**OBJECT ORIENTED CONCEPT & PROGRAMMING**

**(SE-201) LAB-6**

**TAQI HAIDER\_CSIT\_SECTION:B\_ROLL#CT-22092**

**Exercise:-**

**Q1:-**

#include <iostream>

using namespace std;

class rectangle {

    float length, width, area, parameter;

public:

    rectangle() : length(0), width(0) {}

    void setData(float len, float wid) {

        length = len;

        width = wid;

    }

    void calculateArea() {

        area = length \* width;

    }

    void calculateParameter() {

        parameter = 2 \* (length + width);

    }

    float getArea() const {

        return area;

    }

    float getParameter() const {

        return parameter;

    }

    void display() {

        cout << "Area Of Rectangle is: " << getArea() << endl;

        cout << "Parameter of Rectangle is: " << getParameter() << endl;

    }

};

class dormRoom {

    int room\_no;

    int capacity;

    bool isOccupied;

public:

    dormRoom(int x, int y) : room\_no(x), capacity(y), isOccupied(false) {}

    int getRoomNumber() const {

        return room\_no;

    }

    int getRoomCapacity() const {

        return capacity;

    }

    void occupy() {

        isOccupied = true;

        cout << "Room is occupied: " << isOccupied << endl;

    }

    void vacate() {

        isOccupied = false;

        cout << "Room is occupied: " << isOccupied << endl;

    }

};

int main() {

    rectangle r1;

    float len, wid;

    cout << "Enter Length Of Rectangle: ";

    cin >> len;

    cout << "Enter Width Of Rectangle: ";

    cin >> wid;

    r1.setData(len, wid);

    r1.calculateArea();

    r1.calculateParameter();

    r1.display();

    cout << endl;

    cout << "Dorm Room Information:" << endl;

    dormRoom d1(5, 3);

    cout << "Room Number:  " << d1.getRoomNumber() << endl;

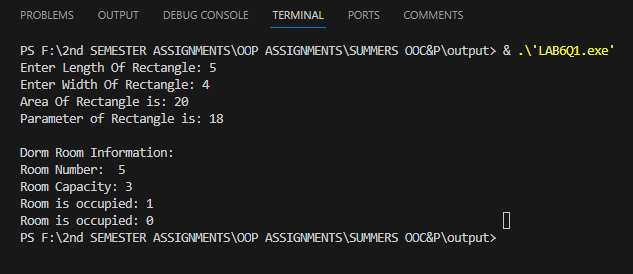
    cout << "Room Capacity: " << d1.getRoomCapacity() << endl;

    d1.occupy();

    d1.vacate();

    return 0;

}



**Q2:-**

#include <iostream>

using namespace std;

class mealBill

{

    float meal\_bill,sales\_tax,Total\_bill,payment,change;

    string restaurant\_name;

public:

    mealBill(string x) : restaurant\_name(x)

    {

        cout << "Enter Meal Bill: ";

        cin >> meal\_bill;

    }

    void bill()

    {

        cout << "Bill" << endl;

        cout << "\tMeal Bill: " << meal\_bill << endl;

        sales\_tax = (meal\_bill \* 0.16); // sale tax is 16%

        cout << "\tTax: " << sales\_tax << endl;

        cout << "\tTip: " << (meal\_bill \* 0.15) << endl;

        Total\_bill = meal\_bill + sales\_tax + (meal\_bill \* 0.15);

        cout << "\tTotal cost: " << Total\_bill << endl;

        cout << endl;

    }

    void receipt()

    {

        cout << "Enter Payment Paid: ";

        cin >> payment;

        cout << "Receipt" << endl;

        cout << "\t Total cost: " << Total\_bill << endl;

        cout << "\t Payment: " << payment << endl;

        change = payment - Total\_bill;

        cout << "\t Change: " << change << endl

             << endl;

        cout << "\t Thank you for dining at " << restaurant\_name << endl;

    }

};

int main()

{

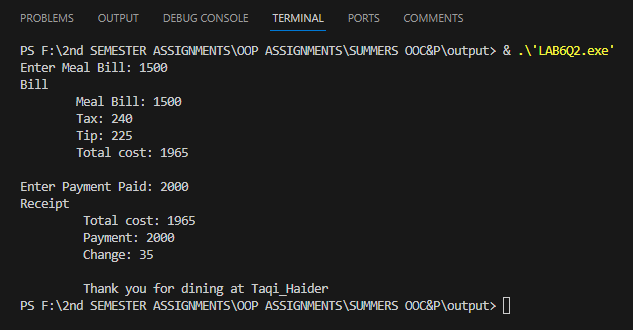
    mealBill customer("Taqi\_Haider");

    customer.bill();

    customer.receipt();

    return 0;

}

****

**Q3 :-**

#include <iostream>

using namespace std;

class management\_system

{

    float meal\_bill;

    float sales\_tax;

    float Total\_bill;

    float payment;

    float change;

    float current\_balance;

    string name;

    string Student\_Id;

public:

    management\_system(float u, string v, string w) : current\_balance(u), name(v), Student\_Id(w)

    {

        cout << "Enter Meal Bill: ";

        cin >> meal\_bill;

    }

    void bill()

    {

        cout << "Bill" << endl;

        cout << "\tMeal Bill: " << meal\_bill << endl;

        sales\_tax = (meal\_bill \* 0.06); // sale tax is 16%

        cout << "\tTax: " << sales\_tax << endl;

        cout << "\tTip: " << (meal\_bill \* 0.15) << endl;

        Total\_bill = meal\_bill + sales\_tax + (meal\_bill \* 0.15);

        cout << "\tTotal cost: " << Total\_bill << endl;

        cout << endl;

    }

    void updateBalance()

    {

        current\_balance = current\_balance + Total\_bill;

        current\_balance = current\_balance - payment;

    }

    void receipt()

    {

        cout << "Charger the meal to " << name << "'s account \n" << endl;

        cout << "Enter Payment Paid: ";

        cin >> payment;

        updateBalance();

        cout << "Receipt" << endl;

        cout << "\t Total cost: " << Total\_bill << endl;

        cout << "\t Payment: " << payment << endl;

        if (payment == 0)

        {

            change = payment;

        }

        else

        {

            change = payment - Total\_bill;

        }

        cout << "\t Change: " << change << endl

             << endl;

        cout << "\t Thank you for dining at University Summit " << endl;

        cout<<endl;

    }

    void showStudentData()

    {

        cout << "Current Status of Ahmed's account:" << endl;

        cout << "ID: " << Student\_Id << endl;

        cout << "Name: " << name << endl;

        cout << "Balance: " << current\_balance << endl;

    }

};

int main()

{

    management\_system Ahmed\_Ali(2000, "Ahmed", "CT-22092");

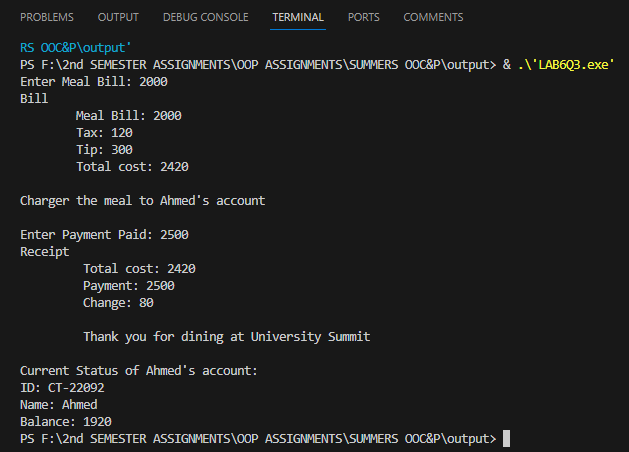
    Ahmed\_Ali.bill();

    Ahmed\_Ali.receipt();

    Ahmed\_Ali.showStudentData();

    return 0;

}



**Q4:-**

#include <iostream>

using namespace std;

class StudentRecord {

    string studentId;

    int numCourses;

    float \*creditHours;

    float totalCreditHours;

    float gradePoints;

    float totalGradePoints;

    char \*grades;

    float CGPA;

public:

    StudentRecord(string id) : studentId(id), creditHours(nullptr), gradePoints(0.0), CGPA(0.0), totalCreditHours(0.0), totalGradePoints(0.0) {}

    void setData() {

        cout << "Enter Number Of Courses in this semester: ";

        cin >> numCourses;

        creditHours = new float[numCourses];

        grades = new char[numCourses];

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

            cout << "Enter credit hours of the course: ";

            cin >> creditHours[i];

            cout << "Enter grade in the course: ";

            cin >> grades[i];

            totalCreditHours += creditHours[i];

        }

    }

    void updateRecords(float newCredits, float newGradePoints) {

        totalCreditHours += newCredits;

        totalGradePoints += newGradePoints;

    }

    void calculateCGPA() {

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

            if (grades[i] == 'A' || grades[i] == 'a') {

                gradePoints = creditHours[i] \* 4;

                totalGradePoints += gradePoints;

            } else if (grades[i] == 'B' || grades[i] == 'b') {

                gradePoints = creditHours[i] \* 3;

                totalGradePoints += gradePoints;

            } else if (grades[i] == 'C' || grades[i] == 'c') {

                gradePoints = creditHours[i] \* 2;

                totalGradePoints += gradePoints;

            } else if (grades[i] == 'D' || grades[i] == 'd') {

                gradePoints = creditHours[i] \* 1;

                totalGradePoints += gradePoints;

            } else if (grades[i] == 'F' || grades[i] == 'f') {

                gradePoints = creditHours[i] \* 0;

                totalGradePoints += gradePoints;

            } else {

                cout << "Invalid Input" << endl;

            }

        }

        CGPA = totalGradePoints / totalCreditHours;

        updateRecords(totalCreditHours, totalGradePoints);

    }

    void display() {

        cout << "Student Id: " << studentId << endl;

        cout << "Total credit hours: " << totalCreditHours << endl;

        cout << "Total grade points: " << totalGradePoints << endl;

        cout << "CGPA: " << CGPA << endl;

    }

    ~StudentRecord() {

        delete[] creditHours;

        delete[] grades;

    }

};

int main() {

    string studentId;

    cout << "Enter Student Id: ";

    cin >> studentId;

    StudentRecord student(studentId);

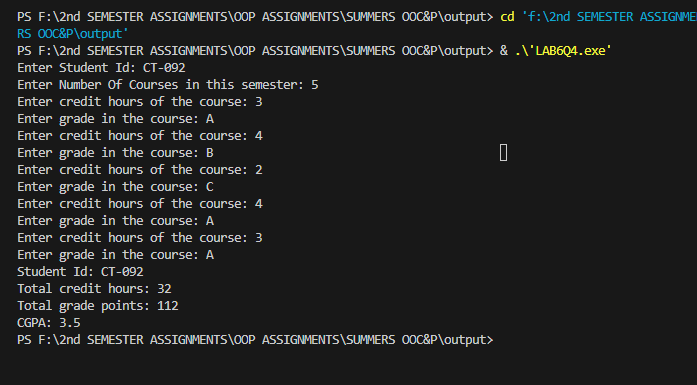
    student.setData();

    student.calculateCGPA();

    student.display();

    return 0;

}

****

**Q5:-**

#include<iostream>

using namespace std ;

class student\_record

{

    string student\_name;

    string student\_Id;

    int no\_of\_courses;

    float \*credit\_hours;

    float total\_credit\_hours;

    float grade\_points;

    float total\_gradepoint;

    char \*grade;

    float Cgpa;

public:

    student\_record(string x, string y) : student\_name(x), student\_Id(y), credit\_hours(nullptr), grade\_points(0.0), Cgpa(0.0), total\_credit\_hours(0.0), total\_gradepoint(0.0) {

        cout<<student\_name<<"'s GPA is:  "<<Cgpa<<endl;

        cout<<endl;

    }

    void setData()

    {

        cout << "Enter Number Of Courses in this semester: ";

        cin >> no\_of\_courses;

        credit\_hours = new float[no\_of\_courses];

        grade = new char[no\_of\_courses];

        for (int i = 0; i < no\_of\_courses; i++)

        {

            cout << "Enter credit hours of the course: ";

            cin >> credit\_hours[i];

            cout << "Enter grade in the course: ";

            cin >> grade[i];

            total\_credit\_hours = total\_credit\_hours + credit\_hours[i];

        }

    }

    void grading\_system()

    {

        for (int i = 0; i < no\_of\_courses; i++)

        {

            if (grade[i] == 'a' || grade[i] == 'A')

            {

                grade\_points = credit\_hours[i] \* 4;

                total\_gradepoint = total\_gradepoint + grade\_points;

            }

            else if (grade[i] == 'b' || grade[i] == 'B')

            {

                grade\_points = credit\_hours[i] \* 3;

                total\_gradepoint = total\_gradepoint + grade\_points;

            }

            else if (grade[i] == 'c' || grade[i] == 'C')

            {

                grade\_points = credit\_hours[i] \* 2;

                total\_gradepoint = total\_gradepoint + grade\_points;

            }

            else if (grade[i] == 'd' || grade[i] == 'D')

            {

                grade\_points = credit\_hours[i] \* 1;

                total\_gradepoint = total\_gradepoint + grade\_points;

            }

            else if (grade[i] == 'f' || grade[i] == 'F')

            {

                grade\_points = credit\_hours[i] \* 0;

                total\_gradepoint = total\_gradepoint + grade\_points;

            }

            else

            {

                cout << "Invalid Input" << endl;

            }

        }

        Cgpa = total\_gradepoint / total\_credit\_hours;

        cout<<"\n"<<endl;

    }

    void display()

    {

        cout << student\_name <<"'s 1st Semester grade point and gpa is: "<< total\_gradepoint<<", "<<total\_credit\_hours<< endl;

        cout<< "Student Id: "<<student\_Id<<"\tCredit Hours: "<<total\_credit\_hours<<"\tGrade Points: "<<total\_gradepoint<<"\tGPA: "<<Cgpa<<endl;

    }

     ~student\_record()

    {

        delete[] credit\_hours;

        delete[] grade;

    }

};

int main()

{

    string studentName, studentId;

    cout << "Enter Student Name: ";

    getline (cin, studentName);

    cout << "Enter Student Id: ";

    getline (cin, studentId);

    student\_record s1( studentName, studentId);

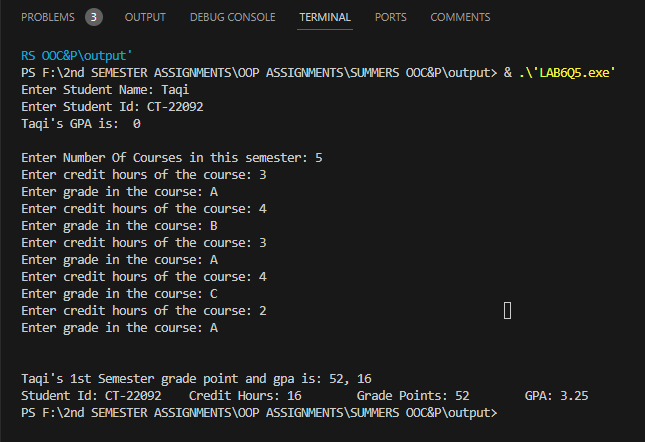
    s1.setData();

    s1.grading\_system();

    s1.display();

    return 0;

}

****

**Q6:-**

#include <iostream>

using namespace std;

class Person {

private:

    string name;

    int age;

    string gender;

public:

    Person(string n, int a, string g) : name(n), age(a), gender(g) {}

    virtual void displayData() {

        cout << "Name: " << name << endl;

        cout << "Age: " << age << endl;

        cout << "Gender: " << gender << endl;

    }

};

class Student : public Person {

private:

    string studentId;

public:

    Student(string id, string n, int a, string g) : Person(n, a, g), studentId(id) {}

    void displayStudentRecord() {

        cout << "NED Student Record " << endl;

        cout << endl;

    }

    void displayData() override {

        cout << "Student ID: " << studentId << endl;

        Person::displayData();

    }

};

class GraduateStudent : public Student {

private:

    int graduationYear;

    string degreeName;

public:

    GraduateStudent(string degree, int year, string id, string n, int a, string g) : Student(id, n, a, g), degreeName(degree), graduationYear(year) {}

    void displayData() override {

        cout << "NED Alumni Record" << endl;

        cout << endl;

        Student::displayData();

        cout << "Student Graduated in year " << graduationYear << endl;

        cout << "Bachelors of " << degreeName << endl;

    }

};

int main() {

    cout << endl;

    Student student1("CT-22092", "Taqi", 20, "Male");

    student1.displayStudentRecord();

    student1.displayData();

    cout << endl;

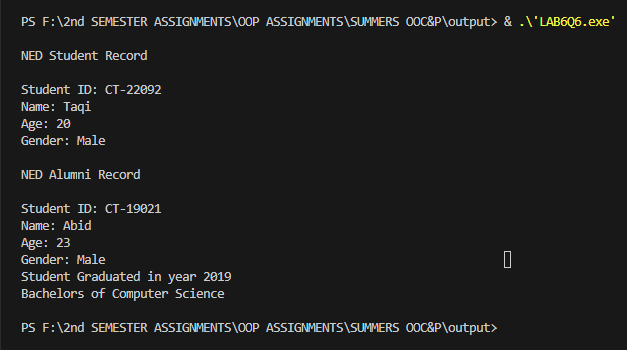
    GraduateStudent gradStudent1("Computer Science", 2019, "CT-19021", "Abid", 23, "Male");

    gradStudent1.displayData();

    cout << endl;

    return 0;

}

****