

Prog No.	
Date	

Program - I ($a > 0$) - Roots

using java.util.Scanner; on class) library.java.util

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 discriminant $b^2 - 4ac$ is negative, display a message stating that "there are no real solutions."

```

import java.util.Scanner;
import static java.lang.Math.sqrt;
import static java.lang.Math.abs;
public class squad {
    public static void main (String [] args) {
        Scanner in = new Scanner (System. in);
        System.out.println ("Enter coefficients:");
        int a = in.nextInt();
        int b = in.nextInt();
        int c = in.nextInt();
        if (a==0) {
            System.out.println ("Invalid input");
        }
        else {
            int d = b*b - 4*a*c;
            if (d>0) {
                System.out.println ("Roots are real");
                float r1 = (float) (-b+sqrt (d)) / (2*a);
                float r2 = (float) (-b-sqrt (d)) / (2*a);
                System.out.println (r1);
                System.out.println (r2);
            }
        }
    }
}

```

else if ($d < 0$) {

System.out.println ("Roots are imaginary. There are no real solutions");

float $r_1 = (\text{float}) - b / (2 * a);$

float $r_2 = (\text{float}) \sqrt{(\text{abs}(d))} / (2 * a);$

System.out.println ($r_1 + " + i " + r_2);$

System.out.println ($r_1 + " - i " + r_2);$

}

else {

System.out.println ("Roots are equal");

float $r = (\text{float}) - b / (2 * a);$

System.out.println ($r);$

}

3 branches and switching

Output:
Enter coefficients:

} ($a == 0$) fi

4 : ("Equation bilinear") nothing. two. m12y2;

5

6

} else

Roots are imaginary. There are no real solutions

-0.625 + i 1.0532687 } ($a < b$) fi

-0.625 - i 1.0532687 } (nothing. two. m12y2)

: ($(b^2) / ((b) * r_1 * r_2 + d -) * (two / 4) = 18.75$)

: ($(b^2) / ((b) * r_1 * r_2 + d -) * (two / 4) = 18.75$)

: (r) nothing. two. m12y2

: (r) nothing. two. m12y2

Flowchart:

: matching A

(Start)

S, d, r, b are & b, r, d, s addition variables : Sqrls

2. Initialize variables a,b,c,taking ($a \neq 0$) if ; Sqrls

$$d = a^2 + b^2 - 4ac$$

/ Read a,b,c /

$$a \neq b \neq c : Sqrls$$

"Two or more" thing

$$(a^2 + b^2) - 4ac = 0$$

Print "invalid input"

Sqrls stop (exit) thing

$$a > b \neq c : Sqrls$$

d is less than zero $\Rightarrow d = b^2 - 4ac$ point roots are real

$$\begin{aligned} \text{if } d > 0 \text{ then } \\ x_1 = (-b + \sqrt{d}) / (2 * a) \\ x_2 = (-b - \sqrt{d}) / (2 * a) \end{aligned}$$

Sqrls stop (exit) thing

Print x_1 & x_2 : Sqrls

"Less than zero" thing
 if $d < 0$ then
 Print "Roots are imaginary. There are no real solutions"

$$\begin{aligned} x_1 = -b / (2 * a) \\ x_2 = \sqrt{|d|} / (2 * a) \end{aligned}$$

Print $(x_1 + i x_2)$ Print $(x_1 - i x_2)$

Print "roots are equal"

$$r = -b / (2 * a)$$

Print r

(Stop)

Algorithm:

: truthwif

Step1: Start

Step2: Initialise variable a,b,c,d & read a,b,c

Step3: if ($a=0$) print "Invalid input" goto step 8

Step4: $d = b^2 - 4 * a * c$

Step5: if $d > 0$

print "roots are real"

$$x_1 = (-b + \sqrt{d}) / (2 * a)$$

$$x_2 = (-b - \sqrt{d}) / (2 * a)$$

print (x_1, x_2) goto step 8

Step6: if $d < 0$

print ("Roots are imaginary. There are no real solution")

$$x_1 = -b / (2 * a)$$

$$x_2 = \sqrt{|d|} / (2 * a)$$

print $(x_1 + i x_2)$

print $(x_1 - i x_2)$ goto step 8

Step7: if $d = 0$

print "Roots are equal"

$$x_1 = x_2 = -b / (2 * a)$$

print x_1

Step8: Stop

Q8
6
9/12/23

Program-8) ~~without the list of the student~~ b) or

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

```
import java.util.Scanner;
class Subject
{
    int subjectMarks[] = new int[5];
    int credits;
    int grade;
}
```

```
class Student
{
    Subject subject[];
}
```

```
Subject subject[];
String name;
String USN;
double SGPA;
```

```
Scanner s;
Student()
{
```

```
int i;
subject = new Subject[9];
for (i=0; i<9; i++)
{
```

```
subject[i] = new Subject();
    }
```

```
s = new Scanner(System.in);
    }
```

~~if (grade > 0 & grade < 100) grade = 100;~~

```
void getStudentDetails()
{
```

```
    System.out.print("Enter your name:");
    name = s.next();
```

```
    System.out.println("Enter your USN:");
    USN = s.next();
```

```
}
```

```
void getMarks()
```

```
for (int i=0; i<=8; i++)
```

```
{
```

```
    System.out.print("Enter marks for subject " + (i+1) + ":" );
    Subject[i].subjectMarks = s.nextInt();
```

```
    System.out.print("Enter your credits for subject " + (i+1) + ":" );
    Subject[i].credits = s.nextInt();
```

```
    Subject[i].grade = (Subject[i].subjectMarks/10)+1;
```

```
    if (Subject[i].grade == 11)
        Subject[i].grade = 10;
```

```
    if (Subject[i].grade <= 4)
        Subject[i].grade = 0;
```

```
}
```

```
}
```

```
void computeSGPA()
```

```
{
```

```
    int effectiveScore = 0;
```

```
    int totalCredits = 0;
```

```
    for (int i=0; i<9; i++)
```

```
{
```

```
        effectiveScore += (Subject[i].grade * Subject[i].credits);
```

```
        totalCredits += Subject[i].credits;
```

```
}
```

```
    SGPA = (double) effectiveScore / (double) totalCredits;
```

3

3

class Main

{

public static void main (String args [])

{

student sl = new Student ();

sl.getStudentDetails ();

sl.getMarks ();

sl.computeSGPA ();

System.out.println ("Name: " + sl.name);

System.out.println ("USN: " + sl.USN);

System.out.println ("SGPA :" + sl.SGPA);

}

3

Output:

Enter your USN : 1BN22CS039

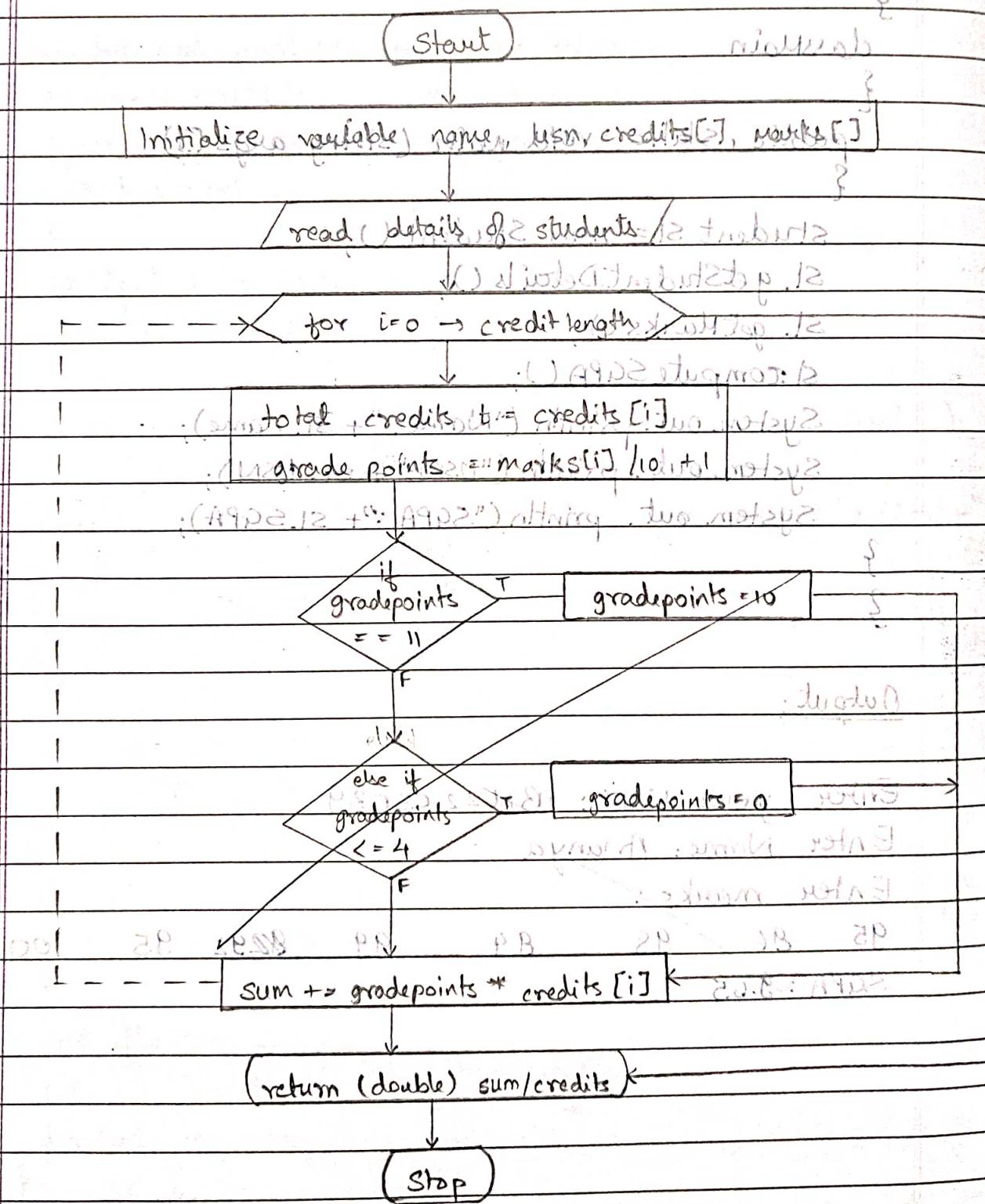
Enter Name: Ananya

Enter marks :

95 86 98 84 99 82 92 95 100

SGPA : 9.65

Flow chart:



Algorithm:

P - program

- Step 1: Start
- Step 2: Initialize variables arr[8], marks, usn, name, sgpa.
- Step 3: Calling class Student in which calling func first()
- Step 4: Input : "Enter usn" + usn from keyboard
- Step 5: Input : "Enter name" + name from keyboard
- Step 6: Print "Enter marks" and a newline character
- Step 7: for (i=0; i<8; i++) {

 arr[i] = in.nextInt(); // taking input

 }
- Step 8: Read array credits as {4,4,3,3,3,1,1,1}
- Step 9: for (i=0; i<arr.length; i++) {

 if (arr[i] >= 100) {

 arr[i] = arr[i]-10; // removing trailing zeros

 }

 } // (i+1)th credit

 else if (arr[i] < 40) {

 arr[i] = 40; // minimum marks

 }

 marks += arr[i]*credit[i]; // calculating total marks

 credit[i] = 1; // incrementing credit

 }
- Step 10: sgpa = marks / 20 // formula: sgpa = marks / total marks
- Step 11: Print "SGPA : " + sgpa
- Step 12: Stop

Print output will be:
 + "SGPA : " + (sgpa) + "\n" + "Total Marks : " + marks + "\n" + "Minimum Marks : " + min + "\n" + "Maximum Marks : " + max + "\n" + "Average Marks : " + avg + "\n" + "Percentage : " + percentage + "\n"

Program- 9

Q Create a class Book which contains four members: name, author, price, num-pages. Include a constructor to set initial values of for the members. Include 8 methods to set and get the details of the objects. Include also a static method that could "display" complete details of the book. Develop a java program to create 5 book objects.

```

import java.util.Scanner;
class Books {
    String name;
    String author;
    int price;
    int num_pages;
    public void set(int i) {
        Scanner in = new Scanner(System.in);
        System.out.println("Enter details of book " + (i+1));
        name = in.nextLine();
        author = in.nextLine();
        price = in.nextInt();
        num_pages = in.nextInt();
    }
    public String toString() {
        return "Details of Book " + (i+1) + "\n" +
            "Name: " + name + "\n" +
            "Author: " + author + "\n" +
            "Price= " + price + "\n" +
            "No. of pages= " + num_pages;
    }
}

```

```

class Main {
    public static void main (String [] args) {
        int n;
        Scanner in = new Scanner (System.in);
        System.out.println ("Enter number of books: ")
        n = in.nextInt();
        Books b[] = new Books[n];
        for (int i=0, i<n; i++) {
            b[i] = new Books();
            b[i].set(i);
        }
        System.out.println ();
        for (int i=0, i<n, i++) {
            System.out.println (b[i].toString ());
        }
    }
}

```

Output:

Enter the no. of books: 1

Enter name:

abc

Enter author:

xyz

Enter price:

500

Enter number of pages

450

Details of all books

Book 1:

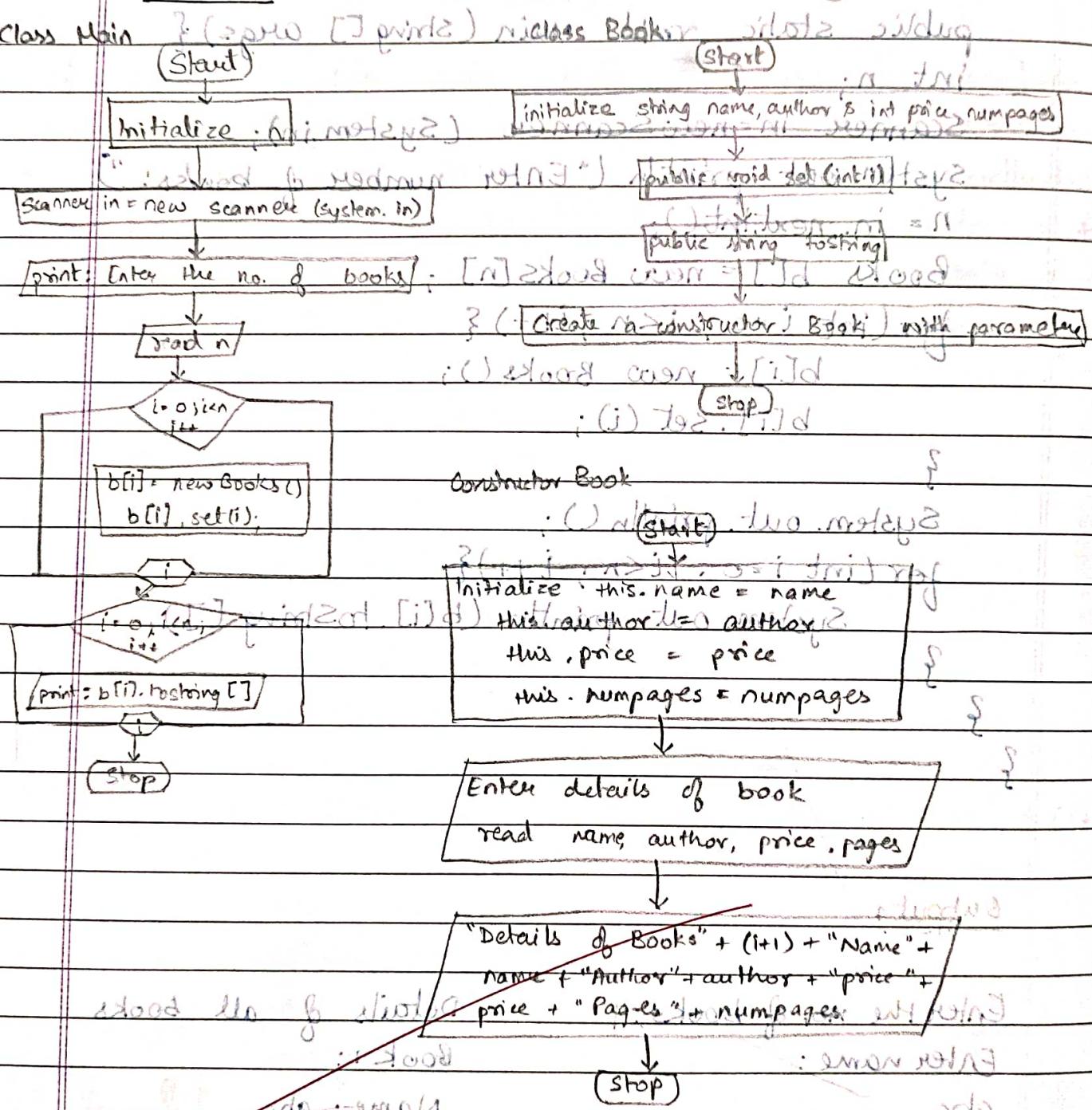
Name: abc

Author: xyz

Price: 500

Num of pages: 450

Flowchart:



Algorithm:

Q1 - program

Step 1: Start ~~process~~ the program until a go back

Step 2: Initialize variables, name, author, price, num_pages

Step 3: Enter no. of books using ~~scanf~~ function

Step 4: Enter name, author, price, num_pages

Step 5: for ($i=0$; $i < n$; $i++$)
 do {
 b[i] = new_books();
 b[i] = set[i];
}

Step 6: display books
 for ($i=0$; $i < n$; $i++$)
 point book details

Step 7: stop

~~(1) zero book details~~

~~(2) zero price) menu book details~~

~~(3) zero menu = right option~~

~~(4) zero right~~

~~(5) zero left option = right option~~

~~(6) zero left~~

~~(7) zero right option = right option~~

~~(8) zero right~~

~~repeat charted option until~~

~~3) why?~~

~~(n). message menu ==> menu?~~

~~(1) zero left option add nothing to menu?~~

~~{; x = 0~~

Program-10

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

```

import java.util.Scanner;
abstract class Shape {
    int x, y;
    abstract void area();
    public static void main (String args[])
    {
        Shape obj1 = new Circle ();
        obj1.area();
        Shape obj2 = new Rectangle ();
        obj2.area();
        Shape obj3 = new Triangle ();
        obj3.area();
    }
}

class Circle extends Shape {
    Circle ()
    {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter the radius of the circle");
        x = sc.nextInt ();
        y = x;
    }
}

```

void area()

{

System.out.println("area of a circle is "+3.14*x*y);

}

}

class Rectangle extends Shape{

Rectangle(){}

Scanner sc = new Scanner (System.in);

System.out.println("enter the length and breadth of
the rectangle");

x= sc.nextInt();

y= sc.nextInt();

void area()

{

System.out.println("area of rectangle is "+ x*y);

}

class Triangle extends Shape{

Triangle(){}

Scanner sc = new Scanner (System.in);

System.out.println("enter the base and height of the
triangle");

x= sc.nextInt();

y= sc.nextInt();

void area()

{

System.out.println("area of triangle is "+ 0.5*x*y);

}

}

Output:

(ans) bin

: Enter the "radius" of the circle, type two,metre²

2

area of circle is 12.56

Enter the length & breadth of the rectangle, type

5

; (m.metre²) remain 0.00 = 22 remain 2

6 area of rectangle is 25 m², type two,metre²

Enter the base & height of the triangle

5

2

area of triangle is 5

(ans) bin

; (x*x + "ai spoint fo zero") nothing, type

12.012a

second character spoint and

{("spoint

; (m.metre²) remain 0.00 = 22 remain 2

11 of triangle bin and all nothing, nothing, type two,metre²

; ("spoint

; (1) nothing, 0.00 = x

{; (1) nothing, 0.00 = x

nothing

(ans) bin

; (x*x *2.0 + "ai spoint fo zero") nothing, type two,metre²

Algorithm

- Step 1: Create abstract class named shape
Step 2: Include 2 members x & y
Step 3: Declare abstract method area();
Step 4: Create sub-class Rectangle that extends shape
Step 5: Create & override area method to calculate area of rectangle.
Step 6: Repeat steps 4 and 5 for triangle and circle.
Step 7: In main method create object rectangle, triangle and circle.
Step 8: Stop

~~Step 8~~

~~Stop~~

Flow chart:

Circle

(start)

Scanner sc = new Scanner (System.in)

[Enter the radius of circle] / base obj : 1 qrtz

abstract void
 read x,y

["area of circle is " + 3.14*x*y]

(stop)

bns direct ref & bns + 2 qrtz

Triangle

(start)

Scanner sc = new Scanner (System.in)

[Enter the base & height of triangle]

read x,y

["area of triangle is " + 0.5*x*y]

(stop)

Rectangle

(start)

Scanner sc = new Scanner (System.in)

[Enter length & breadth of rectangle]

abstract void
 read x,y

["area of rectangle is " + x*y]

(stop)

bns direct ref & bns + 2 qrtz

odd shape

abstract void prior()

Initialize x,y/2 : 2 qrtz

abstract void area()

shape obj1 = new Circle()

obj1.area();

shape obj2 = new Rectangle()

obj2.area();

shape obj3 = new Triangle()

obj3.area();

(stop)

Program-12

Q Develop a Java prog to create a 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 and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a min. balance & if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account no. & type of account.

From this derive the classes cur-acct and sav-acct to make them more specific to their requirements. Include the necessary methods in order to make them achieve the following tasks:

- a) Accept deposit from customer and update the balance
- b) Display the balance
- c) Compute and deposit interest
- d) Permit withdrawal & update the balance. Check for min balance, impose of penalty if necessary & update the balance

import java.util.Scanner;

class Account {

 String customerName;

 long accno;

 String accountType;

 double balance;

 public Account (String customerName, long accno, String accountType) {

```
this.customerName = customerName; // setting
```

```
this.accno = accno; // withdrawal is allowed
```

```
this.accountType = accountType; // withdrawal
```

from withdraw this.balance = 0.0; // setting the balance

```
{ withdrawal = "22 : withdraw" }
```

```
public void displayBalance () {
```

```
System.out.println ("Account Number: " + accno);
```

```
System.out.println ("Customer Name: " + customerName);
```

```
System.out.println ("Account Type: " + accountType);
```

```
System.out.println ("Balance: $" + balance);
```

```
}
```

{ (not withdraw) print } throw setting

```
; ("service", min, withdrawal) type
```

```
class CurAcct extends Account {
```

```
double minBalance;
```

```
double serviceCharge;
```

```
public CurAcct (String customerName, long accno) {
```

```
super (customerName, accno, "Current");
```

```
this.minBalance = 500.0; // sets min balance
```

throws this.serviceCharge = 50.0; // sets service charge

```
}
```

```
public void withdraw (double amount) {
```

```
if (balance - amount >= minBalance) {
```

```
balance -= amount; // throw setting
```

```
System.out.println ("withdrawal successful. Current
```

balance = " + balance); // throw setting

```
else { // throw setting
```

balance); throw System.out.println ("Insufficient funds. Withdrawal

```
is not allowed."); // throw setting
```

```

public void imposeServiceCharge() {
    if (balance < minBalance) {
        balance += serviceCharge;
        System.out.println("Service charge imposed. Current
                           Balance: Rs." + balance);
    }
}

```

```

;(long) + " : account number") + " interest. Two methods
;(minBalance + " : min balance") + " interest. Two methods
;(double + interestRate;) + " interest. Two methods

```

```

public SavAcct (String customerName, long accno) {
    super(customerName, accno, "Savings");
    this.interestRate = 0.05;
}

```

```

;(long) public void depositInterest() {
    double interest = balance * interestRate;
    balance += interest;
    System.out.println("Interest deposited. Current Balance:
                      $" + balance);
}

```

```

public void compoundInterest(double initialAmount,
                             double interestRate, int term) {
    double compoundInterest = initialAmount * Math.pow(
        (1+interestRate), term) - initialAmount;
    balance += compoundInterest;
    System.out.println("Compound Interest deposited
                      current Balance is Rs." + balance);
}

```

```

public class Bank {
    public static void main (String [] args) {
        Scanner scanner = new Scanner (System.in);
        System.out.println ("Choose account type:");
        System.out.println ("1. Current");
        System.out.println ("2. Savings");
        System.out.println ("Enter choice (1 or 2):");
        int choice = scanner.nextInt();
        if (choice == 1) {
            CurAcct curAccount = new CurAcct (customerName,
                accno);
            System.out.print ("Enter initial balance: $");
            double initialBalance = scanner.nextDouble();
            curAccount.balance = initialBalance;
            System.out.print ("Enter withdrawal amount: $");
            double withdrawalAmount = scanner.nextDouble();
            curAccount.withdraw (withdrawalAmount);
            curAccount.imposeServiceCharge ();
            curAccount.displayBalance();
        } else if (choice == 2) {
            SavAcct savAccount = new SavAcct (customerName,
                accno);
            System.out.print ("Enter initial balance: $");
            double initialBalance = scanner.nextDouble();
            savAccount.balance = initialBalance;
        }
    }
}

```

```

System.out.print("Enter withdrawal amount: $");
double withdrawalAmount = scanner.nextDouble();
if (withdrawalAmount < 0) {
    System.out.println("Withdrawal amount cannot be negative");
} else {
    savAccount.balance -= withdrawalAmount;
    System.out.println("Withdrawal successful. Current
        Balance: $" + savAccount.balance);
}

System.out.print("Enter interest rate:");
double interestRate = scanner.nextDouble();
if (interestRate < 0) {
    System.out.println("Interest rate cannot be negative");
} else {
    savAccount.interestRate = interestRate;
    savAccount.displayBalance();
}

System.out.print("Enter term (in years) for
    compound interest calculation:");
int term = scanner.nextInt();
if (term < 0) {
    System.out.println("Term cannot be negative");
} else {
    savAccount.compoundInterest(term);
    savAccount.displayBalance();
}

if (choice == 1) {
    System.out.println("Enter initial balance for current account");
    double initialBalance = scanner.nextDouble();
    if (initialBalance < 0) {
        System.out.println("Initial balance cannot be negative");
    } else {
        savAccount.currentBalance = initialBalance;
        System.out.println("Current account created successfully");
    }
} else if (choice == 2) {
    System.out.println("Enter initial balance for savings account");
    double initialBalance = scanner.nextDouble();
    if (initialBalance < 0) {
        System.out.println("Initial balance cannot be negative");
    } else {
        savAccount.savingsBalance = initialBalance;
        System.out.println("Savings account created successfully");
    }
}

Output: java Account
Choose account type: 1. Current 2. Savings
1. Current
Enter choice (1 or 2): 1
Enter customer name: Ananya
Enter account number: 2004
Enter initial balance: $5000
Enter withdrawal amount: $500
Withdrawal successful. Current Balance: $4500.0

```

Account Number : 2004

Customer Name : Ananya

Account Type : current

Balance : ₹ 4500.0

Algorithm:

Step1: Create a class Account

Step2: Initialise variables accountno, accno, accounttype, balance.

Step3: Input: Enter customer name, accno, balance, account type as saving or current account from the user.

Step4: Enter choice of savings & current account.

Step5: Read the choice & if choice is saving step 6, for current step 7.

Step6: Enter initial balance, withdraw amount & min balance from user. Check condition for min balance i.e initial balance withdrawal should be \geq min i.e 800
 then new Balance = balance - withdrawal;

Then, print current balance, enter interest rate & time to calculate CI.

$$CI = \text{balance} * \text{Power} (1 + \text{interest rate}, \text{time}) - \text{initial balance}$$

Step7: else if choice = 2

Create object of saving account.

Step8 : Stop

Program-13

; BEE operating

; InstabD.312 taught

Q Create a package CIE which has 2 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 5 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 an array that stores the SEE marks scored in 5 courses of the current semester of the student. Import the 2 packages in a file that declares the final marks of n students in all 5 courses.

(T) Java program: main body static void

package CIE;

import java.util.*;

public class Student {

{

 public int sem;

 public String usn;

 public String name;

 public void accept ()

 {

 Scanner scan = new Scanner (System.in);

 System.out.println ("Enter U, N, S");

 usn = scan.nextLine ();

 name = scan.nextLine ();

 sem = scan.nextInt ();

 }

 public void show ()

 {

 System.out.println ("U : " + usn + " N : " + name + " S : " + sem);

 }

}

public class Internals {

 public int im[] = new int [5];

 public void accept ()

 {

```

package SEE;
import CIE.Student;
final public class External extends Student {
    public int[] marks = new int[5];
    public void accept() {
        System.out.println("Enter marks of sub");
        for (int i=0; i<5; i++) {
            marks[i] = sc.nextInt();
        }
    }
    public void print() {
        System.out.println("Total marks = " + sum);
    }
}

```

$fm[j] = s[i].im[j] + st[i].sm[j];$

}

System.out.println("Final marks of "+st[i].name);
 for (int k=0; k<5; k++) {
 }

System.out.println("course "+(k+1)+" = "+fm[k]);

Algorithm:

Step 1 : Create package CIE. Declare a public class Student with members USN, name, sem.

Step 2 : A method input allows user to give inputs for variables USN, name & sem.

Step 3 : Create another class Internals. In same package CIE. It consists of an array int[] that stores internal marks, scored in 5 courses.

Step 4 : Create package SEE. Import student class from which extends student class.

Step 5 : Import package CIE & SEE, create array m that stores total marks. Read user input.

Step 6 : Add internal & external marks.

Step 7 : Display total marks of the students

Step 8 : Stop.

Output: $[p]_{ms} \cdot [i]_{fs} + [p]_{ml} \cdot [i]_s = [i]_{ml}$

(www.[REDACTED]IBH22CS029, L0N7F1) affirms the above.

Enter no. of students \Rightarrow (1 to 100) \Rightarrow (1 to 100)

Enter details 1

~~(Ex) Entert (sent) USN 8 "Nanticoe" due. May 22.~~

Enter internal and SEE marks of Sub 1 : 45 45

Enter internal and SEE marks of sub : 48 46

~~Enter internal and SEE marks of sub3 : 47~~

~~Enter internal and SEE marks of sub 4: 48~~

Final marks of IBN22G5039

Course 1: 90

Course 2 : 92

~~sub course 3 : 94b31 E15 speaking stage : 1992~~

Course 4 n=96 median 0.000 "threshold"

stupri sp. *coronatus* scutell. *tuoni* bantam A.: 8 opt.

~~med~~ ~~✓~~ when (not) subdivisions not

~~Mass W. demotri sub montana stage: Sept~~

retire from the state of Florida after he had served as

2. *bermudiana*: Observe longish, upright, sword-like leaves.

Week 10: 10/12/2018 - 10/18/2018 Total: \$972

such turbid channels which are

Wiederholung der 3D- und 4D-Spektroskopie

...togni genn brad, abura lofai amata tuu

Strong Lepidoptera & Lomelinia blitoides

strebst du so etwas jetzt weiter? raus

$$\underline{g_{12} = g_{11}}$$

Program #11 abstract and sub

Q) Write a program that demonstrates inheritance handling of exceptions in inheritance tree. Create a base class called "father" and derived class called "son" which extends the base class. In父 father class, implement a constructor which takes the age & throws the exception wrongAge() when the input age < 0. In Son class, implement a constructor that takes both father's son's age & throws an exception if son's age is \geq father's age.

```

import java.util.Scanner;
class wrongAge extends Exception {
    public wrongAge(String message) {
        super(message);
    }
}

class Father {
    int fatherAge;
    public Father(int fatherAge) throws wrongAge {
        if (fatherAge < 0) {
            throw new wrongAge("Age cannot be negative");
        }
        this.fatherAge = fatherAge;
    }
}

```

class Son extends Father { (part)

int sonAge;

public Son (int fatherAge, int sonAge) throws wrongAge
throws Super (fatherAge); but it is not tested at
about if (sonAge >= fatherAge) { why doesn't this
ability throw new wrongAge ("son's age must be less
than father's age"); } so it is not
relevant in this program, as both will be > 0 in this case

this.sonAge = sonAge; but - suffix error. Test
{ } so it is not tested in this case

public class FatherSon { this is the main program

public static void main (String [] args) { solution

Scanner sc = new

Scanner (System.in);

System.out.println ("Enter father's age & son's age:");

int fa = sc.nextInt();

int sa = sc.nextInt();

try { except from

PathError s = new Son (fa, sa); what sibling

System.out.println ("Father's age: " + s.fatherAge);

System.out.println ("Son's age: " + s.sonAge); { } but it is not tested in this case

catch (wrongAge e) { { }

System.out.println ("Error: " + e.getMessage()); { }

{ { }

{ { }

Output:

piyush@9

Enter father's age and son's age: 50 20

50 is greater than 20

20 is less than 50

Father's age > 50

Son's age : 20

Parent abides A seeks

Algorithm:

Step 1: Create an ~~exception~~ exception ~~wrong age~~ which extends exception ~~wrong age~~

Step 2: Create base class father

Step 3: Make a constructor ~~father (age)~~ which inputs age of father

Step 4: If Age < 0; throw exception ~~wrong age~~

Step 5: Create son class derived from father

Step 6: Take both "father" & "son's age" in constructor.

Step 7: If son's age >= father's age, throw exception ~~(Wrong Age.)~~

Step 8: Stop ~~nothing else~~

Program - 15

- Q) Write a program which creates two threads, one thread displaying "BMS College Of Engineering" once every ten seconds & another displaying "CSE" once every two sec.

class A extends Thread

3

```
int t1=0, time;
```

ACJS

Since CH_3COOH has a higher concentration than NaCl , it will dissociate more, leading to a higher H^+ concentration.

time = 21000; noItgaze3 abroba

3. ~~the~~ ~~sent~~ ~~the~~ ~~web~~ ~~and~~ ~~shows~~ : soft

studiò storia public void run() throws IOException {super

~~rest of the sun~~

soft errors while ($t1 \leq time$) ; o > 90% / : 149/2

With many thanks to the author.

```
System.out.println("BMS College")  
try {  
    // code  
} catch (Exception e) {  
    // code  
}
```

~~agent sleep(10000); f < 300 sleep 11 : 1972~~

catch (Exception e) {

System.out.println("error");

3. *Constitutive* *Regulation* *of* *Gene* *Expression*

$$t_1 + = 10\,000;$$

1970-1971 *1971-1972* *1972-1973* *1973-1974* *1974-1975* *1975-1976*

...and the last time I saw him he was sitting in a chair, holding a cigarette, looking very weary.

[A faint, illegible signature or mark is visible above the line.]

...and the last time I saw him he was sitting in a chair, holding a cigarette, looking very weary.

class B extends Thread {

int t2=0, time;

B() {

time = 21000;

t2 = 2000; }

public void run() {

while (t2 <= time) {

System.out.println ("CSE");

try {

sleep (2000); }

catch (Exception e) {

System.out.println ("error"); }

t2 += 2000;

}

}

class th

public static void main (String args[])

A a = new A();

B b = new B();

a.start();

b.start();

↳ output : 1972

↳ output : 1972

4

↳ output : 1972

↳ output : 1972

↳ output : 1972

Output:

{ boomit charles & sebs
unit,0-1st flr}

BMS College Of Engineering

CSE

CSE

CSE

CSE

CSE

; ("32") althing, tba . not,y,z

BMS College Of Engineering

CSE

CSE

{(" mitpxz") althing, tba . not,y,z

CSE

CSE

Log of other

(1) log pride) min, charles, idots, ilduq

Algorithm:

Step 1 : start

Step 2 : Create a class A that extends to thread

Step 3 : Initialise fr=0, time = 3000 in constructor of A,

Step 4 : write a method run.

Step 5 : Use a while loop which runs until the condition
 $t <= \text{time}$ becomes false.

Step 6: Print BMS college of Engineering.

Include toy block & use sleep method for number of times 10000. In catch block, print other statement ability less than 0 error if it encounters exception.

Step 7: Create a class B that extends to thread.

Initialise t2 = 0, time = 3000 in constructor of B

Step 8: Create a method run() in current thread.

Step 9: Use a while loop which runs until the condition $t2 \leq time$ becomes false. Print CSE.

Step 10: Include toy block & call sleep method for 2008. In catch block, print the statement error if it encounters exception.

Increment $t2 = t2 + 2000$; sleep

Step 11: Create a class demo.

Step 12: Initialise object a of ? datatype A, b of datatype B.

Step 13: Call method (start) of object A.

Step 14: Call method (start) of object B.

(1) b.start(); a.start();

(2) a.start(); b.start();

(3) b.start(); a.start();

("start()", a.start(); b.start());

(1) b.start(); a.start();

(2) b.start(); a.start();

(3) b.start(); a.start();

(4) b.start(); a.start();