VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Harbakhshish Singh Arora (1BM23CS104)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019

Sep-2024 to Jan-2025

B.M.S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Object Oriented Java Programming (23CS3PCOOJ)" carried out by **StudentName (1BM23CS000),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Prof. Swathi Sridharan Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

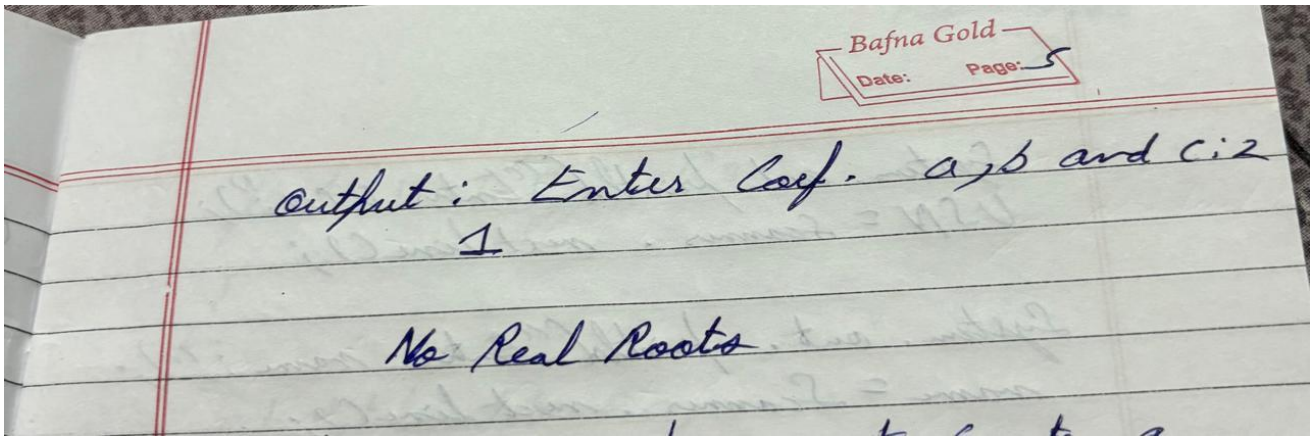
Github Link:

Program 1
Java program to find real solution in quadratic equations.

Algorithm:

(3) Write a Java program to find real solutions in quadratic eqn.

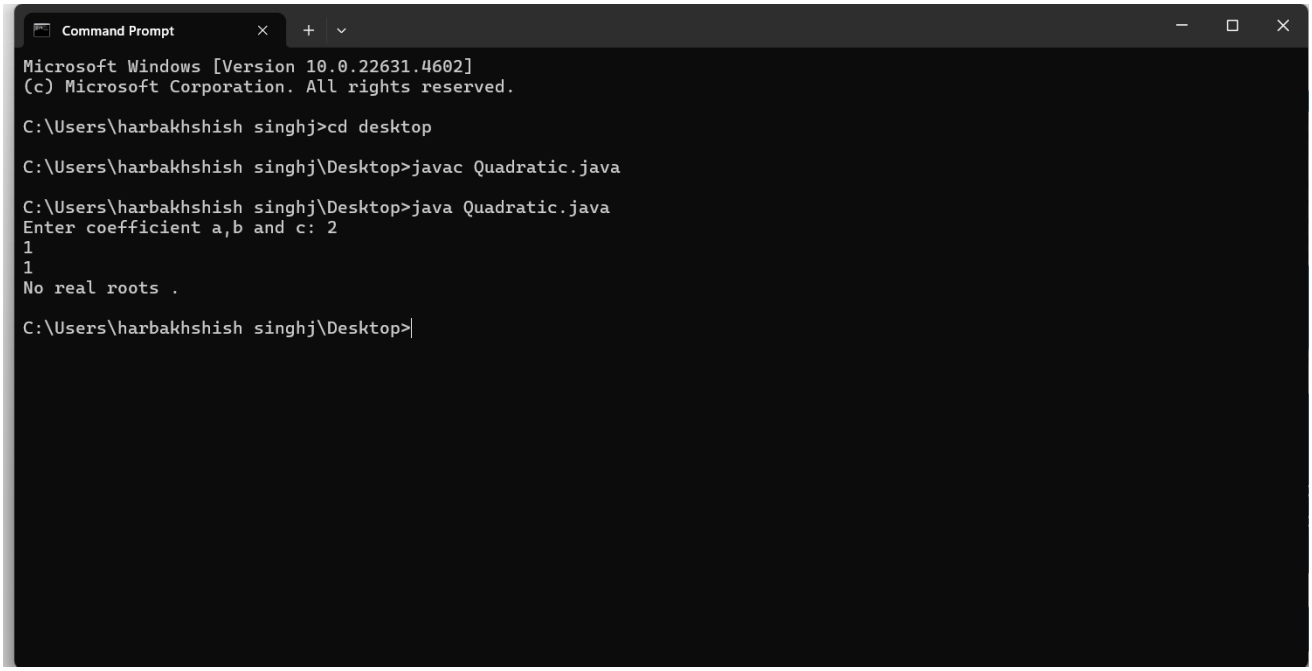
```
import java.util.*;
public class quadratic
{
  public static void main(String[] args)
  Scanner sc = new Scanner(System.in);
  system.out.print("Enter Coeff a, b and c: ");
  double a = sc.nextDouble();
  double b = sc.nextDouble();
  double c = sc.nextDouble();
  double discriminant = b*b - 4*a*c;
  if (discriminant > 0)
  {
  double root1 = (-b + maths.sqrt
                  (discriminant))/(2*a);
  } else {
      system.out.println
  double root2 = (-b - maths.sqrt
                  (discriminant))/(2*a);
  system.out.println("Roots are:" + root1
                  + "and" + root2);
  } else if (discriminant == 0)
  {
      double root = -b/(2*a);
      system.out.println("Root is:" + root);
  } else {
      system.out.println("No real roots
  }
  sc.close();
}
```



Code:

```
import java.util.Scanner;

public class Quadratic {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter coefficient a,b and c: ");
      double a = scanner.nextDouble();
      double b = scanner.nextDouble();
      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 roots are real and distinct: " + root1 + " and " + root2);
      } else if (discriminant == 0) {
         double root = -b / (2 * a);
         System.out.println("The root is real and repeated: " + root);
      } else {
         System.out.println("No real roots .");
      }
      scanner.close();
   }
}
```

Output-



Program 2-
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

Algorithm-

Output: Enter coef. a,b and c:2
1

No Real Roots

Q→ Develope 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 Stu {

private string usn;
private string name;
private int[] credit;
private int[] marks;

Public Stu (int num of subjects)

Credits = new int[num of students]
marks = new int[num of students]

Public Void accept Details()
{
    Scanner scanner = new Scanner (System.
    USN = Scanner.next line();
    System.out.print ("Enter name: ");
```

```
System.out.println("Enter USN:");
USN = Scanner.nextLine();

System.out.println("Enter name:");
name = Scanner.nextLine();

for(int i=0; i<Credits.length; i++)
{
system.out.println("enter credit for subject"
        +(i+1)+":");

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

~~System.out.println("Enter the~~

```
public void display()
{
System.out.println("USN"+""+USN);
System.out.println("Name "+" "+name);
```

~~System.out.println("Credit"+Credit);~~

```
for(i=0; i<num of subjects; i++)
system.out.println("Credits"+Credits[i]);
system.out.println("Marks"+Marks[i]);

}
```

```
public void Sgpa ()
{
int total credits =0;
int total grade = 0;

for (i=0; i< num of subjects, i++)

total credits += credits[i];
total grade += (marks[i]/10) * credits[i];
return ( total grade / total credits );
}

public static void main ()

{
Scanner input = new Scanner (system.in);
System. out. println ("Enter total number
                       of subjects");
num of subjects = input. next int ()
Student S1 = new Student (num of subjects);
Student S2 = new Student (num of subjects);
S1. accept ()
S1. display ()
```

output:

Enter USN: 1bm23cs104
Enter Nam: harbakshish
Enter no. of Subjects: 2
Enter marks for Subject 1: 89
Enter Credits for S1: 3
Enter Credits S2: 3
Enter marks for S2: 98
SGPA: 9.333333334

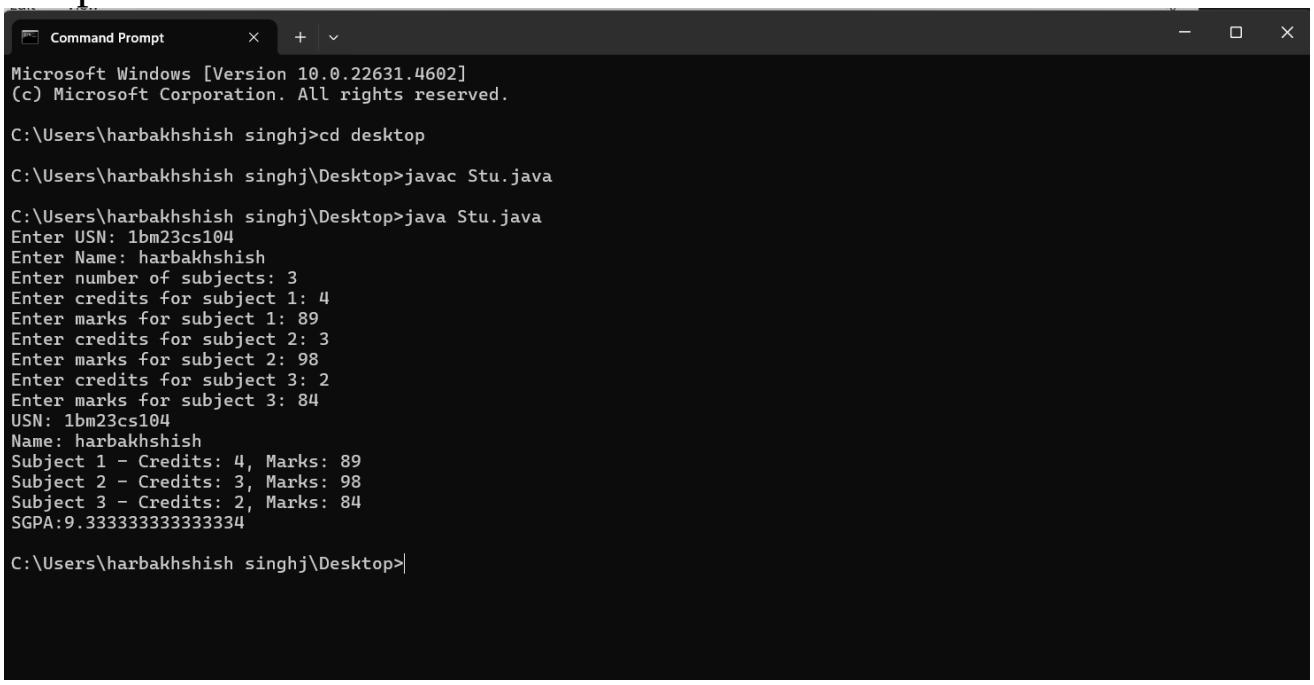
8/10

Code-

```
import java.util.Scanner;

class Stu {
    String usn;
    String name;
    int[] credits;
    int[] marks;

    void acceptDetails() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter Name: ");
        name = scanner.nextLine();
        System.out.print("Enter number of subjects: ");
        int n = scanner.nextInt();
        credits = new int[n];
        marks = new int[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = scanner.nextInt();
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }

    void displayDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + " - Credits: " +
```

```
credits[i] + ", Marks: " + marks[i]);
        }
    }

    double calculateSGPA() {
        int totalCredits = 0;
        double totalGradePoints = 0;
        for (int i = 0; i < credits.length; i++) {
            int gradePoint = getGradePoint(marks[i]);
            totalGradePoints += gradePoint * credits[i];
            totalCredits += credits[i];
        }
        return totalGradePoints / totalCredits;
    }

    int getGradePoint(int marks) {
        if (marks >= 90) return 10;
        if (marks >= 80) return 9;
        if (marks >= 70) return 8;
        if (marks >= 60) return 7;
        if (marks >= 50) return 6;
        if (marks >= 40) return 5;
        return 0;
    }

    public static void main(String[] args) {
        Stu student = new Stu();
        student.acceptDetails();
        student.displayDetails();
        System.out.println("SGPA:"+ student.calculateSGPA());
    }
}
```

Output-



Program 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 and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.

Algorithm-

Q→ Create a class Book which Contains four members: name, author, price, num_pages. Include methods to set and Constructor. Include toString() method that displays Complete details of book. Develop a Java Program to create n book objects.

```
import java.util.Scanner;

class Book {

private string name;
private string author;
private double price;
private int num_pages;

public Book (String name, String author,
             double, int num_pages)
{
  this.name = name;
  this.author = author;
  this.price = price;
  this.num_pages = num_pages;

public String getName() {
        return Name;
}

public String getauthor() {

        return author;
}
```

```
public String getPrice() {
        return price;
}

public String getNum_pages() {
                return Num_pages;
}

public String toString()

{
   return "Book Name:" + name + "\n Author:"
      + author + "\n Price: $" + price +
   "\n Num_pages" + num_page;
}
}
public class BookStore {

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);
System.out.println("Enter no. of books:");
int n = scanner.nextInt();
        scanner.nextLine();
     Book[] books = new Book[n];
for(int i=0; i<n; i++)
{
  system.out.println("Enter Book details",
                                + (i+1));
system.out.println("Enter Book name:"
String name = scanner.nextLine();
```

```
System.out.println("Enter the Author name:");
String author = scanner.nextLine();
system.out.println("Enter the book price");
String price = scanner.nextLine();
system.out.println("Enter the no of pages");
String num_pages = scanner.nextLine();

book[i] = new Book[name, author, price,
                   num_pages];
}
system.out.println("\n--Book details--");
for(Book book : books)
{
System.out.println(book.toString());
System.out.println("-------");
}

scanner.close();
}
}
```

Q →

Output:

Enter number of books : 1
Enter Book Name : C Program
Enter author Name : Sam
Enter price : 200
Enter no. of pages: 300

Book details:

Book 1 Name = C Program, Author = Sam,
price = 200.0, numpages = 300

22.10

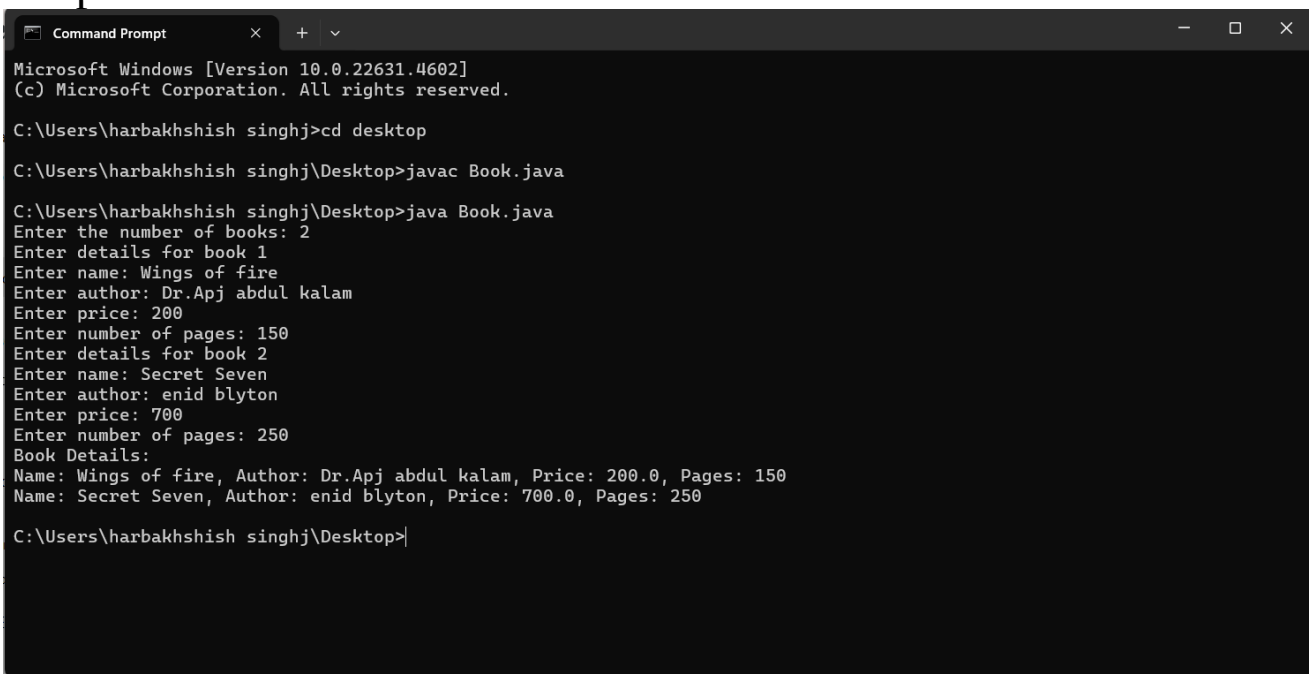
Code-

```
import java.util.Scanner;

class Book {
    private String name;
    private String author;
    private double price;
    private int num_pages;

    Book(String name, String author, double price, int num_pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }

    void setName(String name) {
        this.name = name;
    }

    String getName() {
        return name;
    }

    void setAuthor(String author) {
        this.author = author;
    }

    String getAuthor() {
        return author;
    }

    void setPrice(double price) {
```

```
        this.price = price;
    }

    double getPrice() {
        return price;
    }

    void setNumPages(int num_pages) {
        this.num_pages = num_pages;
    }

    int getNumPages() {
        return num_pages;
    }

    public String toString() {
        return "Name: " + name + ", Author: " + author + ", Price: " +
price + ", Pages: " + num_pages;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of books: ");
        int n = scanner.nextInt();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for book " + (i + 1));
            scanner.nextLine();
            System.out.print("Enter name: ");
            String name = scanner.nextLine();
            System.out.print("Enter author: ");
            String author = scanner.nextLine();
            System.out.print("Enter price: ");
```

```
            double price = scanner.nextDouble();
            System.out.print("Enter number of pages: ");
            int num_pages = scanner.nextInt();
            books[i] = new Book(name, author, price, num_pages);
        }
        System.out.println("Book Details:");
        for (Book book : books) {
            System.out.println(book);
        }
    }
}
```

Output-



Program 4- 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 classes extends the class Shape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.

Algorithm-

```
abstract class Shape {
int dim1, dim2

Shape (int dim1, int dim2)
{
  this.dim1 = dim1
  this.dim2 = dim2
}
  abstract void printArea()
}
Class Rectangle extends Shape {

Rectangle (int length, int breadth)
{
  super (length, breadth);
}
  Void printArea()
{
  int area = dim1 * dim2;
System.out.println("Area of Rectangle" + area
}
}

class Triangle extends Shape {

Triangle (int base, int height)
{
  super (Base, height);
}
```

(a) override

```
void printArea() {
double area = 0.5 * dim1 * dim2;
System.out.println("Area of Triangle" + area);
}
}
class Circle extends Shape {
    Circle(int radius) {
        super(radius, 0);
    }
@Override
void printArea() {
    double area = Math.PI * dim1 * dim2
System.out.println("Area of Circle:" + area);
}
}
public class Main {
    public static void main(String[] args)
{
    Shape rectangle = new Rectangle(10, 5);
    Shape Triangle = new Triangle(10, 5);
    Shape circle = new Circle(7);

rectangle.printArea();
triangle.printArea();
circle.printArea();
}
}
```

Output:

Area of Rectangle: 50
Area of triangle: 25.0
Area of Circle: 153.93804

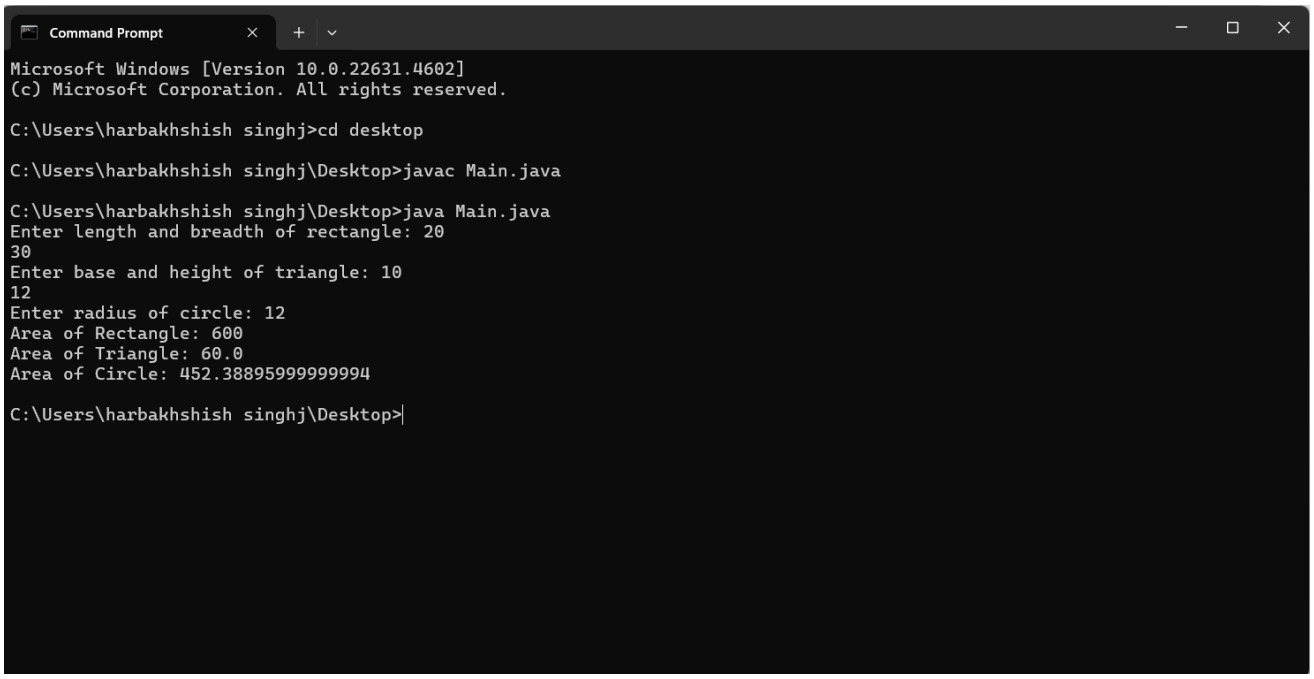
22/10

Code-

```
import java.util.Scanner;

abstract class Shape {
    int a, b;
    abstract void printArea();
}

class Rectangle extends Shape {
    Rectangle(int length, int breadth) {
        a = length;
        b = breadth;
    }
    void printArea() {
        System.out.println("Area of Rectangle: " + (a * b));
    }
}

class Triangle extends Shape {
    Triangle(int base, int height) {
        a = base;
        b = height;
    }
    void printArea() {
        System.out.println("Area of Triangle: " + (0.5 * a * b));
    }
}

class Circle extends Shape {
    Circle(int radius) {
        a = radius;
    }
```

```
    void printArea() {
        System.out.println("Area of Circle: " + (3.14159 * a * a));
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter length and breadth of rectangle: ");
        Rectangle rectangle = new Rectangle(scanner.nextInt(),
scanner.nextInt());
        System.out.print("Enter base and height of triangle: ");
        Triangle triangle = new Triangle(scanner.nextInt(),
scanner.nextInt());
        System.out.print("Enter radius of circle: ");
        Circle circle = new Circle(scanner.nextInt());
        rectangle.printArea();
        triangle.printArea();
        circle.printArea();
    }
}
```

Output-



Program 5- Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and 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 minimum balance and if the balance falls below this level, a service charge is imposed.
Create a class Account that stores customer name, account number and 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 achieve the following tasks:

Algorithm-

Bafna Gold
Date: Page: 22

Q→ Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called saving account and other current account. The saving 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 minimum balance and if the balance falls below this level, service charge is imposed.
Create a class account that stores customer na account number and type of account. Fro this device the class Cur-acct and Sav-ac to make them more specific to their requirements include the necessary methods in order to achieve the following task.

1) Accept deposit from customer and update bala
2) Display the balance.
3) Compute and deposit interest.
4) Permit withdrawal & update balance.
Check for minimum balance impost pen if necessary and update the balan

```
import java.util.Scanner;
class Account {
String CustomerName;
int accountNumber;
String accountType;
double balance;

public Account (String name, int number,
                String type, double balance)
{   this.CustomerName = name;
    this.accountnumber = number;
    this.accounttype = type;
    this.balance = balance;

public void deposit (double amount)

{   balance += amount;
    system.out.println ("Balance:" + balance
}
}
class cur-acct extends Account
{
    double minimumBalance;
    double servicecharge;

public cur-acct (String name, int num
        double balance, double minBalance,
            double servicecharge)
super (name, number, type: "Current",
    this.minimumBalance = minBalance,
```

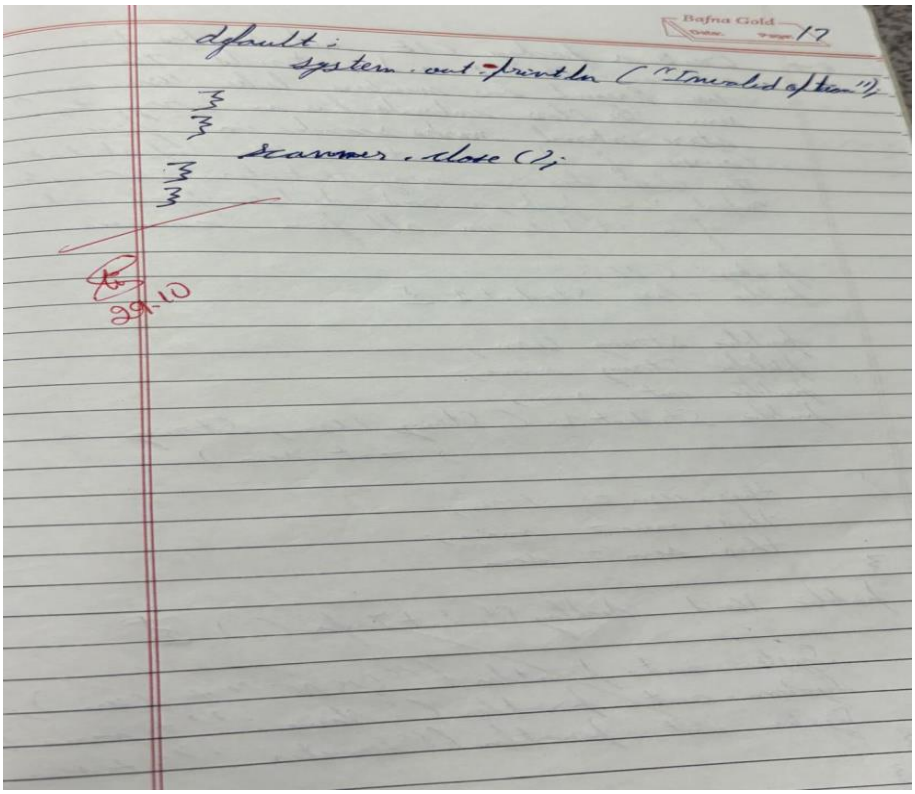
```
        this.serviceCharge = serviceCharge;
    }
    public void displayBalance() {
        System.out.println("Balance:" + balance);
    }
}
class Cur_acct extends Account {
    double minimumBalance;
    double serviceCharge;

public Cur_acct(String name, int number,
        double balance, double minBalance,
        double serviceCharge) {

    super(name, number, type: "current",
            balance);

    this.minimumBalance = minBalance;
    this.serviceCharge = serviceCharge;
}

public void withdraw(double amount)
{
    if (balance - amount < minBalance)
    {
        balance -= serviceCharge;
System.out.println("Service charge imposed
            due to insufficient balance" + serviceCharge);
    else
```

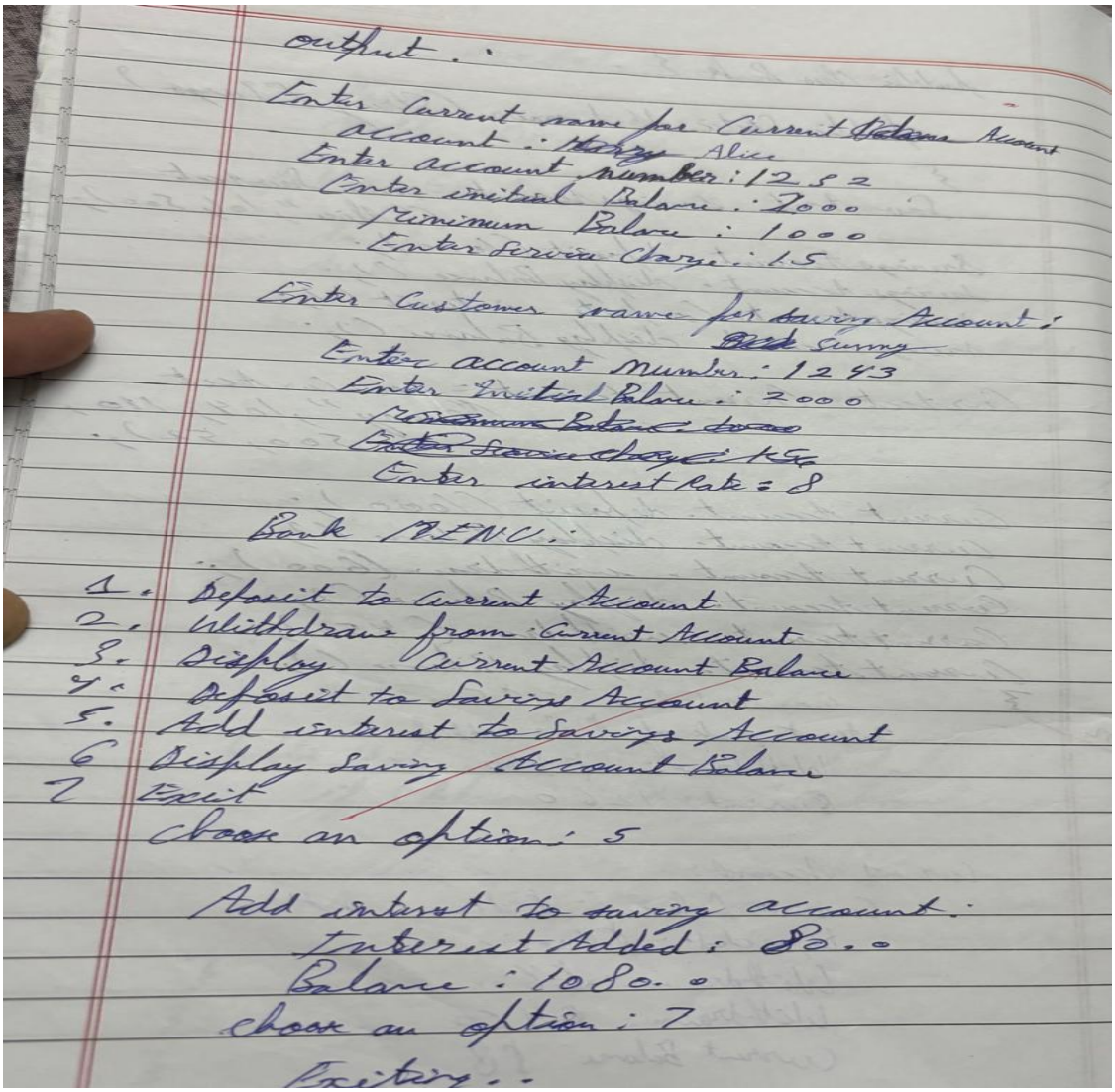
```
    } balance -= amount;
      System.out.println("Withdrawn: " + amount);
    }
  }
}

class SavAcct (String name, int number
               , double balance, double interestRate)
  super(name, number, type "Savings", balance);
    this.interestRate = interestRate;
  }
    public void depositInterest() {
      double interest = balance * interestrate/100;
      balance += interest;
      System.out.println("Interest added" + interest);
    }
  }
}
public class Bank {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in
    System.out.println("Enter Customer Name for Curren
                       account");
    String name = scanner.nextLine();
    System.out.println("Enter account num
    int number = scanner.nextLine();
```

```
System.out.println("Enter initial balance");
double initialBalance = scanner.nextDouble();
System.out.println("Enter minimum
                    Balance");
double minBalance = scanner.nextDouble();
System.out.println("Enter service charge");
double serviceCharge = scanner.nextDouble();

Cur_Acct CurrentAccount =
new Cur_Acct(name, number, initial
             Balance, minBalance, serviceCharge);

System.out.println("\n Enter Customer
                    name for Saving account");
scanner.nextLine();
System.out.println("Enter account
                    number:");
number = Scanner.nextInt();
System.out.print("Enter initial balance");
initialbalance = Scanner.nextDouble();
System.out.println("Enter interest rate");
double InterestRate = scanner.nextDouble();

Sav_Acct
Sav_Acct SavingAccount = new Sav_Acct
        (name, number, initial balance, interestrate);

boolean exit = false;

while(exit){
```

```
System.out.println("\n Bank Menu");
System.out.println("Deposit CT");
System.out.println("Withdraw CT");
System.out.println("Display CT");
System.out.println("Deposit SA");
System.out.println("Add interest");
System.out.println("Display SA");
System.out.println("Exit");
System.out.println("Choose option");
    int choice = scanner.nextint();

switch (choice)

{

Case 1:
    System.out.println("Enter amount deposit to CT

    double depositAmount = Scanner.nextDouble();
    CurrentAccount.deposit(depositAmount);
CurrentAccount.displayBalance();
break;

Case 2:
    System.out.println("Enter amount to
                withdraw from current Accou

    double withdrawAmount = scanner.nextDou
CurrentAccount.withdraw(withdraAmount);
CurrentAccount.displayBalance();
break;
```

```
Case 3: &
    System.out.println("Current Account Balance");
    CurrentAccount.displayBalance();
    break;

Case 4:

system.out.println("Enter amount to be
                    deposit to saving account");

    depositamount = Scanner.nextDouble();
    SavingAccount.deposit(depositamount);
    SavingAccount.displayBalance();
    break;

Case 5:
    system.out.println("Adding Interest to saving
                        Account");

    SavingAccount.depositinterest();
    SavingAccount.displayBalance();
      break;

Case 6:
    system.out.println("Saving Account Bala

    SavingAccount.displayBalance();
    break;

Case 7:

        exit = true;
    System.out.println("Exiting
    break;
```


Bafna Gold
17
default:
System.out.println("Invalid option");
scanner.close();
29.10



Code-

```
Import java.util.scanner;
class Account {
    String customerName;
    String accountNumber;
    String accountType;

    Account(String customerName, String accountNumber, String accountType) {
        this.customerName = customerName;
```

```
        this.accountNumber = accountNumber;
        this.accountType = accountType;
    }

    void displayDetails() {
        System.out.println("Customer Name: " + customerName);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Type: " + accountType);
    }
}

class SavAcct extends Account {
    double balance;

    SavAcct(String customerName, String accountNumber, double
balance) {
        super(customerName, accountNumber, "Savings");
        this.balance = balance;
    }

    void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount + ", New Balance:
" + balance);
    }

    void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient balance");
        } else {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ", Remaining
Balance: " + balance);
```

```
        }
    }

    void computeInterest(double rate, int time) {
        double interest = balance * Math.pow(1 + rate / 100, time) -
balance;
        balance += interest;
        System.out.println("Interest: " + interest + ", Updated Balance:
" + balance);
    }
}

class CurAcct extends Account {
    double balance;
    double minimumBalance;
    double serviceCharge;

    CurAcct(String customerName, String accountNumber, double
balance, double minimumBalance, double serviceCharge) {
        super(customerName, accountNumber, "Current");
        this.balance = balance;
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }

    void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount + ", New Balance:
" + balance);
    }

    void withdraw(double amount) {
        if (amount > balance) {
```

```
            System.out.println("Insufficient balance");
        } else {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ", Remaining
Balance: " + balance);
            checkMinimumBalance();
        }
    }

    void checkMinimumBalance() {
        if (balance < minimumBalance) {
            balance -= serviceCharge;
            System.out.println("Balance below minimum. Service
charge imposed. New Balance: " + balance);
        }
    }

    void chequeBookFacility() {
        System.out.println("Cheque book facility is available.");
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter Customer Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter Account Number: ");
        String accNumber = scanner.nextLine();
        System.out.print("Enter Account Type (Savings/Current): ");
        String accType = scanner.nextLine();

        switch (accType.toLowerCase()) {
```

```
            case "savings":
                System.out.print("Enter Initial Balance: ");
                double savingsBalance = scanner.nextDouble();
                SavAcct savingsAccount = new SavAcct(name,
accNumber, savingsBalance);
                savingsAccount.displayDetails();
                savingsAccount.deposit(1000);
                savingsAccount.withdraw(500);
                savingsAccount.computeInterest(5, 1);
                break;

            case "current":
                System.out.print("Enter Initial Balance: ");
                double currentBalance = scanner.nextDouble();
                CurAcct currentAccount = new CurAcct(name,
accNumber, currentBalance, 1000, 50);
                currentAccount.displayDetails();
                currentAccount.chequeBookFacility();
                currentAccount.deposit(1000);
                currentAccount.withdraw(1500);
                currentAccount.withdraw(600);
                break;

            default:
                System.out.println("Invalid Account Type");
}
}
}
```
Output-

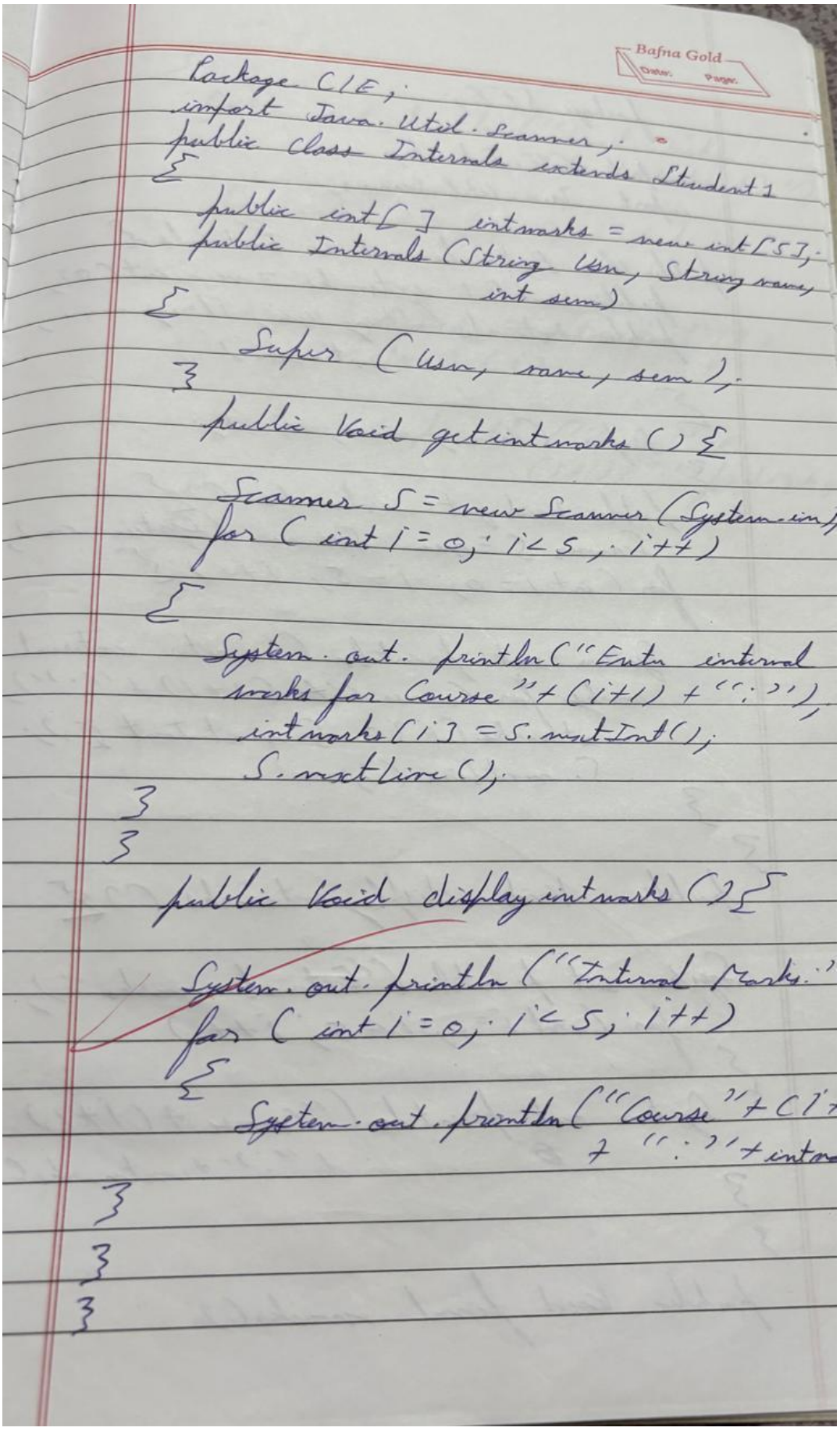
Program 6-

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 an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

Algorithm-

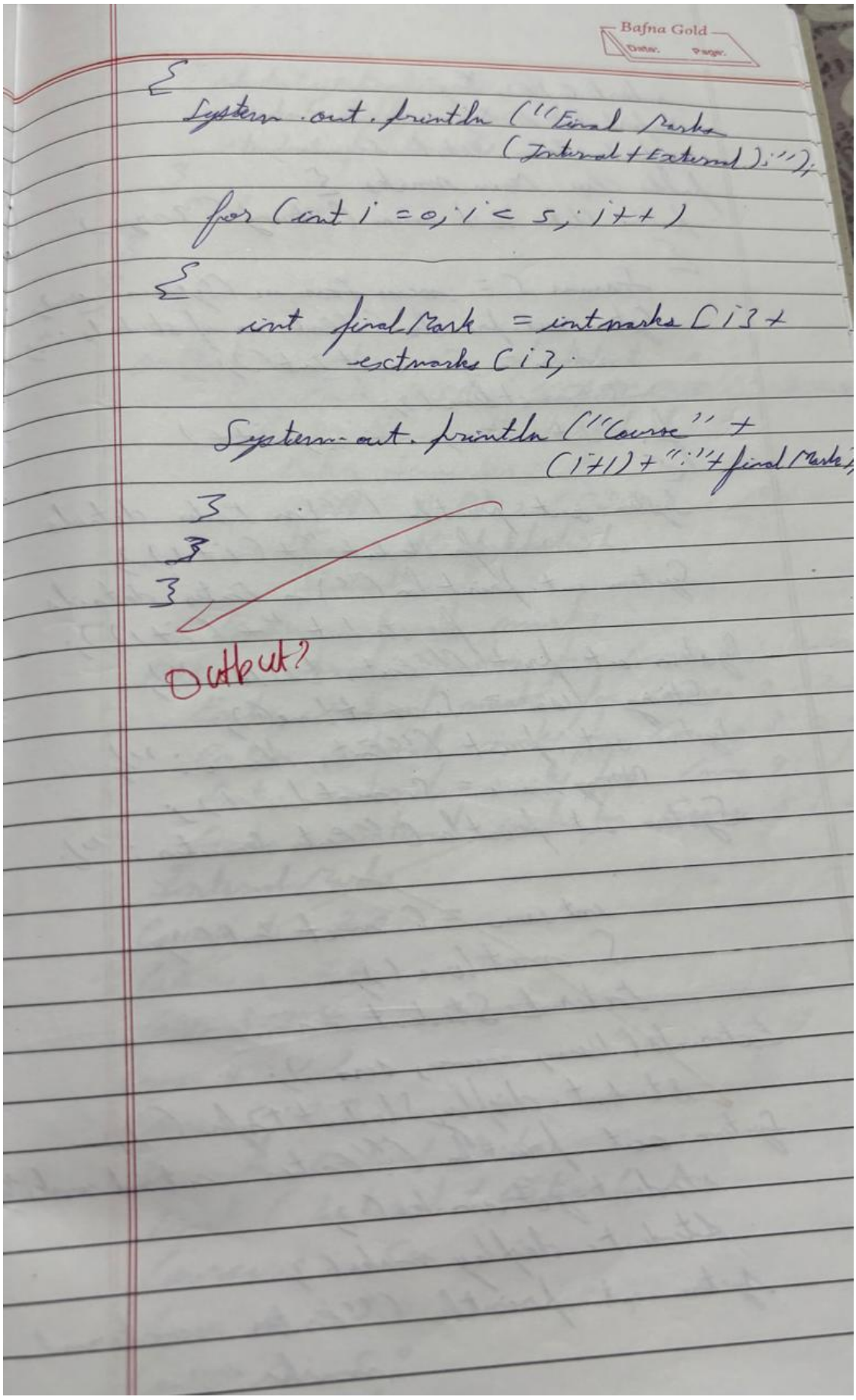
9 → Create a package CIE which has two classes - Student and internals. The class Personal has members like usn, name, sem. The class internals has an array to store internal marks scored in five courses of the current semester of student. This class has an array that stores SEE marks. Import two packages and declare final marks of n students in all five courses.

```
package CIE;
public class Student1 {

public String usn;
public String name;
public int sem;
public Student1 (String usn, String name
                 , int sem)

{   this.usn = usn;
    this.name = name;
    this.sem = sem;
}
public Void displayStudentInfo () {

System.out.println ("USN:" + usn);
System.out.println ("Name:" + name);
System.out.println ("Semester:" + sem);
}
}}
```



```
package CIE;
import java.util.Scanner;
public class Internals extends Student {
{
    public int[] intmarks = new int[5];
    public Internals (String usn, String name,
                      int sem)
{
    super (usn, name, sem);
}
    public void getintmarks () {
    Scanner S = new Scanner (System.in);
    for (int i=0; i<5; i++)
{
    System.out.println("Enter internal
    marks for course" + (i+1) + ":");
    intmarks[i] = S.nextInt();
    S.nextLine();
}
}
}
    public void displayintmarks () {
    System.out.println("Internal Marks:"
    for (int i=0; i<5; i++)
    {
        System.out.println("Course" + (i+
                + ":" + intm
}
}
}
```

```
package SEE;

import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
    public int[] extmarks = new int[5];
    public Externals(String usn, String name,
                     int sem)

    {
        super(usn, name, sem);
    }
    public void get_extmarks(){
        Scanner S = new Scanner(System.in);
        for(int i=0; i<5; i++){

            System.out.println("Enter external
            marks for Course:"+(i+1)+":");
            extmarks[i] = S.nextInt();
            S.nextLine();
        }
    }
    public void display_extmarks(){

    System.out.println("External marks:");
        for(int i=0; i<5; i++)
    {
    System.out.println("Course"+(i+1)
                    +":"+extmarks[
    }
    }
    public void final_marks()
```


Bafna Gold
{
System.out.println("Final Marks (Internal+External):");
for(int i=0;i<5;i++)
{
int finalMark = intmarks[i]+extmarks[i];
System.out.println("Course"+(i+1)+":"+finalMark);
}
}
}
Output?

```
import CIE.*;
import SEE.*;
import java.util.Scanner;
public class MainMarks {
  public static void main(String[] args)
  {
    Scanner S = new Scanner(System.in);
    System.out.println("Enter no of students:");
      int n = S.nextInt();
      S.nextLine();
  for(int i=0; i<n; i++)
  {
  System.out.println("\n Enter details
                for student "+(i+1));
  System.out.println("\n Enter details
                for student "+(i+1));
System.out.print("Enter USN:");
  String usn = S.nextLine();
System.out.print("Enter Name:");
    String name = S.nextLine();
System.out.println("Enter Semester:");

      int sem = S.nextInt();
      S.nextLine();
      External Student = new
External(usn, name, sem);
    Student.displayStudentInfo();
System.out.println("Enter internal marks");
    student.getmarks();
    Student.displaymarks();
  System.out.println("Enter external
                marks");
```

```
        student.getExtmarks();
        student.displayextmarks();
        student.finalmarks();
    }
    s.close();
}
}
```

output: Enter no. of students: 1
Enter USN: ~~XXXXXX~~1BM23CS104
Enter Name: Harbakhsish
Enter Semester: 3
USN: 1BM23CS104
Name: Harbakhsish
Semstr: 3
Enter internal marks:
Enter internal marks for Course 1: 20
Enter internal marks for Course 2: 22
Enter internal marks for Course 3: 24
Enter internal marks for Cours 4: 18
Enter internal ~~marks~~ for Course 5: 20
Enternal Marks:
Course 1: 20
Course 2: 22
Course 3: 24
Course 4: 18
Course 5: 20
Same for Exctenal Marks
final (Internal + External):
Course 1: 50
Course 2: 50
Course 3: 59
Course 4: 58
Course 5: 58

Code-

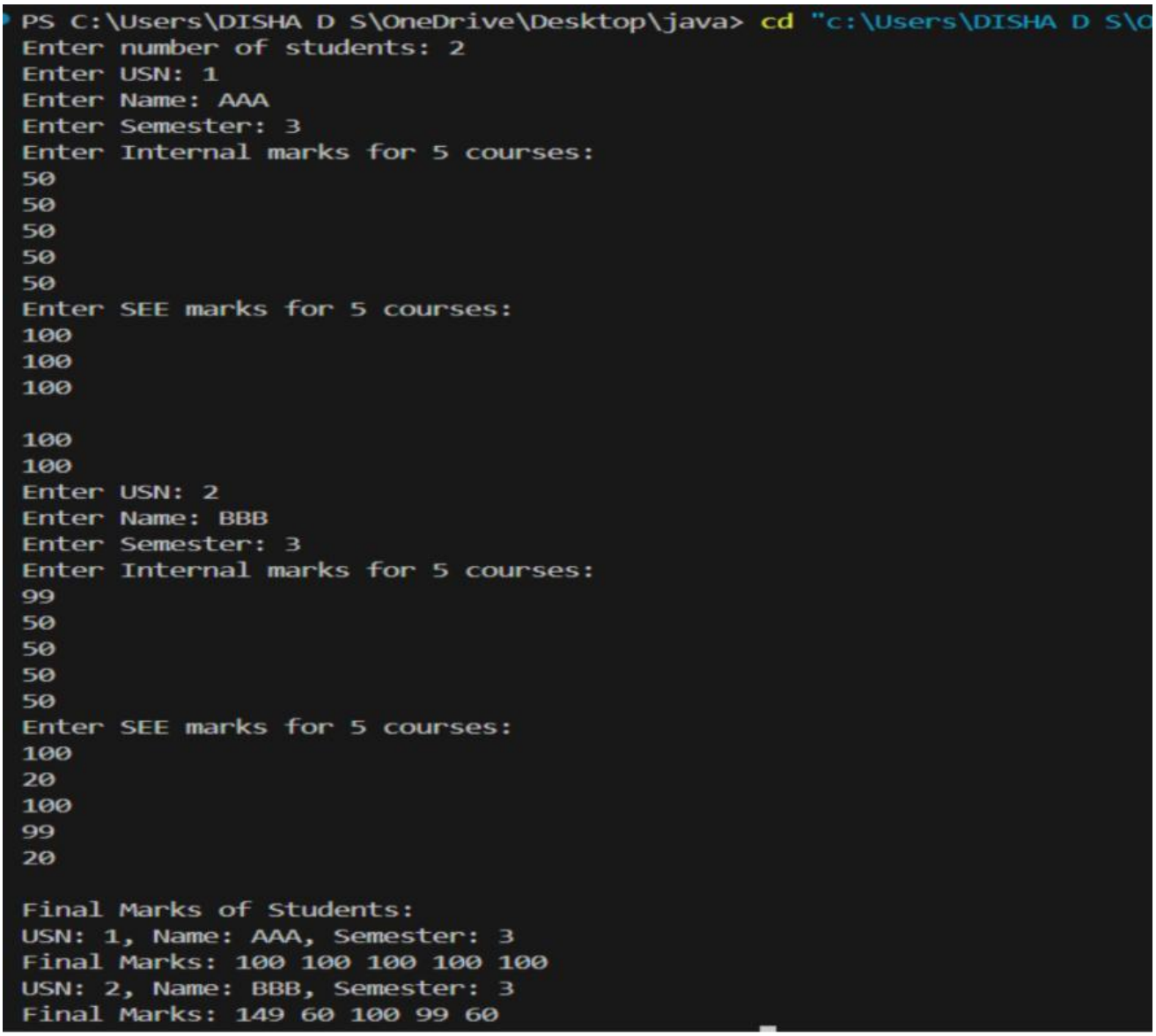
```
package CIE;
public class Student1 {
    public String usn;
    public String name;
    public int sem;
    public Student1(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
    public void displayStudentInfo() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}}
package CIE;
import java.util.Scanner;
public class Internals extends Student1 {
    public int[] intmarks = new int[5];
    public Internals(String usn, String name, int sem) {
        super(usn, name, sem);
    }
    public void getintmarks() {
        Scanner s = new Scanner(System.in);
        for (int i = 0; i < 5; i++) {
            System.out.println("Enter internal marks for course " + (i +
1) + ":");
            intmarks[i] = s.nextInt();
            s.nextLine();
        }
    }
```

```
    public void displayintmarks() {
        System.out.println("Internal Marks:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + intmarks[i]);
        }
    }
}

package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals {
    public int[] extmarks = new int[5];
    public Externals(String usn, String name, int sem) {
        super(usn, name, sem);
    }
    public void getextmarks() {
        Scanner s = new Scanner(System.in);
        for (int i = 0; i < 5; i++) {
            System.out.println("Enter external marks for course " + (i +
1) + ":");
            extmarks[i] = s.nextInt();
            s.nextLine();
        }
    }
    public void displayextmarks() {
        System.out.println("External Marks:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + extmarks[i]);
        }
    }
    public void finalmarks() {
        System.out.println("Final Marks (Internal + External):");
```

```
        for (int i = 0; i < 5; i++) {
            int finalMark = intmarks[i] + extmarks[i];
            System.out.println("Course " + (i + 1) + ": " + finalMark);
        }
    }
}
}
import CIE.*;
import SEE.*;
import java.util.Scanner;
public class Mainmarks {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter number of students:");
        int n = s.nextInt();
        s.nextLine();

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for student " + (i + 1));
            System.out.print("Enter USN: ");
            String usn = s.nextLine();
            System.out.print("Enter Name: ");
            String name = s.nextLine();
            System.out.print("Enter Semester: ");
            int sem = s.nextInt();
            s.nextLine();

           Externals student = new Externals(usn, name, sem);


            student.displayStudentInfo();
```

```
            System.out.println("Enter internal marks:");
            student.getintmarks();
            student.displayintmarks();


            System.out.println("Enter external marks:");
            student.getextmarks();
            student.displayextmarks();


            student.finalmarks();
        }

        s.close();
    }
}
```

Output-



Program 7-write 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 and son's age and throws an exception if son's age is >=father's age.

Algorithm-

Bafna Gold
Date: Page:

Q→ Write a Program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called 'Son' which extends 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 and son's age and throws an exception if son's age >= father's age.

```
import Java.Util.*;
Class Main{
 public static void main(String args[]){

Scanner scan = new
Scanner(System.in);
System.out.println("enter the son's
    and father's details");
Son s = new
Son(scan.next(), scan.nextInt(),
 scan.next(), scan.nextInt());
}
}
Class Father{
String name; int age;
Father(String name, int age ){
 this.name = name;
try{
 if(age >0){
  this.age = age;
```

```
System.out.println("father age is valid");
} else {
throw new WrongAge(age);
}
Catch (WrongAge e) {

System.out.println(e);
System.out.println("negative fathers age");
}
}
}

Class Son extends Father {
Son (String name, int age, String fathername,
        int fatherage) {
super (fathername, fatherage);

try {
if (age >=0 && age < fatherage) {

this.age = age;
System.out.println(e);
System.out.println(age < 0 ? "negative error":
("Sons age cannot be more than fathers");
}
}
}

Class WrongAge extends
Exception {
int age;
WrongAge (int age) {
this.age = age;
}
```

Bafna Gold

```
public String toString ()
return "invalid age" + age;
}
}
```

Output :

enter son's and father's details

father age is Valid
invalid age 45
Sons age can not be more than fathers
~~enter the son's and father's details :~~

if input was son's age 25 and
father's age 40

then output :

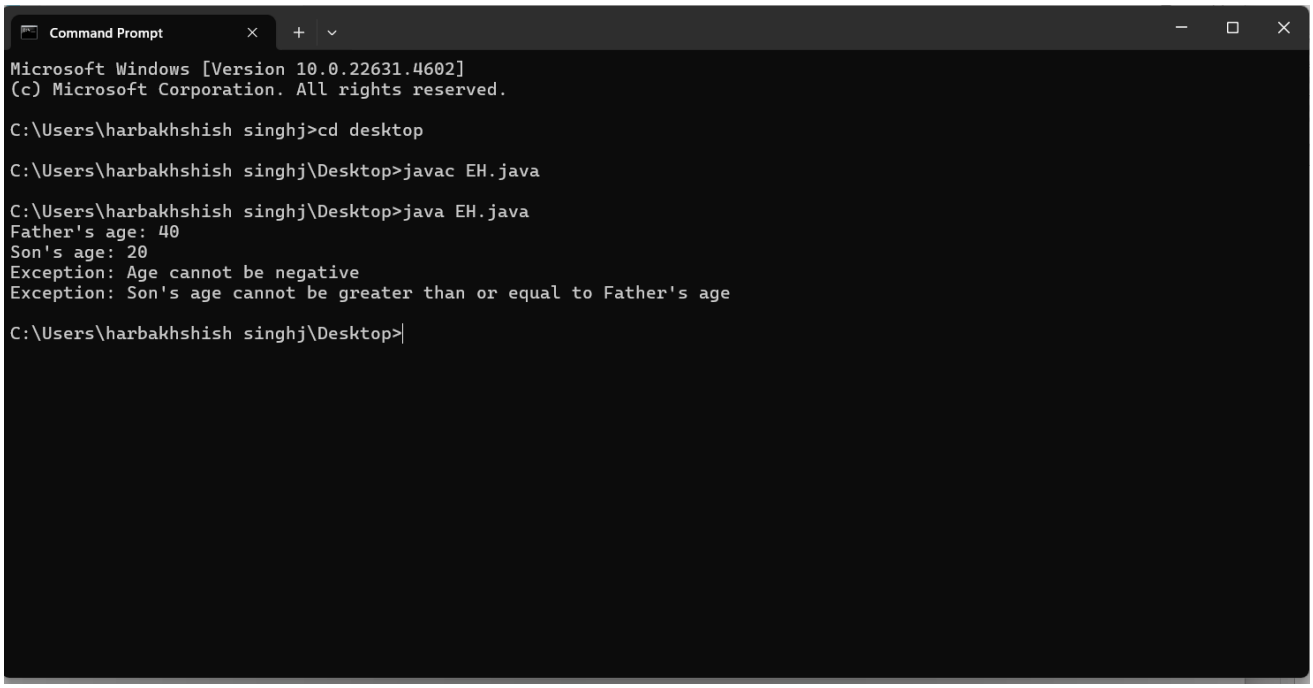
father age is Valid
Son age is Valid

Code-

```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

class Father {
    private int age;

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

    public int getAge() {
        return age;
    }
}

class Son extends Father {
    private int sonAge;

    public Son(int fatherAge, int sonAge) throws
WrongAgeException {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age cannot be
greater than or equal to Father's age");
        }
        this.sonAge = sonAge;
```

```
    }

    public int getSonAge() {
        return sonAge;
    }
}

public class ExceptionHandlingInheritance {
    public static void main(String[] args) {
        try {
            Father father = new Father(40);
            Son son = new Son(40, 20);
            System.out.println("Father's age: " + father.getAge());
            System.out.println("Son's age: " + son.getSonAge());
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        try {
            Father father = new Father(-5);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        try {
            Son son = new Son(30, 35);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

Output-



Program 8-
We have created an interface named Polygon. It includes a default method
getPerimeter() and an abstract method getArea().
We can calculate the perimeter of all polygons in the same manner so we
implemented the body of getPerimeter() in Polygon.
Now, all polygons that implement Polygon can use getPerimeter() to calculate
perimeter.
However, the rule for calculating the area is different for different polygons.
Hence, getArea() is included without implementation.
Any class that implements Polygon must provide an implementation of
getArea()

Algorithm-

1→ Interface named Polygon. It includes a default method getPerimeter() and an abstract method getArea(). We can calculate perimeter of all polygons in the same manner so we implement body of getPerimeter() in polygon. Any class that implement Polygon must provide implementation of getArea().

```
import Java.Util.Scanner;

interface Polygon {
default double
getPerimeter(double[] sides) {
  double Perimeter = 0;
for (double side : sides) {
  perimeter += side;
}
  return perimeter;
}
  double getArea();
}
class Rectangle implements
 Polygon {
  private double length;
  private double width;

public Rectangle(double length, double width)

{
  this.length = length;
  this.width = width;

~~public Rectangle(double length, double width)~~
{
public double getArea()
{
  return 0.5 * base * height;
}
}
```

```
public class Perimeter {
public static void main(String[] args)
{
Scanner scanner = new Scanner(System.in);
System.out.println("Enter length & width
                    of rectangle");
double rectLength = scanner.nextDouble();
double rectWidth = scanner.nextDouble();
Rectangle rect = new Rectangle(rectLength,
                               rectWidth);
System.out.println("Enter four sides
                    of rectangle");
double[] rectSides = new double[4];
for (int i=0; i<4; i++)
{
rectSides[i] = scanner.nextDoub
}
System.out.println("Rectangle Perimeter:" +
                   rect.getPerimeter(rectSides));
System.out.println("Rectangle Area:" +
                   rect.getArea());

System.out.println("Enter the base and
                    height of rectangle");
double triBase = Scanner.nextDouble();
double triHeight = scanner.nextDo
```

```
Triangle tri = new
Triangle (triBase, triHeight);
System.out.println("Enter three sides of
                    triangle");
double[] triSides = new double[3];
for(int i=0; i<3; i++) {

  triSides[i] = Scanner.nextDouble();
}
System.out.println("Triangle Perimeter:" +
    tri.getPerimeter(triSides));

System.out.println("Triangle Area:" +
                    tri.getArea();

  Scanner.close();
}
}
```

output: Enter the length and width of rectangle;

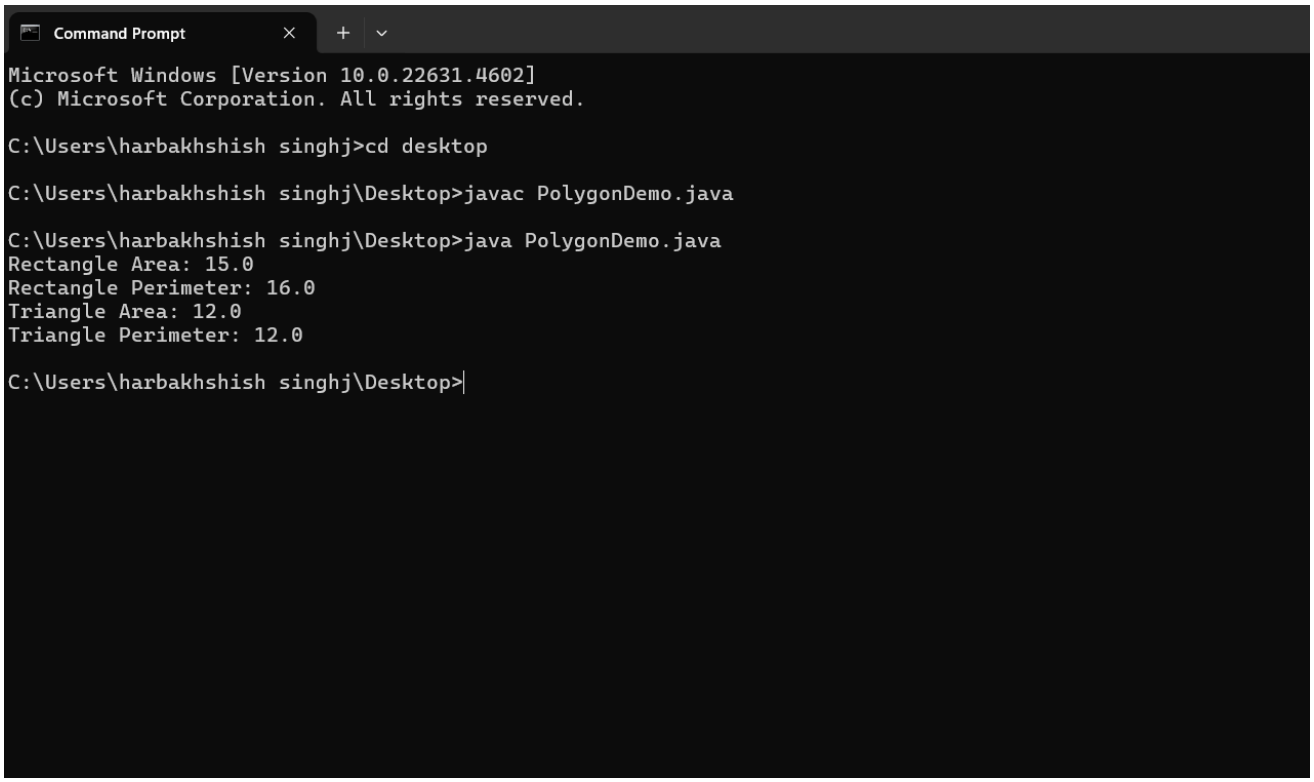
5 10;
Enter four sides of rectangle;
5 10 5 10
Rectangle Perimeter: 30.0
Rectangle Area: 50.0
Enter base and height of the triangle:
6 8 10
Triangle Perimeter: 24.0
Triangle Area: 24.0

2G11

Code-

```
interface Polygon {
    default double getPerimeter(double... sides) {
        double perimeter = 0;
        for (double side : sides) {
            perimeter += side;
        }
        return perimeter;
    }

    double getArea();
}

class Rectangle implements Polygon {
    private double length;
    private double breadth;

    public Rectangle(double length, double breadth) {
        this.length = length;
        this.breadth = breadth;
    }

    @Override
    public double getArea() {
        return length * breadth;
    }
}

class Triangle implements Polygon {
    private double base;
    private double height;

    public Triangle(double base, double height) {
```

```
        this.base = base;
        this.height = height;
    }

    @Override
    public double getArea() {
        return 0.5 * base * height;
    }
}

public class PolygonDemo {
    public static void main(String[] args) {
        Polygon rectangle = new Rectangle(5, 3);
        System.out.println("Rectangle Area: " + rectangle.getArea());
        System.out.println("Rectangle Perimeter: " +
rectangle.getPerimeter(5, 3, 5, 3));

        Polygon triangle = new Triangle(4, 6);
        System.out.println("Triangle Area: " + triangle.getArea());
        System.out.println("Triangle Perimeter: " +
triangle.getPerimeter(3, 4, 5));
    }
}
```
Output-



Program 9-
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.

Algorithm-

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 Displaythread extends thread {
private String message;
private int delay;
public Display thread (String message, int delay)
{
  this.message = message;
  this.delay = delay;
}
@override
public void run () {

try {
    while (True) {
        System.out.println (message);
          Thread.sleep (delay);

}
} catch (InterruptedException e) {

    System.out.println ("thread interrupted:" +
                        e.getMessage ());

}
}
}

public class ThreadDisplay {
```

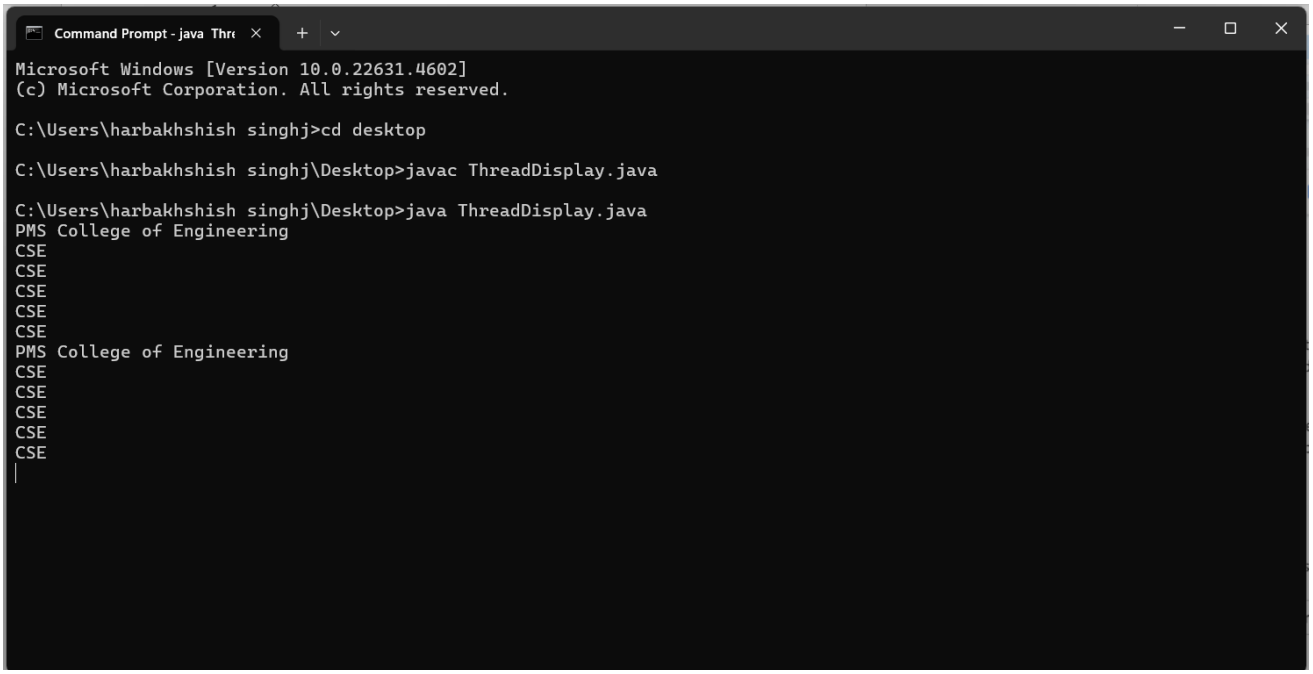
```
public static void main(String[] args)
{
    thread t1 = new Display thread
                ("BMS college of Engineering", 10000);
    Thread t2 = new Display thread ("CSE", 2000);
    t1.start();
    t2.start();
}
}
```

output: BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
CSE
.
.
.
.
.
.

Code-

```
class DisplayThread extends Thread {
    private String message;
    private int delay;

    public DisplayThread(String message, int delay) {
        this.message = message;
        this.delay = delay;
    }

    @Override
    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(delay);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted: " +
e.getMessage());
        }
    }
}

public class ThreadDisplay {
    public static void main(String[] args) {
        Thread t1 = new DisplayThread("PMS College of
Engineering", 10000);
        Thread t2 = new DisplayThread("CSE", 2000);

        t1.start();
        t2.start();
    }
```

}
Output-



Program 10-

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

Algorithm-

Q-1 Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num 1 and Num 2. Division of Num 1 and Num 2 is displayed in the Result. If Num 1 and Num 2 were not an integer, the program would throw a NumberFormat Exception. If Num 2 were zero, program would throw an Arithmetic Exception display the exception in a message dialog box.

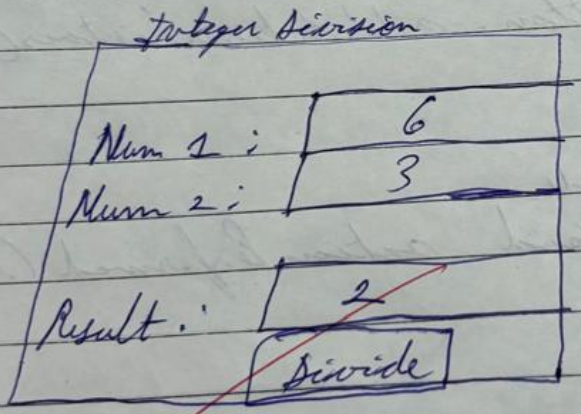
```
import Javax.swing.*;
import Java.awt.event.ActionEvent;
import Java.awt.event.ActionListener;

public class IntegerDivisionApp {

public static void main(String[] args){

JFrame frame = new JFrame("Integer Division");
frame.setSize(300,200);
frame.setDefaultCloseOperation(JFrame.EXIT_ON
                               -CLOSE);
frame.setLayout(null);

~~JLabel num1Label = new JLabel("Num 1 :");~~
~~num1Label.setBounds~~

JTextField num1Field = new JTextField(10);
JTextField num2Field = new JTextField(10);
JTextField resultField = new JTextField(10);
```

Bafna Gold
Date: Page:

```
resultField.setEditable(false);
JButton divideButton = new JButton("Divide");
JPanel panel = new JPanel();
panel.setLayout(new GridLayout(4,2));
panel.add(num1Field);
panel.add(new JLabel("Num2 :"));
panel.add(num2Field);
panel.add(new JLabel("Result :"));
panel.add(resultField);
panel.add(new JLabel(" "));
panel.add(divideButton);
frame.add(panel);

divideButton.addActionListener(new ActionListener()
{
@Override
public void actionPerformed(ActionEvent e)
{
try {
int num1 = Integer.parseInt(num1Field.getText());
int num2 = Integer.parseInt(num2Field.getText());
int result = num1 / num2;
resultField.setText(String.valueOf(result));
}
catch (NumberFormatException ex)
{
JOptionPane.showMessageDialog(frame, "Invalid input. Enter integers only", "Error",
JOptionPane.ERROR_MESSAGE);
```

```
} catch (ArithmeticException ex) {
    JOptionPane.showMessageDialog(frame,
            "Invalid Input it Division By Zero not allowed",
            "Error",
            JOptionPane.ERROR_MESSAGE);
}
}
});
frame.setVisible(true);
}
}
```

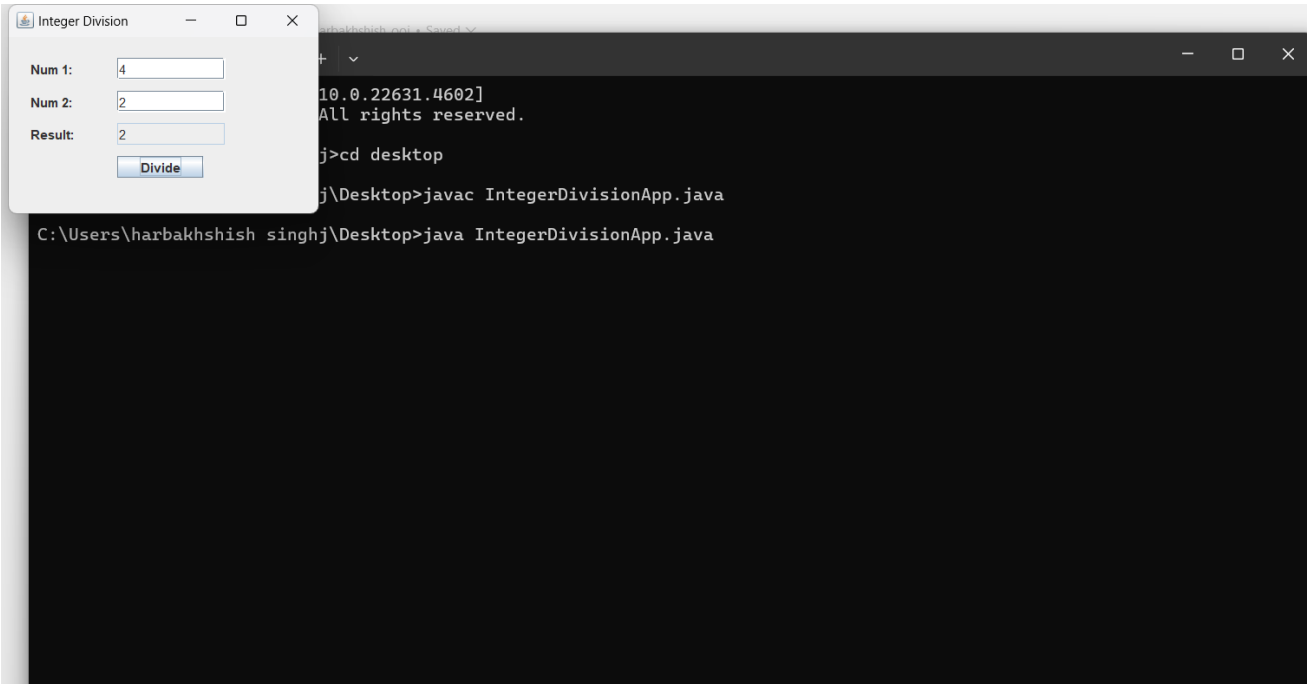
output:



Code-

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class IntegerDivisionApp {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Integer Division");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(null);

        JLabel num1Label = new JLabel("Num 1:");
        num1Label.setBounds(20, 20, 80, 20);
        JTextField num1Field = new JTextField();
        num1Field.setBounds(100, 20, 100, 20);

        JLabel num2Label = new JLabel("Num 2:");
        num2Label.setBounds(20, 50, 80, 20);
        JTextField num2Field = new JTextField();
        num2Field.setBounds(100, 50, 100, 20);

        JLabel resultLabel = new JLabel("Result:");
        resultLabel.setBounds(20, 80, 80, 20);
        JTextField resultField = new JTextField();
        resultField.setBounds(100, 80, 100, 20);
        resultField.setEditable(false);

        JButton divideButton = new JButton("Divide");
        divideButton.setBounds(100, 110, 80, 20);

        divideButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
```

```
            try {
                int num1 = Integer.parseInt(num1Field.getText());
                int num2 = Integer.parseInt(num2Field.getText());
                int result = num1 / num2;
                resultField.setText(String.valueOf(result));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(frame, "Invalid
input! Enter integers.");
            } catch (ArithmeticException ex) {
                JOptionPane.showMessageDialog(frame, "Cannot
divide by zero.");
            }
        }
    });

    frame.add(num1Label);
    frame.add(num1Field);
    frame.add(num2Label);
    frame.add(num2Field);
    frame.add(resultLabel);
    frame.add(resultField);
    frame.add(divideButton);

    frame.setVisible(true);
  }
}
```
Output-


Integer Division
Num 1:
4
Num 2:
2
Result:
2
Divide
10.0.22631.4602]
All rights reserved.
j>cd desktop
j\Desktop>javac IntegerDivisionApp.java
C:\Users\harbakhshish singhj\Desktop>java IntegerDivisionApp.java