1.

Q. Develop a java program that prints all real solutions to the quadratic equation $ax^2 + bx + c = 0$ Read in a,b,c and use the quadratic formula. If the discriminate $b^2 - 4ac$ is negative, display a message stating that there are no real solutions.

→
```
import java.util.Scanner;
public class Quadratic {
public static void main (String [] args) {

Scanner scanner = new Scanner (System.in);

System.out.println ("Enter coeff of a: ");
double a = scanner.nextDouble();
System.out.println("Enter coeff of b: ");
double b = scanner.nextDouble();
System.out.println("Enter coeff of c: ");
double c = scanner.nextDouble ();
double discriminant = b*b - 4*a*c;

  if (discriminant > 0) {
     double root1 = (-b + Math.sqrt(discriminant)) / (2*a);
     double root2 = (-b - Math.sqrt (discriminant)) / (2*a);
     System.out.println ("The equation has two real solution:"+
                          root1 + " and " + root2);
  } else if (discriminant == 0) {
        double root = -b / (2* a);
        System.out.println ("The equation has one real
                             solution :" + root);
  }else {
     System.out.println ("The equation has no real solutions.");
  }
  scanner.close();
}
```

Output -

* Enter coefficient of a : 2
Enter coefficient of b : -8
Coefficient of C : 3
Real & Distinct Roots
Roots are : 3.58 and 0.42

* Enter coefficient of a : 1
Coff of b : -4
Coff of C : 4
Roots are real and equal
Roots : 2

* Enter coff of a : 1
Coff of b : 1
Coff of C : 1
Equation has no real solution

2.

a Develop a java program to create a class student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

→
```
import java.util.scanner;
class Student {
String USN;
String name;
double [] credits;
double [] marks;
int numsubjects;

STUDENT (int num subjects){
 this.num subject = num subjects;
 credits = new int [num subject];
 marks = new double [num subject];

public void accept Student Details(){
 Scanner sc = new Scanner (System.in);

System.out.println ("Enter your USN");
USN = sc.~~nextline~~ nextLine ();

System.out.println ("Enter your name");
name = sc.nextLine();

for ( int i=0 ; i < num subjects ; i++ ) {
System.out.println (" Enter credits for subject "+(i+1)+":");
 credits [i] = sc.nextDouble();
```

```
System.out.println("Enter marks for subject"+(i+1)+": ");
marks[i] = sc.nextDouble();
}
}
public void displayDetails() {
System.out.println("\n Student details");
System.out.println("USN : "+ USN);
System.out.println("Name : "+Name);
System.out.println("Subjects Details:");
for (int i=0; i < numSubjects; i++) {
System.out.println("Subject "+(i+1)+" : Credits = "+
        credits[i] +"marks = " +marks[i]);
}
}

private double getGradePoints (double marks ){
if (marks >=90 ) return 10;
else if (marks >=80) return 9;
else if (marks >= 70) return 8;
else if (marks >=60) return 7;
else if (marks >=50) return 6;
else if (marks >=40) return 5;
else return 0;
}
```

```
public void calculateSGPA() {
    double totalCredits = 0;
    double totalGrade = 0;

for (int i=0; i<numSubjects; i++) {
double gradePoint = ~~getGp~~ getGradePoint(marks[i]);

totalGradePoints += gradePoint * Credits[i];
totalCredits += Credits[i];
}

double SGPA = totalGradePoints / totalCredits;
System.out.println("SGPA :" + sgpa);

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

Students Student stud = new student(3);

Student acceptDetails();
student.DisplayDetails();
Student.CalculateSGPA();
}
}
```

Output –

Enter USN : Shubhanshu Raj
Enter Name : 1BM23CS321
Enter number of subjects : 3
Credits for ~~subject~~ subject 1 : 2
Marks : 70
Credits for subject 2 : 3
Marks : 80
Credits for subject 3 : 4
Marks 90

USN : 1BM23CS325
Name : Shubhanshu Raj
Subject 1 – ~~6~~ Credits : 2 Marks : 70
Subject 2 – Credits : 3 Marks : 80
Subject 3 – Credits : 4 Marks : 90
SGPA : 8.22

3.

Q Create 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 and get details of the objects. Include a string() method that could display the complete details of the Book. Develop a program to create n book objects.

```
class Book {
    String name, author;
    double price;
    int num_pages;
    Book (String name, string author, float price, int
          num_pages) {
        this.name = name;
        this.author = author;
        the this.price = price;
        this.num_pages = num_pages;
    }
public void setDetails () {
    Scanner sc = new Scanner ( System.in);
    System.out.print ("Enter the name of the book :");
    name = sc.nextLine ();
    System.out.println ("Enter name of the author:");
    author = sc.nextLine();
    System.out.println ("Enter price of the book :");
    price = sc.nextFloat();
    System.out.println (" Enter the number of pages :");
    num_of_pages = sc.nextLine();
```

```
public void getDetails (){
    System.out.println ("Name of Book : "+name);
    System.out.println ("Name of Author : "+ author );
    System.out.println ("Price : "+ price);
    System.out.println ("Number of Pages : " +num_pages);
    System.out.println(

public class Lab {
    public static void main (String [] args){
        Scanner sc = new Scanner (System.in);
        System.out.println (" Enter the number of Books")
        int n = sc.nextInt ();
        Book [] books = new Book [n]

        for ( int i=0 ; i<n ; i++ ){
                books [i]. setDetails ();
        for (int i=0; i<n; i++) {
                books [i].getDetails ();

        }
    }
}
```

Output:-

Enter the number of books :)
Enter the name of book : Charlie and the Chocolate Factory
Name of the Author : Roald Dahl
Enter price of the Book : 450
Enter the number of pages : 200

Name of the Book : Charlie and the Chocolate factory
Name of Author : Roald Dahl
Price : 450
Number of Pages : 200

4.
Q. Develop a
named s
empty m
named Ped
one of
the Cla
that pr

import
Abstract
int
pub

}
}
class
pub

}
void
}

class

void

4.

Q. Develop a java program to create an abstract class named shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and circle such that each one of the class extends class shape. Each one of the classes contain only the method printArea() that prints area of the given shape.

```
import java.util.Scanner;
abstract class shape {
    int dim1, dim2;
    public ~~static void~~ shape (int dim1, dim2) {
        this.dim1 = dim1;
        this.dim2 = dim2;
    }
    abstract double printArea();
}
class rectangle extends shape {
    public Rectangle (dim1, dim2) {
        super (dim1, dim2);
    }
void printArea () {
    double area = dim1 * dim2 ~~*dim3~~;
    System.out.println(area);
}

Class Triangle extends shape {
    public Triangle (dim1, dim2) {
        super (dim1, dim2);
    void printArea {
        double area = 0.5 * dim1 * dim2;
        System.out.println (area);
```

```
class circle extends shape {
    public circle (int dim1)
        super (dim1, dim2);
    }
void Area (){
    double area = Math.PI * dim1 * dim1;
    System.out.println(Area);
}

public class ShapeArea {
    public static void main (String [] args){
    int choice;
    Sc. scanner.nextInt();
    switch (choice){
        case 1: System.out.println("Enter Rectangle of Rect");
                int a = nextInt();
                int b = Sc.nextInt();
                Rectangle x = new Rectangle (a, b)
                x.printArea();
                break;
        case 2: System.out.println("Triangle")
                int base = Sc.nextInt();
                int height = Sc.nextInt();
                Triangle t = new Triangle (ibase, weight)
                t.printArea();
                break;
        case 3: System.out.println("Circle")
                int radius = Sc.nextInt();
                Circle c = new Circle (Radius)
                c.printArea
                break;
    }
```

Output -

Select a shape 1. Rectangle 2. Triangle 3 Circle
Enter length of Rectangle : 12
Enter length of Rectangle : 12
Area : 132

5.

Q Develop a Java program to create a class bank that maintain two kinds of account, one called savings and the other current account. The savings account provide compound interest and withdrawal facilities but no cheque facility. The current account provides facilities cheque book facility but no interest.
Create a class account that stores customer name, acc. number and type of account. From this derive classes cur-acct, saving-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks.

(a) Accept deposit from customer and update the balance
(b) Display the balance
(c) Compute the deposit interest
(d) Permit withdrawl and update balance.

```
import java.util.Scanner;
class Account {
    String customerName, accountNumber;
    double balance;
    String accountType;
    public Account (String CustomerName , String accNo,
                    String AccountType, double balance)
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
}
```

```
public void deposit (double amount) {
  if (amount >0) {
      balance += amount ;
      System.out.println ("Deposit Successful") ; }
      else {
        System.out.println ("Invalid Deposit");

public void displayBalance () {
    System.out.println (" Account Balance : " + balance);
}
public abstract void withdraw (double amount);
}

class current extends Account {
    private static final double Minimum_Balance = 10000;
    private static final double double service_charge = 50;

    public void CurrentAct (String customerName, double
                                                balance)
    super( customerName, accountnumber, " current", balance)
}
public void withdraw (double amount) {
    if (balance-amount >=0 ) {
        balance -= amount ;
        System.out.println ("withdraw successful");
        updateBalance() ; }
    else {
      System.out.println ("Insufficient Balance");
      }
public void updateBalance () {
    if (Balance < Minimum_Balance) {
      Balance -= Service Charge;
}
}
```

```
class SavAct extends Account {
    private static final double Interest_Rate = 0.05;
    public SavAct (String customerName, double Balance) {
        super (customerName, accountNumber, "Savings");
    }

    public void compoundInterest () {
        double interest = balance * interest_rates;
        balance += interest;
        System.out.println ("Interest ~~compound~~ computed");
    }

    public void withdraw () {
        if (balance - amount >= 0) {
            balance -= amount;
            System.out.println ("withdraw successful");
        else {
            System.out.println ("Insufficient balance");

    public void updateBalance () {
        compound interest ();

public class Bank {
    public static void main (String [] args) {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Customer Name:");
        String name = sc.nextLine ();
        System.out.println ("Account number:");
        String accNum = sc.nextLine ();
        Account account;
        if (type.equalsIgnoreCase ("Savings")) {
            account = new SavAct (name, accNum); }
```

REDMI 13C 5G

02/01/2025 09:22

```
} else if (type.equalsIgnoreCase("Current")){
        account = new CurrentAccount();  3
    else {
        System.out.println("Invalid");

Boolean quit = false;
while (!quit) {
    System.out.println("1. Deposit \n
                        2. Withdraw \n
                        3. Display Balance \n
                        4. Update Balance \n
                        5. Quit");
switch (choice) {
case 1: System.out.println("Enter Amount:");
        double depositAmount = sc.nextDouble()
        account.deposit(depositAmount);
        break;
case 2: System.out.println("Withdraw Amt");
        double withdrawAmt = sc.nextDouble();
        account.withdraw(withdrawAmt);
        break;
case 3: account.displayBalance();
        break;
case 4: account.updateBalance();
        break;
case 5: quit = true
        break;
        }
}
        System.out.println("Thank You");
    }
}
```

6.

- Create a class time with variables hrs, mins, secs include methods to accept and display time objects, include a method at time which adds 2 time objects and displays the result in an acceptable time format. (pass time object as a parameter and return the sum of 2 time objects)

```
→ Time add (Time t2) {
  int min1 = (secs + t2.secs) / 60;
  int sec1 = (secs + t2.secs) % 60;
  int hr1 = (mins + t2.mins) / 60;
  int min2 = (mins + t2.mins) % 60;
  int hr2 = (hrs + t2.hrs + hr1);

  Time t3 = new Time();
  t3.hrs = hr2;
  t3.mins = min2;
  t3.secs = sec1;
  return t3;
}

PSVM ()
Time t1 = new Time ()
Time t2 = new Time ()
Time t3 = t1.add (t2);
```

```
Account account;
if (type.equalsIgnoreCase("Savings")) {
    account = new SavAct(name, accNum);
else if (type.equalsIgnoreCase("Current")) {
    account = new ~~Savi~~ CurrAct();
else {
    System.out.println("Invalid");
}
Boolean quit = false;
while (!quit) {
  System.out.println("\1. Deposit \n
                      2. Withdraw \n
                      3. Display Balance \n
                      4. Update Balance \n
                      5. Quit");
switch (choice){
    case 1: System.out.println("Enter Amount:");
            double depositAmount = sc.nextDouble();
            account.dpoint(depositAmount);
            break;
    case 2: System.out.println("Withdraw Amt");
            double withdrawAmt = sc.nextDouble();
            account.withdraw(withdrawAmt);
            break;
    case 3: account.displayBalance();
            break;
    case 4: account.updateBalance();
            break;
    case 5: quit = True;
            break;
}} System.out.println("Thank you");
}
}
```

Output -

Enter name : Shubhendu Raj
Account number : 12234
Account type : Savings
Enter Initial Balance : 100000

1. Deposit 2. Withdraw 3. Display Balance
4. Update Balance 5. Quit
Enter amount to deposit : 10100
Deposit successful. New balance : 1010100.0

1. Deposit 2. Withdraw 3. Display Balance
4. Update Balance 5. Quit
Enter your choice : 4
Interest computed. New balance : 1060500.0

1. Deposit 2. Withdraw 3. Display Balance
4. Update Balance 5. Quit
Enter choice : 5
Thank you for banking with us.

Enter Name : Shubhendu Raj
Account Number : 13545
Account Type : Current
Enter initial balance : 20000

1. Deposit 2. Withdraw 3. Display Balance
4. Update balance
Enter choice : 1
Enter amount to deposit : 10000
Deposited successfully. New Balance : 30,000.0

Enter choice : 4
Interest updated and added.
Enter choice : 3
Balance is 30125.0
Enter choice : 5
Thank you

7.
Q. Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called 'father' and derived class 'son' which extends the base class. In father class implement a constructor which takes the age and throws the exception WrongAge() when input age<0. In son class, implement a constructor that uses both father and son's age and throws an exception if son's age >= father's age.

→
```
class WrongAge extends Exception {
    string message;
    WrongAge (String message) {
        this.message = message;
    }
    public String toString () {
        return "WrongAge Exception :" + message;
    }
}
class Father {
    int fAge;
    Father (int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge ("Father's age cannot
                be negative");
        }
        fAge = age;
    }
}
```

```
class son extends Father {
    int sAge;
    son (int fAge, int sAge) throws WrongAge {
        super(fAge);
        if (sAge < 0) {
            throw new WrongAge("Son's Age is negative");
        }

    if (sAge >= fAge) {
        throw new WrongAge("Son's age can't be greater
                                than Father's Age");
    }
    this.sAge = Age;
    }
}
public class FatherSon {
    public static void main (String []args) {
    try {
        Father father1 = new Father(40);
        son son1 = new son (40,20);
        System.out.println("Father's Age:" +fAge);
        System.out.println("Son's Age:" +sAge);
        Father father2 = new Father(-5);
    }
    catch (WrongAge e) {
        System.out.println(e);
    }
    try {
        son son2 = new son(35,40);
    }
    catch (WrongAge e)
            System.out.println(e);
    }
```

```
try {
    son sons = new son(50, -10);
}
catch (WrongAge e)
        System.out.println(e);
    }
}
}
```

Output:

Father's Age = 40
Son's Age = 20
Wrong Age Exception : Father's age cannot be negative
Wrong Age Exception : Son's Age cannot be greater than Father's Age.
Wrong Age Exception : Son's Age cannot be negative.

8
Q. Write
→ class T
3
3
public

8

Q. Write a program which creates two threads, one thread displaying "Bms college of Engineering" once every ten seconds and another displaying "CSE" once every two seconds.

→
```
class Threads extends Thread {
    String s; int time;
    Threads (String s, int time) {
        this.s = s
        this.time = time;
    }
    public void run();
    try {
        while (true) {
        System.out.println(s);
        Thread.sleep(time * 1000);
        }
    }
    catch (InterruptedException ie){
        System.out.println("ThreadOccurs:" + ie);
    }
}
public class main {
    public static void main (String args[]) {
        Threads t1 = new threads ("Bms College of Engineering",10

        Threads t2 = new threads ("CSE", 2);
        t1.start();
        t2.start();
    }
```

Output -

BMS college of engineering
CSE
CSE
CSE
CSE
CSE
BMS college of engineering