Q> Calculate the Quadi Cgpa of Student ..

```
import java.util.scanner;
public class cgpa{
    public static void main(string[] args){
        scanner scanner = new
        scanner (system.in);

        System.out.print ("Enter the
number of subjects:");
        int numsubjects = scanner.
        nextInt();

        double[] grade points = new
double[num subjects];
        int[] credits = new int[num
    subjects];
        int[] credits = new int[numsubjects];
        int totalcredits = 0;
        double total = 0;

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

            System.out.print("Enter the grade
points for subject" + (i+1)+":");
            gradepoints[i] = scanner.nextDouble()

            System.out.print("Enter the credits
for subject" + (i+1) +":");
            credits[i] = scanner.nextInt();
            total += gradePoints[i] * credits[i];
```

```
totalCredits += credits[i];
}
double cgpa = total / totalCredits;
System.out.printf("Your CGPA is: "
        + cgpa);
}
}
```

output:

```
Enter the number of subjects: 3
Enter the grade points for subject 1: 9
Enter the credits for subject 1: 4
Enter the grade points for subject 2: 9
Enter the credits for subject 2: 4
Enter the grade points for subject
3: 9
Enter the grade points
credits for subject 3: 4
Your CGPA is: 9.0
```

Quadratic Equation

```
import java.util.scanner;
class quadratic{
    float d;
    scanner sc = new scanner(system.in);
    void check()
    {
        system.out.println("enter the values of
            a,b,and c");
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();

        if(a==0){
            system.out.println("invalid
        equation");
        }
        else{
            d = b*b - 4*a*c;
            system.out.println(d);
            system.out.println("the solutions
                are");

            if(d>0){
                system.out.println("roots are
            unique");
                double r1 = (-b + Math.sqrt(d))/(2*a);
                double r2 = (-b - Math.sqrt(d))/
                        (2*a);
                system.out.println("r1 " "+r2);
        }
```

```
if(d==0){
        system.out.println("roots are
equal");
    double r = -b/(2*a);
    System.out.println(r);
}
if(d<0){
    System.out.println("roots are
imaginary");
    double r1 = Math.sqrt(-d)/(2*a);
    double r2 = (-b)/(2*a);

    System.out.println("r2 +" + "i" + r1 +
"i" + r2 + "-i" + r1);
    }

}
}
public class Main{ Quadratic{
    public static void main(String[] args){
        quadratic q1 = new quadratic();
        q1.check();
    }
}
```

Output

Enter the values of a, b, c
2
-6
7
Roots are unique
Enter the values of a, b, c
10
5
-10
Roots are imaginary.
Enter the values of a, b, c
2
4
2
Roots are equal
Enter the values of a, b, c
2
5
3
1.0
Roots are unique

3] create a class Book which contains four members: name, author, price, num-pages. include a Constructor to set the values for the members include methods to set & get the details of the objects. include a to string() method that could display the complete details of the book. Develop a Java program to create an book objects.

→

```
import java.util.scanner;
class Book{
    private string name;
    private string author;
    private double price;
    private int numpages;

public Book (String name, string
    author, double price, int numpages)
{
    this.name = name;
    this.author = author;
    this.price = price;
    this.numpages = numpages;
}
public void setname (string name){
    this.name = name;

public void setauthor (String author){
    this.author = author
```

```
public void setPrice(double price){
    this.price = price;
}
public void setNumPages(int numpages){
    this.numpages = numpages;
}
public String getName(){
    return name;
}
public String getAuthor(){
    return author;
}
public double getPrice(){
    return price;
}
public int getNumPages(){
    return numpages;
}
@Override
public String toString(){
    return "Book Details: \n" +
    "Name: " + name + "\n" +
    "author: " + author + "\n" +
    "price: " + price + "\n" +
    "Num of pages: " + numpages + "\n";
}
}
public class Main {
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
```

```
System.out.println("Enter number of Books: ");
int n = sc.nextInt();
sc.nextLine();
Book[] books = new Book[n];

for(int i=0; i<n; i++){
System.out.println("Enter details for Book " + (i+1) + ":");
System.out.println("Enter Book name:");
String name = scanner.nextLine();
sc.nextLine();
System.out.println("Enter author name:");
String author = sc.nextLine();

System.out.println("Enter price:");
double price = sc.nextDouble();

System.out.println("Enter numpages:");
int numPages = scanner.nextInt();
scanner.nextLine();
book[i] = new Book(name, author, price, numpages);
System.out.println(book[i].getDetails());
}
scanner.close();
}
```

Enter the number of books : 2
Enter the details of book 1 :
Enter Book name : The Night Circus
Enter author : Erin Morgenstern
Enter price : 400.00
Enter the number of pages : 200.

Enter the details of book 2 :
Enter Book name : Atomic Habits
Enter author : James Clear
Enter price : 600
Enter the number of pages

Details of Books :

Book [name : The Night Circus ; author : Erin Morgenstern, price : 400.0, pages : 200]
Book [name : Atomic Habits ; author : James Clear ; price : 600.0 pages : 300]

8/11/24

4] Develop a java program to create abstract class named shap that contains two integers & an empty method named print area(). provide 3 classes named rectangle, Triangle, circle such that each one of it extends class shape.

```
import Java.Util.scanner;

abstract class shape{
    int s1, s2;
    abstract void printarea();

class Rectangle extends shape{
    Rectangle (int s1, int s2){
    this.s1 = s1;
    this.s2 = s2; }
    void printarea(){
        system.out.println("area of
            rectangle is" + (s1*s2));}
}

class Triangle extends shape{
    Triangle (int s1, int s2)
    this.s1 = s1
    this.s2 = s2; }
    Void printarea(){
        System.out.println(area of
    triangle is" + (0.5*s1*s2));}}
```

```
}
class Circle extends Shape{
    Circle (int s1, int s2) {
    this.s1 = s1;
    this.s1 = s2;
    void printarea() {
    System.out.println("area of
        Circle is " + ( 3.14 * s1 * s1))}
}

class shapemain{

public static void main(String [] args)
{
    Rectangle rectangle = new Rectangle(5,4);
    rectangle.printarea();
    Triangle triangle = new Triangle(5,4);
    triangle.printarea();
    Circle Circle = new circle(5.f);
    Circle.printarea();}}
```

area of rectangle is 20
area of triangle is 0
area of Circle is 5-

14/11/24

Develop a java program to create a

5) class Bank that maintains two kinds of account for its customers, one called savings account & the other current account. The savings account provides compound interest & withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should maintain a minimum balance & if the balance falls below this level.

```
class Account {
    string customerName;
    string acc_no;
    string acc_Type;
    double balance;

public Account (string customerName,
    string acc_no, string acc_Type,
    double initialbalance) {
    this.customername = customer name;
    this.acc_no = acc_no;
    this.acc_Type = acc_Type;
    this.balance = initial balance;
}
public void deposit (double amount) {
    if (amount > 0) {
        balance += amount;
        S.O.P ("Deposited:" + amount);
    }
    else {
        S.O.P ("Invalid deposit
                amount:");
    }
}
```

```
public void displayBalance(){
    S.O.P("Balance:" + balance);
}
}

class savAcct extends Account{
    private double interestRate;

    public savAct(String customerName,
    String acc_no, double initialBalance,
        double interestRate){

    super(customerName, acc_no, "SAVINGS",
            initialBalance);
    this.interestRate = interestRate;
}

public void computeAndDepositInterest(){

    double interest = balance*(interestRate/100);
    balance += interest;
    S.O.P("interest added:" + interest);
}

Public void withdraw(double amount){
    if(amount <= balance){
        balance -= amount;
        S.O.P("withdrawn:" + amount);
    }else{
        S.O.P("Insufficient balance");
    }
}
}
```

```
class curAcct extends Account{
    private static final double
    MIN_BALANCE = 100.0;
    private static final double
    SERVICE_CHARGE = 50.0;

    public curAcct(String customerName,
    String acc_no, double initialbalance){

    super(customerName, acc_no,
    "Current", initialbalance);
    }

public void withdraw(double amount){
    if(amount <= balance){
        balance = balance - amount;
    S.o.p("Withdrawn:" + amount);

    if(balance < MIN_BALANCE){
    balance -= SERVICE_CHARGE;
    S.o.p("service charge imposed"
            + SERVICE_CHARGE);
    }else
        S.o.p("Insufficient balance");
    }
}
public class Bank{
public static void main(String[]
                        args){
```

```
SavAcct savings = new SavAcct(
        "Alice", "SAV123", 1500.0, 5.0);
savings.displayBalance();

savings.computeAndDepositInterest();
savings.displayBalance();
savings.withdraw(100);
savings.displayBalance();

S.O.P();

CurAcct current = new CurAcct("Bob",
    "CUR456", 600.0);
current.deposit(300);
current.displayBalance();
current.withdraw(700);
current.displayBalance();
current.withdraw(100);
current.displayBalance();
}
}
```

output :-

Deposited : 200.0
Balance : 1200.0
Intrest added : 60.0
Balance : ~~1200.0~~
withdrawn : 300.0

Deposited : 300.0
Balance : 900.0
withdrawn : 700.0
Service charge imposed : 50.0
Balance : 150.0
withdrawn : 100.0
Service charge imposed : 50.0
Balance : 150.0

Read i/p from user

Intrest Interest

LAB program - 06

a) write a java program to create an interface shape with the getarea() method. Create three classes Rectangle, circle & Triangle that implements shape interface. Create objects of the three classes & print the area of the shape.

```
interface Shape {
    double getArea();
}

class Rectangle implements Shape {
    private double length;
    private double width;
    Rectangle(double l, double w) {
        length = l;
        width = w;
    }
    public double getArea() {
        return length * width;
    }
}

class circle implements Shape {
    private double radius;
    circle(double r) {
        radius = r;
    }
    Public double getArea() {
        return Math.PI * radius * radius
    }
}
```

```
class Triangle implements shape{
    private double base;
    private double height;
    Triangle (double b, double h){
        base = b;
        height = h;
    }
    public double getArea(){
        return 0.5 * base * height;
    }
}

public class Main {
    public static void main(String
                            [] args){

        Rectangle r = New Rectangle (5.0, 4.0);
        Circle c = New Circle (3.0);
        Triangle t = New Triangle (6.0, 2.0);

        System.out.println ("Area of Rectangle:" +
                            r.getArea());
        System.out.println ("Area of Circle:" +
                            c.getArea());
        System.out.println ("Area of Triangle:" +
                            t.getArea);
    }
}
```

output

Area of Rectangle : 20.0

Area of circle : 28.274339

Area of Triangle : 6.0.

b) Implements two stacks using array of integers of size six each. Include using the concept of interface check the appropriate conditions for stack overflow & underflow. The other conditions then perform pop() from these two stacks & store it in a new array check for all combinations.

```
interface stacko {
    Void push(int num, int val);
    int pop(int num);
    void display(int num);
}
class TwoStacks implements stacko {
    private final int[] arr =
        new in [10];
    private int top1 = -1;
    private int top2 = arr.length;

    public Void push(int num, int val){
        if(top1+1 == top2){
            System.out.println("stack overflow!");
            return;
        }
```

```
if (num == 1){
    arr[++top1] = val;
}
else if (num == 2){
    arr[--top2] = val;
}
else{
    System.out.println("Invalid Number!");
}
public int pop(int num){
    if (num == 1){
        if (top1 == -1){
            System.out.println("stack1 underflow");
            return -1;
        }
        return arr[top1--];
    }
    else if (num == 2){
        if (top2 == arr.length){
            s.o.p("stack 2 underflow");
            return -1;
        }
        return arr[top2++];
    }
    else{
        s.o.p("Invalid stack number!");
        return -1;
    }
```

```
public void display(int Num){
    if (num == 1){
        if (top1 == -1){
            s.o.p("stack 1 is empty");
            return;
        }
        System.out.print("stack 1 : ");
        for (int i=0; i<=top1; i++){
            system.out.print(arr[i]+" ");
        }
        system.out.println();
    }
    else if (num == 2){
        if (top2 == arr.length){
            s.o.p("stack 2 is empty!");
            return;
        }
        System.out.print("stack : ");
        for (int i=arr.length-1; i>=top2;
                i--);
            system.out.println(arr[i]+" ");
        }
        System.out.println();
    }
    else{
        s.o.p("Invalid stack number!");
    }
}
```

```
public class Main{
    public static void main(String[]
                    args){
    TwoStacks s = new TwoStacks();
        s.push(1,10);
        s.push(1,20);
        s.push(1,30);

        s.display(1);
        s.push(2,40);
        s.push(2,50);
        s.push(2,60);
        s.display(2);

    }
}
```

output:

stack 1: 10 20 30

stack 2: 40 50 60

o/p not seen

Bhatti

LAB - PROGRAM - 07

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "son" which extends the base class. In father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age < 0. In son class, implement a Constructor that uses both father & son's age & Constructor that uses both father & son's age & throws an exception if son's age > father's age.

=>
```
class WrongAge Exception extends
        Exception {
    public WrongAge Exception (String
                message) {
        super (message);
    }
}
class InvalidSonAge Exception
    Extends Exception {
    public InvalidSonAge Exception
            (String message) {
        super (message);
    }
}
```

```
class Father {
    int age;

    public Father(int Age) throws
                    WrongAgeException {
        if(age<0)
            throw new WrongAgeException
            ("Father's age cannot be -ve");
        this.age = age;
    }
}

class Son Extends Father {
    int sonAge;

    public Son(int fage, int sage) throws
    WrongAgeException, InvalidSonAgeException
    {
        super(fage);
        if(sage<0){
            throw new WrongAgeException("
            Son's age cannot be -ve");
        }
        if(sage >= fatherage){
            throw new InvalidSonAg_Exception
            ("Son's age cannot be greater than
            or equal to fathrs age.");
        }
        this.sonAge = sage;
    }
```

```
public class ExceptionDemo{
    public static void main(String
            [] args){
    try{
        Father f = new father (40);
        Son s = new Son (40,20);
        S.o.p("Father's Age:" +
                f.age);

        s.o.p("Son's age:" + s.age);
    }
    catch(WrongAgeException e){
            s.o.p(e);
    }
    try{
        Father f2 = new father(-5);
    }
    catch(WrongAgeException e){
        s.o.p(e);
    }
    try{
        Son s2 = new Son(30,35);
    }catch(InvalidSonAgeException
                e){
        s.o.p(e);
    }
}
```

Mangal
Date
Page

Output :-

Enter your father Age : 45
Enter son's Age = 12
Sons age cannot be less than zero
Would like re enter details (yes/no)
yes
Enter father age : 12
Enter son's age : 23
wrong age

~~Would you like re~~

Seen
19/12/24

Create a package CIE which has two classes - student and internals. The class personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of student. This class has array that stores the SEE Marks scored in five courses of the current sem of the student. import the two packages in a file that declares the final marks of n students in all five courses.

```
package CIE;

public class student {
    protected string usn;
    protected string name;
    protected int sem;

    public student (string usn,
                    string name, int sem){
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
```

```
public String getUsn(){
        return usn;
}
public String getName(){
        return name;
}
public int getSem(){
        return sem;
}
}

package CIE;

public class Internals{
    private int[] internalMarks=
        new int[5];

    public Internals(int[] marks){
        if(marks.length==5){
            this.internalMarks = marks;
        }
        else{
        System.out("Error: Exactly 5 marks
                must be provided:");
        }
    }
    public int[] getInternalMarks(){
        return internalMarks;
```

```
}
public int calculateTotal(){
    int total = 0;
    for(int marks : internalMarks){
        total += marks;
    }
    return total;
}
}

package SEE;

import CIE.Student;
import CIE.Internals;

public class External extends Student{
    private int[] External marks = new
                int[5];

    public External(String usn,
            String name, int sem,
            int[] external marks){
        super(name, usn, sem);
        if(ExternalMarks.length == 5){
            this.External marks = External Marks;
        }else{
            System.out.println("Error: Exactly 5 marks
                must be provided.");
        }
    }
}
```

```
public int[] getExternalMarks(){
    return externalMarks;
}

public int calculateExternalTotal(){
    int total = 0;
    for(int mark : ExternalMarks){
        total += Marks;
    }
    return total;
}

public int calculateFinalMarks(
        Internals Internals){
    int internalTotal = internals.calculate
                        Internal Total();
    int ExternalTotal = External.calculateExternalTotal();

    return internalTotal + ExternalTotal;
}
}
```

```
import CIE.*;
import SEE.*;

import java.util.Scanner;

public class Main{
    p s vm (String [] args){
        Scanner sc = new Scanner(System.in);
        S.o.p("Enter no. of students:");
        int n = sc.nextInt();
        sc.nextLine();

        S.o.p("
        External [] student = new External[n];

        for(int i=0; i<n; i++){
            S.o.p("Enter details for student"
                + (i+1));
            S.o.p("Enter USN:");
            String USN = sc.nextLine();

            S.o.p("Enter Name:");
            String name = sc.nextLine();

            S.o.p("Enter semester:");
            int sem = sc.nextInt();

            int [] internalMarks = new int[5]
```

```
S.o.p("Enter internal marks for 5 courses:");
for(int j=0; j<5; j++){
    internalMarks[j] = sc.nextInt();
}
sc.nextLine();

Internals internals = new Internals(
                        internalMarks);

int[] externalMarks = new int[5];
s.o.p("Enter external marks for 5 courses:");
for(int j=0; j<5; j++){
    externalMarks[j] = sc.nextInt();
}
sc.nextLine();

student[i] = new External(USN, name, sem,
                        externalMarks);
int finalMarks = student[i].CalculateFinal
                    Marks(internals);
s.o.p("final marks for student "+
                    student[i].getName()+"
                    ("+5);
sc.close();
}
}
```

output :

Enter details for student 1
Enter VIN : 1CS21CS001
Enter Name : IC IC
Enter semester : 5
Enter internal marks for 5 courses :
50
60
70
80
90
Enter External marks for 5 courses :
60
70
80
90
100
final marks for student IC (1CS21CS001) 600

Write a program which creates two threads, one thread displaying "BMS COLLEGE OF ENGINEERING" once every ten seconds & another displaying "CSE" one every two seconds.

```
class ThreadBMS extends thread {
    public void RUN() {
        while (true) {
            try {
                S.o.p("BMS College of Engineering");
                Thread.sleep(10000);
            }
            catch (InterruptException e) {
                S.o.p(e);
            }
        }
    }
}
class ThreadCSE extends Thread {
    public void run() {
        while (true) {
            try {
                S.o.p("CSE");
                Thread.sleep(20000);
            }
            catch (InterruptedException e) {
                S.o.p(e);
            }
        }
    }
}
```

```
public class Main {
    p s v m (string [] args) {
        ThreadBMS threadBMS = new ThreadBMS();
        ThreadCSE threadCSE = new ThreadCSE();
        threadBMS.start();
        threadCSE.start();
    }
}
```

output

BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
...