

Object Oriented Programming Lab

Lab Task 2

Submitted by: 241484\_Huzaifa Basit

To: Sir Mahaz

Cyber Security Fall 2024-B

C++ code:

#include<iostream>

using namespace std;

// Enum for course types

enum CourseType { MATH, SCIENCE, ENGLISH, HISTORY, COMPUTER\_SCIENCE };

// Union to store either numerical or letter grade

union Grade {

int numerical\_grade;

char letter\_grade;

};

// Typedef for Course structure

typedef struct {

string course\_name;

int course\_code;

CourseType type;

Grade grade;

} Course;

// Typedef for Student structure

typedef struct {

string student\_name;

int student\_id;

Course courses[5];

int courseCount;

} Student;

// Function to add a student record

void add\_record(Student &students) {

cin.ignore();

cout << "Enter student name: ";

getline(cin, students.student\_name);

cout << "Enter student ID: ";

cin >> students.student\_id;

cout << "Enter number of courses (max 5): ";

cin >> students.courseCount;

if (students.courseCount > 5) students.courseCount = 5;

for (int i = 0; i < students.courseCount; i++) {

cin.ignore();

cout << "Enter course name: ";

getline(cin, students.courses[i].course\_name);

cout << "Enter course code: ";

cin >> students.courses[i].course\_code;

int typeChoice;

cout << "Select course type (0: MATH, 1: SCIENCE, 2: ENGLISH, 3: HISTORY, 4: COMPUTER\_SCIENCE): ";

cin >> typeChoice;

students.courses[i].type = (CourseType)typeChoice;

cout << "Enter 1 for numeric grade, 2 for letter grade: ";

int gradeChoice;

cin >> gradeChoice;

if (gradeChoice == 1) {

cout << "Enter numeric grade: ";

cin >> students.courses[i].grade.numerical\_grade;

} else {

cout << "Enter letter grade (A/B/C/D/F): ";

cin >> students.courses[i].grade.letter\_grade;

}

}

}

// Function to display student information

void display\_record(Student &students) {

cout << "\nStudent Name: " << students.student\_name << "\nStudent ID: " << students.student\_id << "\n";

for (int i = 0; i < students.courseCount; i++) {

cout << "Course: " << students.courses[i].course\_name << "\n";

cout << "Code: " << students.courses[i].course\_code << "\n";

cout << "Grade: " << (students.courses[i].grade.letter\_grade) << "\n";

}

}

// Function to calculate GPA

float calculate\_gpa(Student &students) {

float total = 0.0;

int count = 0;

for (int i = 0; i < students.courseCount; i++) {

total += students.courses[i].grade.numerical\_grade;

count++;

}

return (count > 0) ? (total / count) : 0.0;

}

int main() {

bool flag = false;

int choice;

int size;

cout << "Enter the number of students in your system: ";

cin >> size;

Student students[size];

int counter = 0;

do {

cout << "\tStudent Management System" << endl;

cout << "1. Add a new Student Record" << endl;

cout << "2. Display Student Information" << endl;

cout << "3. Calculate GPA" << endl;

cout << "4. Exit" << endl;

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

cout<<"\tAdding a student Record"<<endl;

if (counter < size) {

add\_record(students[counter]);

counter++;

} else {

cout << "Maximum student limit reached!" << endl;

}

break;

case 2:

cout<<"\tDisplaying student information"<<endl;

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

display\_record(students[i]);

}

break;

case 3:

cout<<"\tCalculating GPA"<<endl;

int id;

cout << "Enter student ID to calculate GPA: ";

cin >> id;

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

if (students[i].student\_id == id) {

cout << "GPA: " << calculate\_gpa(students[i]) << "\n";

}

}

break;

case 4:

cout<<"\tExiting the program"<<endl;

flag = true;

break;

default:

cout << "Invalid Choice! Try again." << endl;

}

} while (!flag);

return 0;

}

Output:



