

courses[i].students[courses[i].numStudents].name = name;

courses[i].students[courses[i].numStudents].attendance = "L";

courses[i].numStudents++;cour++;

cout << "Enrollment successful" << endl;

return;

} else {

cout << "Course is full. Enrollment failed." << endl;

return;

}

}

}

cout << "Invalid course code" << endl;

}

void viewAttendance(Course\* courses, int numCourses) {

string courseCode;

cout << "Enter the course code: ";

cin >> courseCode;

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

if (courses[i].courseCode == courseCode) {

cout << "Attendance for course " << courseCode << ":" << endl;

for (int j = 0; j < courses[i].numStudents; j++) {

cout << courses[i].students[j].name << ": " << courses[i].students[j].attendance << endl;

}

return;

}

}

cout << "Invalid course code" << endl;

}

void generateFeeChallan() {

double transportFee = 2000;

double hostelCharges = 5000;

double ipFee = 1000;

double libraryFee = 1500;

double totalFee=transportFee+hostelCharges+ipFee+libraryFee+(cour\*2500);

cout << "Fee Challan:" << endl;

cout << "Transport Fee: pkr" << transportFee << endl;

cout << "Hostel Charges: pkr" << hostelCharges << endl;

cout << "IP Fee: pkr" << ipFee << endl;

cout << "Library Fee: pkr" << libraryFee << endl;

cout<<"credit hours: pkr"<< 3\*2500 <<endl;

cout << "-------------------------" << endl;

cout << "Total Fee: pkr" << totalFee << endl;

}

void viewGrades(Course\* courses, int numCourses,string name) {

string courseCode;

cout << "Enter the course code to view grades: ";

cin >> courseCode;

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

if (courses[i].courseCode == courseCode) {

for (int j = 0; j < courses[i].numStudents; j++) {

if (courses[i].students[j].name == name) {

cout << "Grades for course " << courseCode << ": " << courses[i].students[j].marks << endl;

return;

}

}

cout << "You are not enrolled in course " << courseCode << endl;

return;

}

}

cout << "Invalid course code" << endl;

}

};

class Faculty : public User {

public:

Faculty(string username, string password) : User(username, password) {}

void enterAttendance(Course\* courses, int numCourses, string name) {

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

if (courses[i].facultyName == name) {

cout << "Enter attendance for course " << courses[i].courseCode << ":" << endl;

for (int j = 1; j < (courses[i].numStudents); j++) {

cout << "Enter attendance for student " << courses[i].students[j].name << " (P/A/L): ";

cin >> courses[i].students[j].attendance;

}

cout << "Attendance entered successfully" << endl;

return;

}

}

}

void createAssessment() {

string assessmentType;

int totalMarks;

double weightage;

cout << "Enter assessment type (Quiz/Assignment/Project/Sessional): ";

cin >> assessmentType;

cout << "Enter total marks for the assessment: ";

cin >> totalMarks;

cout << "Enter weightage of the assessment (0.0 - 1.0): ";

cin >> weightage;

cout << "Assessment created:" << endl;

cout << "Type: " << assessmentType << endl;

cout << "Total Marks: " << totalMarks << endl;

cout << "Weightage: " << weightage << endl;

}

void addMarks(Course\* courses, int numCourses,string name) {

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

if (courses[i].facultyName == name) {

cout << "Enter Marks for course " << courses[i].courseCode << ":" << endl;

for (int j = 1; j < courses[i].numStudents; j++) {

cout << "Enter Marks for student " << courses[i].students[j].name;

cin >> courses[i].students[j].marks;

}

cout << "Marks entered successfully" << endl;

return;

}

}

}

};

int main() {

int numCourses = 1;

while (1) {

Admin admin("admin", "adminpass");

Student student("ahad", "ahadpass");

Student student1("zayan", "zayanpass");

Faculty faculty("irtaza", "irtazapass");

Faculty faculty1("javaid", "javaidpass");

while(1)

{

int wh;

cout<<"1.LOGIN PAGE:\n2EXIT:\n";

cin>>wh;

loadcourses(courses, numCourses);

if(wh==1)

{

string username, password;

cout << "Enter username: ";

cin >> username;

cout << "Enter password: ";

cin >> password;

if (admin.authenticate(password)) {

while (1) {

int cho;

cout << "Admin logged in" << endl;

cout << "Choose an option:\n1. Offer semester.\n2. Offer courses.\n3. Assign faculty.\n4. Register new student.\n5.EXIT\n";

cin >> cho;

if (cho == 1)

admin.offerSemester();

else if (cho == 2)

{

admin.offercourse();

}

else if (cho == 3) {

string courseCode, facultyName, fpass;

cout << "Enter course code to which faculty is to be assigned: " << endl;

cin >> courseCode;

cout << "Enter faculty member name: " << endl;

cin >> facultyName;

cout<<"Enter password for faculty"<<endl;;cin>>fpass;

admin.assignFaculty(courses, numCourses, courseCode, facultyName);

Faculty faculty("facultyName" ,"fpass");

} else if (cho == 4) {

string id,spass;

admin.registerStudent(courses, numCourses);

cout<<"Enter student user ID"<<endl;cin>>id;

cout<<"Enter password :"<<endl;cin>>spass;

Student student("id","spass");

} else if(cho==5)

break;

else {

cout << "INVALID INPUT:" << endl;

}

}

} else if (student.authenticate(password)) {

while (1) {

int choi;

cout << "Choose an option:" << endl;

cout << "1. Enroll in a course.\n2. View attendance.\n3. Generate fee challan.\n4. View grades.\n5. Exit" << endl;

cin >> choi;

if (choi == 1)

student.enrollCourses(courses, numCourses);

else if (choi == 2)

student.viewAttendance(courses, numCourses);

else if (choi == 3)

student.generateFeeChallan();

else if (choi == 4)

student.viewGrades(courses, numCourses, username);

else if (choi == 5)

break;

else {

cout << "INVALID INPUT" << endl;

}

}

}

else if (student1.authenticate(password)) {

while (1) {

int choi;

cout << "Choose an option:" << endl;

cout << "1. Enroll in a course.\n2. View attendance.\n3. Generate fee challan.\n4. View grades.\n5. Exit" << endl;

cin >> choi;

if (choi == 1)

student.enrollCourses(courses, numCourses);

else if (choi == 2)

student.viewAttendance(courses, numCourses);

else if (choi == 3)

student.generateFeeChallan();

else if (choi == 4)

student.viewGrades(courses, numCourses, username);

else if (choi == 5)

break;

else {

cout << "INVALID INPUT" << endl;

}

}

} else if (faculty.authenticate(password)) {

cout << "Faculty logged in" << endl;

while (1) {

int choic;

cout << "Enter your choice:\n1. Enter attendance.\n2. Create assessment.\n3. Add marks for any student.\n4. Exit\n\n";

cin >> choic;

if (choic == 1)

faculty.enterAttendance(courses, numCourses,username);

else if (choic == 2)

faculty.createAssessment();

else if (choic == 3)

faculty.addMarks(courses, numCourses, username);

else if (choic == 4)

break;

else

cout << "INVALID INPUT" << endl;

}

}

else if (faculty1.authenticate(password)) {

cout << "Faculty logged in" << endl;

while (1) {

int choic;

cout << "Enter your choice:\n1. Enter attendance.\n2. Create assessment.\n3. Add marks for any student.\n4. Exit\n\n";

cin >> choic;

if (choic == 1)

faculty.enterAttendance(courses, numCourses,username);

else if (choic == 2)

faculty.createAssessment();

else if (choic == 3)

faculty.addMarks(courses, numCourses, username);

else if (choic == 4)

break;

else

cout << "INVALID INPUT" << endl;

}

}

}

else if(wh==2)

{

saveCoursesToFile(courses,numCourses) ;

cout<<"DATA SAVED TO FILE:";

exit(0);

}

}

}

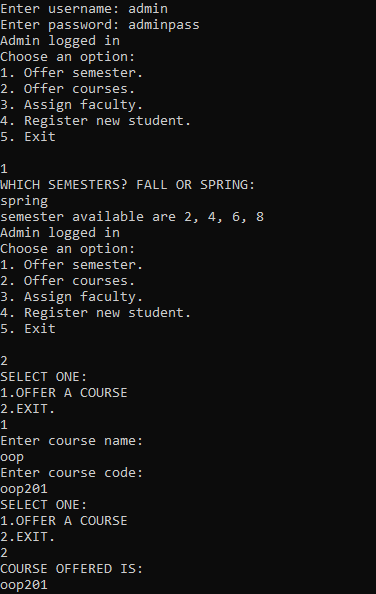
return 0;

}

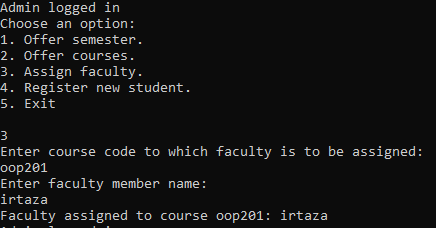
**OUTPUTS:**

**Admin:**

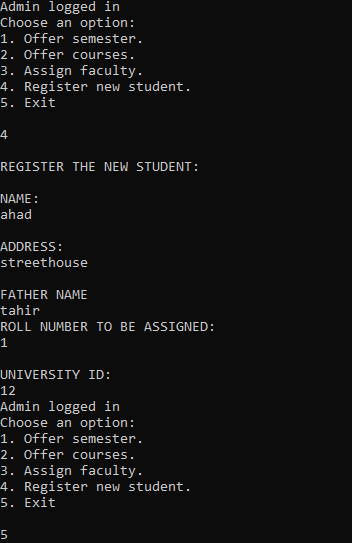
Semester & course offer:



Assigning faculty:

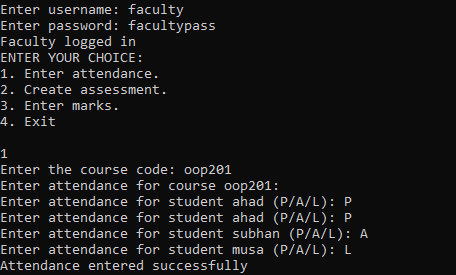


REGISTERING NEW STUDENT and then exiting:

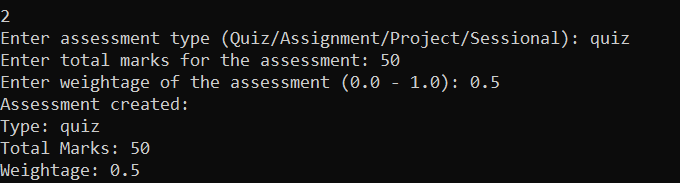


**FACULTY :**

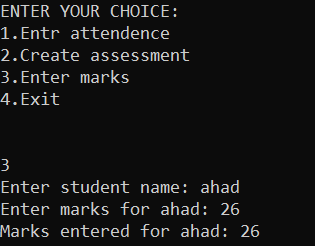
ENTERING ATTENDENCE:



CREATING ASSESSMENT:

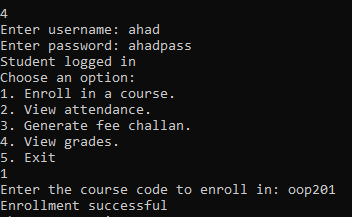


ENTERING MARKS:

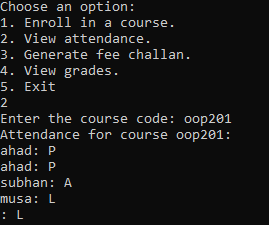


**STUDENT:**

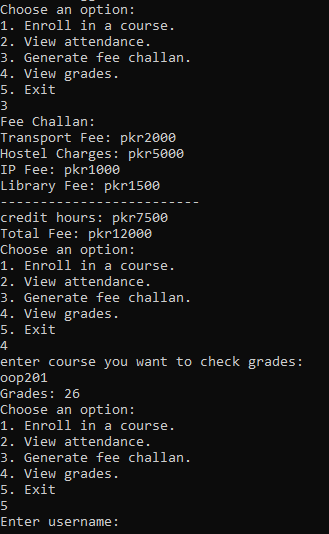
Course enrollment:



VIEWING ATTENDENCE:



Viewing grades and generating fee challan:



THE END ☹