

**B.M.S COLLEGE OF ENGINEERING BENGALURU**  
Autonomous Institute, Affiliated to VTU



**AAT**

**22CS3PCOOJ**

**LAB OBSERVATION**

*Submitted in partial fulfillment of the requirements for LAB COMPONENT*

Bachelor of Engineering  
in  
Computer Science and Engineering

*Submitted by:*

**MOHITH JAIN**  
**1BM22CS162**

## Table of Contents

Sl No	Title	Page.no
1	Quadratic equations	3
2	Student SGPA Calculator	6
3	Book using toString()	10
4	Shape using abstract class	13
5	Bank (Savings/Current account)	16
6	Exception Handling (Father and Son)	22
7	Threads	26
8	Division of two integers on user interface	29
9	Packages (CIE / SEE)	33
10	Abstract Window Toolkit Report	38

1. 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.

- ①. 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.

```

import java.util.Scanner;
class roots
{
    public static void main (String xx[])
    {
        System.out.println("M0 with Jain - IBM 22cs162");
        double r1, r2;
        Scanner s1 = new Scanner (System.in);
        System.out.println("Enter coefficients of equation");
        double a = s1.nextDouble();
        double b = s1.nextDouble();
        double c = s1.nextDouble();
        double d = (b*b) - (4*a*c);
        if (d==0)
        {
            System.out.println("Roots are real and equal");
            r1 = r2 = -b / (2*a);
            System.out.println("R1 = " + r1);
            System.out.println("R2 = " + r2);
        }
        else if (d>0)
        {
            System.out.println("Roots are real and distinct");
            r1 = (-b + Math.sqrt(d)) / (2*a);
            r2 = (-b - Math.sqrt(d)) / (2*a);
            System.out.println("R1 = " + r1);
            System.out.println("R2 = " + r2);
        }
    }
}

```

classmate

Date 18/12/23  
Page 02

else

```

    {
        System.out.println("Roots are distinct & imaginary");
        double x = -b / (2 * a);
        double y = Math.sqrt(-d) / (2 * a);
        System.out.println("R1=" + x + " + i" + y);
        System.out.println("R2=" + x + " + i" + y);
    }
}

```

Output:

Monith Jain - IBM2225162

Enter the coefficients of equation

1 2 1

Roots are real and equal

R1 = -1.0

R2 = -1.0

Enter the coefficients of equation

2 3 1

Roots are real and distinct

R1 = -0.5

R2 = -1.0

Enter the coefficients of equation

2 3 2

Roots are distinct and imaginary

R1 = -0.75 + i 0.6614378277661477

R2 = -0.75 - i 0.6614378277661477

**Output:**

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\bmsce\Desktop\1BM22CS162> javac roots.java
PS C:\Users\bmsce\Desktop\1BM22CS162> java roots
Mohith Jain - 1BM22CS162
Enter cooefecients of equation
1
2
1
Roots are real and equal
R1=-1.0
R2=-1.0
PS C:\Users\bmsce\Desktop\1BM22CS162> java roots
Mohith Jain - 1BM22CS162
Enter cooefecients of equation
2
3
1
Roots are real and distinct
R1=-0.5
R2=-1.0
PS C:\Users\bmsce\Desktop\1BM22CS162> java roots
Mohith Jain - 1BM22CS162
Enter cooefecients of equation
2
3
2
Roots are distinct and imaginary
R1=-0.75+i0.6614378277661477
R2=-0.75-i0.6614378277661477
PS C:\Users\bmsce\Desktop\1BM22CS162> |

```

```

Example Prime Print Calculator Fibon
File Edit View

import java.util.Scanner;
class roots
{
    public static void main (String xx[])
    {
        System.out.println("Mohith Jain - 1BM22CS162");
        double r1,r2;
        Scanner s1=new Scanner(System.in);
        System.out.println("Enter cooefecients of equation");
        double a= s1.nextDouble();
        double b= s1.nextDouble();
        double c= s1.nextDouble();

        double d=(b*b)-(4*a*c);
        if(d==0)
        {
            System.out.println("Roots are real and equal");
            r1=r2=-b/(2*a);
            System.out.println("R1=" +r1);
            System.out.println("R2=" +r2);
        }
        else if(d>0)
        {
            System.out.println("Roots are real and distinct");
            r1=(-b+Math.sqrt(d))/(2*a);
            r2=(-b-Math.sqrt(d))/(2*a);
            System.out.println("R1=" +r1);
            System.out.println("R2=" +r2);
        }
        else
        {
            System.out.println("Roots are distinct and imaginary");
            double x=-b/(2*a);
            double y=Math.sqrt(-d)/(2*a);
            System.out.println("R1=" + x +"+i" +y);
            System.out.println("R2=" + x +"-i" +y);
        }
    }
}

```



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.

②. Develop a java program to create a class student with members usn, name, an array credits & an array marks. Include methods to accept & display details & a method to calculate SGPA of a student.

```
import java.util.Scanner;
```

```
class Student {
```

```
    private String usn;
```

```
    private String name;
```

```
    private int[] credits;
```

```
    private int[] marks;
```

```
    public Student (String usn, String name, int[] credits, int[] marks) // Constructor
```

```
    {
```

```
        this.usn = usn;
```

```
        this.name = name;
```

```
        this.credits = credits;
```

```
        this.marks = marks;
```

```
    }
```

```
    public void acceptDetails()
```

```
    { Scanner scscanner = new Scanner (System.in);
```

```
        System.out.print ("Enter USN:");
```

```
        this.usn = scanner.nextLine();
```

```
        S.O.P ("Enter name:");
```

```
        this.name = scanner.nextLine();
```

```
        this.credits = new int [credits.length];
```

```
        this.marks = new int [marks.length];
```

System.out.println  
S.O. println Subject 1

```

for (int i=0; i < credits.length; i++)
{
    S.O.println("Enter credits for subject "+(i+1)+" :");
    this.credits[i] = scanner.nextInt();

    S.O.println("Enter marks for subject "+(i+1)+" :");
    this.marks[i] = scanner.nextInt();
}

public void displayDetails()
{
    S.O.println("USN: " + usn);
    S.O.println("Name: " + name);
    S.O.println("Credits:");
    for (int i=0; i < credits.length; i++)
    {
        S.O.println("Subject "+(i+1)+" : " + credits[i] + " credits");
    }
    S.O.println("Marks:");
    for (int i=0; i < marks.length; i++)
    {
        S.O.println("Marks Subject "+(i+1)+" : " + marks[i] + " marks");
    }
}

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

```

q6

Student

private int calculateGradePoints (int 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 class Main

```

{
    public static void main (String [] args)
    {
        int numSubjects = 6;
        int[] credits = new int[numSubjects];
        int[] marks = new int[numSubjects];

        Student student = new Student(" ", "John", credits, marks);
        student.acceptDetails();
        System.out.println("Student Details:");
        student.displayDetails();
        double sgpa = student.calculateSGPA();
        System.out.println("SGPA: " + sgpa);
    }
}

```

Enter USN: IBT22CS162

Enter Name: Mohith

Student Details:

Enter credits of subject 1: 4

Enter marks of subject 1: 90

Enter credits of subject 2: 3

Enter marks of subject 2: 88

Enter credit of subject 3: 2

Enter marks of subject 3: 89

Enter credits of -11-4: 3

Enter marks of -11-4: 96

Enter credit of -11-5: 1

Enter marks of -11-5: 90

Student Details:

USN: IBT22CS162

Name: Mohith

Credits:

Subject 1: 4 credit

Subject 2: 3 credit

Subject 3: 2 credit

Subject 4: 3 credit

Subject 5: 1 credit

Marks:

Subject 1: 90 marks

Subject 2: 88 marks

Subject 3: 89 marks

Subject 4: 96 marks

Subject 5: 90 marks

SCPA: 9.23



## Output:

The image shows a Windows PowerShell terminal window on the left and an IDE window on the right. The terminal displays the execution of a Java program named Sgpa.java. The program prompts for USN, Name, and marks for five subjects, then calculates and displays the SGPA. The IDE window shows the source code of the Student and Sgpa classes.

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\bmsce\Desktop\18M22CS162> javac Sgpa.java
PS C:\Users\bmsce\Desktop\18M22CS162> java Sgpa
Enter USN:18M22CS162
Enter Name:Mohith
Student Details:Enter credits of subject1:4
Enter marks of subject1:98
Enter credits of subject2:3
Enter marks of subject2:88
Enter credits of subject3:2
Enter marks of subject3:89
Enter credits of subject4:3
Enter marks of subject4:96
Enter credits of subject5:1
Enter marks of subject5:99
Student Details:
USN:18M22CS162
Name:Mohith
Credits
Subject1:4credit
Subject2:3credit
Subject3:2credit
Subject4:3credit
Subject5:1credit
Marks
Subject1:98marks
Subject2:88marks
Subject3:89marks
Subject4:96marks
Subject5:99marks
SGPA:9.23876923876923
PS C:\Users\bmsce\Desktop\18M22CS162>

import java.util.Scanner;

class Student
{
    private String usn;
    private String name;
    private int[] credits;
    private int[] marks;

    public Student(String usn, String name, int[] credits, int[] marks)
    {
        this.usn = usn;
        this.name = name;
        this.credits = credits;
        this.marks = marks;
    }

    public void acceptDetails()
    {
        Scanner student=new Scanner(System.in);
        System.out.println("Enter USN:");
        this.usn=student.nextLine();
        System.out.println("Enter Name:");
        this.name=student.nextLine();
        System.out.println("Student Details:");
        this.credits = new int[credits.length];
        this.marks = new int[marks.length];
        for(int i=0;i<credits.length;i++)
        {
            System.out.println("Enter credits of subject"+(i+1)+"");
            this.credits[i] = student.nextInt();
            System.out.println("Enter marks of subject"+(i+1)+"");
            this.marks[i] = student.nextInt();
        }
    }

    public void displayDetails()
    {
        System.out.println("USN:"+usn);
        System.out.println("Name:"+name);
        System.out.println("Credits:");
        for(int i=0;i<credits.length;i++)
        {
            System.out.println("Subject"+(i+1)+"="+credits[i]+"credit");
        }
        System.out.println("Marks:");
        for(int i=0;i<marks.length;i++)
        {
            System.out.println("Subject"+(i+1)+"="+marks[i]+"marks");
        }
    }

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

    private int calculateGradePoints(int 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 class Sgpa{
    public static void main(String []args)
    {
        int Subjects=5;
        int[] credits= new int[Subjects];
        int[] marks= new int[Subjects];

        Student student=new Student("18M22CS162","John",credits,marks);
        student.acceptDetails();
        System.out.println("Student Details:\n");
        student.displayDetails();
        double sgpa=student.calculateSGPA();
        System.out.println("\nSGPA:"+sgpa);
    }
}

```

The image shows an IDE window with the Sgpa.java file open. The code defines a Student class and a Sgpa class. The Student class has attributes for USN, Name, credits, and marks, and methods for accepting and displaying details, and calculating SGPA. The Sgpa class has a main method that creates a Student object and calculates the SGPA.

```

File Edit View

public double calculateSGPA()
{
    double totalcredits=0;
    double totalGradePoints=0;

    for(int i=0;i<credits.length;i++)
    {
        totalcredits +=credits[i];
        totalGradePoints +=(calculateGradePoints(marks[i])*credits[i]);
    }
    return totalGradePoints/totalcredits;
}

private int calculateGradePoints(int 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 class Sgpa{
    public static void main(String []args)
    {
        int Subjects=5;
        int[] credits= new int[Subjects];
        int[] marks= new int[Subjects];

        Student student=new Student("18M22CS162","John",credits,marks);
        student.acceptDetails();
        System.out.println("Student Details:\n");
        student.displayDetails();
        double sgpa=student.calculateSGPA();
        System.out.println("\nSGPA:"+sgpa);
    }
}

```

**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.**

③ Develop a java program to 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 & create n Book objects.

```

import java.util.Scanner;

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

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

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

    public void setPrice (double price)
    {
        this.price = price;
    }

    public void setNumPages (int numPages)
    {
        this.numPages = numPages;
    }

    public String getName()
    {
        return name;
    }
}

```

classmate  
Date \_\_\_\_\_  
Page \_\_\_\_\_

```

public String getAuthor()
{
    return author;
}

public double getPrice()
{
    return price;
}

public int getNumPages()
{
    return numPages;
}

public String toString()
{
    return "Book Details: Name: " + name +
        "\n Author: " + author + "\n Price: " + price +
        "\n Number of Pages: " + numPages;
}

public class BookTest
{
    public static void main (String[] args)
    {
        Scanner sc = new Scanner(System.in);

        System.out.println("Enter number of Books:");
        int n = sc.nextInt();

        Book[] books = new Book[n];

```

classmate  
Date \_\_\_\_\_  
Page \_\_\_\_\_

```

for(int i=0; i<n; i++)
{
    System.out.println("Enter details of Book " +
        (i+1) + ":");
    sc.nextLine();
    System.out.println("Enter Book name:");
    String name = sc.nextLine();
    System.out.println("Enter Author name:");
    String author = sc.nextLine();
    System.out.println("Enter price:");
    double price = sc.nextDouble();
    System.out.println("Enter number of pages:");
    int numPages = sc.nextInt();

    books[i] = new Book(name, author, price, numPages);
}

for(int i=0; i<n; i++)
{
    System.out.println("\n Details of Book " + (i+1) +
        ": " + books[i]);
}

sc.close();
}

```

Output: Enter the Details for Book 1  
 Enter Book Name: Code with Java  
 Enter author name: S.N. Bose  
 Enter price: 780  
 Enter number of pages: 562  
 Enter the Details for Book 2  
 Enter Book Name: mySQL  
 Enter author name: L. Navathe  
 Enter price: 950  
 Enter number of Pages: 1452

Details of Book 1:  
 Book Details  
 Name: Code with Java  
 Author: S.N. Bose  
 Price: 780.0  
 Number of Pages: 562

Details of Book 2:  
 Book Details  
 Name: mySQL  
 Author: L. Navathe  
 Price: 950.0  
 Number of Pages: 1452

8/11

## Output:

The screenshot displays a Windows desktop environment. On the left, a Windows PowerShell terminal window is open, showing the execution of a Java program. The program prompts the user to enter the number of books (2), and then for each book, it prompts for the name, author, price, and number of pages. The output shows the details for two books: Book1 (Name: Code with Java, Author: S.M Bose, Price: 780.0, Pages: 562) and Book2 (Name: MySQL, Author: L.Navathe, Price: 950.0, Pages: 1452). On the right, a Java IDE window is open, showing the source code for the Book and BookTest classes. The Book class has private attributes for name, author, price, and numPages, and public methods for setting and getting these attributes. The BookTest class has a main method that uses a Scanner to read user input and instantiate Book objects.

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\umsce\Desktop\18M22CS162> javac BookTest.java
PS C:\Users\umsce\Desktop\18M22CS162> java BookTest
Enter number of Books:
2
Enter the Details for Book1:
Enter Book Name:
Code with Java
Enter Author name:
S.M Bose
Enter price:
780
Enter number of pages:
562
Enter the Details for Book2:
Enter Book Name:
MySQL
Enter Author name:
L.Navathe
Enter price:
950
Enter number of pages:
1452

Details of Book1:Book Details:
NameCode with Java
AuthorS.M Bose
Price780.0
Number of Pages562

Details of Book2:Book Details:
NameMySQL
AuthorL.Navathe
Price950.0
Number of Pages1452
PS C:\Users\umsce\Desktop\18M22CS162> |

File Edit View
Main.java Method.java Sgpa.java BookTest.java
import java.util.Scanner;

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

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

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

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

    public void setPrice(double price)
    {
        this.price=price;
    }

    public void setNumPages(int NumPages)
    {
        this.numPages=numPages;
    }

    public String getName()
    {
        return name;
    }

    public String getAuthor()
    {
        return author;
    }

    public double getPrice()
    {
        return price;
    }

    public int getNumPages()
    {
        return numPages;
    }

    public String toString()
    {
        return "Book Details:\nName:"+name+"\nAuthor:"+author+"\nPrice:"+price+"\nNumber of Pages:"+numPages;
    }
}

public class BookTest
{
    public static void main(String[] args)
    {
        Scanner scanner=new Scanner(System.in);

        System.out.println("Enter number of Books:");
        int n=scanner.nextInt();

        Book[] books=new Book[n];

        for(int i=0;i<n;i++)
        {
            System.out.println("Enter the Details for Book"+(i+1)+":");
            scanner.nextLine();
            System.out.println("Enter Book Name:");
            String name=scanner.nextLine();
            System.out.println("Enter Author name:");
            String author=scanner.nextLine();
            System.out.println("Enter price:");
            double price=scanner.nextDouble();
            System.out.println("Enter number of pages:");
            int numPages=scanner.nextInt();

            books[i]=new Book(name,author,price,numPages);
        }

        for(int i=0;i<n;i++)
        {
            System.out.println("\nDetails of Book"+(i+1)+": "+books[i]);
        }

        scanner.close();
    }
}

```

```

public class BookTest
{
    public static void main(String[] args)
    {
        Scanner scanner=new Scanner(System.in);

        System.out.println("Enter number of Books:");
        int n=scanner.nextInt();

        Book[] books=new Book[n];

        for(int i=0;i<n;i++)
        {
            System.out.println("Enter the Details for Book"+(i+1)+":");
            scanner.nextLine();
            System.out.println("Enter Book Name:");
            String name=scanner.nextLine();
            System.out.println("Enter Author name:");
            String author=scanner.nextLine();
            System.out.println("Enter price:");
            double price=scanner.nextDouble();
            System.out.println("Enter number of pages:");
            int numPages=scanner.nextInt();

            books[i]=new Book(name,author,price,numPages);
        }

        for(int i=0;i<n;i++)
        {
            System.out.println("\nDetails of Book"+(i+1)+": "+books[i]);
        }

        scanner.close();
    }
}

```

Activate Windows  
Go to Settings to activate Windows.



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 print Area( ) that prints the area of the given shape.

Details of Book 1:  
Book Details  
Name: Code with Java  
Author: S.N. Bose  
Price: 780.0  
Number of Pages: 562

Details of Book 2:  
Book Details  
Name: MySOL  
Author: I. Navroth  
Price: 950.0  
Number of Pages: 1162

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, 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.

```

abstract class Shape
{
    public int length;
    public int width;

    public Shape(int length, int width)
    {
        this.length = length;
        this.width = width;
    }

    abstract void printArea();
}

class Rectangle extends Shape
{
    public Rectangle(int length, int width)
    {
        super(length, width);
    }

    void printArea()
    {
        int area = length * width;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Triangle extends Shape
{
    public Triangle(int length, int width)
    {
        super(length, width);
    }

    void printArea()
    {
        double area = 0.5 * length * width;
        System.out.println("Area of Triangle: " + area);
    }
}

```

```

class Circle extends Shape
{
    public Circle(int radius)
    {
        super(radius, 0);
    }
    void printArea()
    {
        double Area = Math.PI * length * length;
        System.out.println("Area of circle : " + Area);
    }
}

```

```

public class ShapePCA {
    public static void main(String[] args)
    {
        Rectangle rectangle = new Rectangle(8, 20);
        rectangle.printArea();

        Triangle triangle = new Triangle(7, 11);
        triangle.printArea();

        Circle circle = new Circle(6);
        circle.printArea();
    }
}

```

Output

Area of Rectangle: 160  
 Area of Triangle: 38.5  
 Area of Circle: 113.0973355



```
abstract class Shape {
    public int length;
    public int width;

    public Shape(int length, int width) {
        this.length = length;
        this.width = width;
    }

    abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    void printArea() {
        int area = length * width;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int length, int width) {
        super(length, width);
    }

    void printArea() {
        double area = 0.5 * length * width;
        System.out.println("Area of Triangle: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }

    void printArea() {
        double area = Math.PI * length * length;
        System.out.println("Area of Circle: " + area);
    }
}

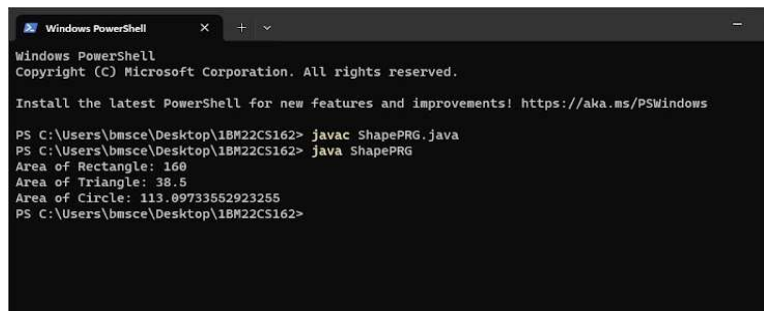
public class ShapePRG {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(8, 20);
        rectangle.printArea();

        Triangle triangle = new Triangle(7, 11);
        triangle.printArea();

        Circle circle = new Circle(6);
        circle.printArea();
    }
}
```

Ln 1, Col 1 | 90% | Windows (CRLF) | UTF-8

Activate Windows  
Go to Settings to activate Windows.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\bmsce\Desktop\IBM22CS162> javac ShapePRG.java
PS C:\Users\bmsce\Desktop\IBM22CS162> java ShapePRG
Area of Rectangle: 160
Area of Triangle: 38.5
Area of Circle: 113.09733552923255
PS C:\Users\bmsce\Desktop\IBM22CS162>
```



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:

- Accept deposit from customer and update the balance.
  - Display the balance.
  - Compute and deposit interest
  - Permit withdrawal and update the balance
- Check for the minimum balance, impose penalty if necessary and update the balance.

classmate  
Date 19/2/23  
Page

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 holder should also maintain a minimum balance & if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number & type of account. From this derive the classes Cur\_acct & Sav\_acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

- Accept deposit from customer & update balance.
- Display the balance.
- Compute & deposit interest.
- Permit withdrawal & update balance.

```
import java.util.Scanner;

class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    public Account(String customerName, int accountNumber, String accountType, double balance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }
}
```



```

public void deposit (double amount)
{
    balance += amount;
    System.out.println ("Deposit of $" + amount + "
                        successful.");
}

public void displayBalance ()
{
    System.out.println ("Current Balance: $" + balance);
}

public void withdraw (double amount)
{
    if (balance >= amount)
    {
        balance -= amount;
        System.out.println ("Withdrawal of $" + amount + "
                        successful.");
    }
    else
    {
        System.out.println ("Insufficient balance.");
    }
}

class SavingsAccount extends Account
{
    double interestRate;

    public SavingsAccount (String customerName,
        int accountNumber, String accountType,
        double balance, double interestRate)
    {
        super (customerName, accountNumber, accountType,
            balance);
        this.interestRate = interestRate;
    }

    public void computeInterest ()
    {
        double interest = balance * (interestRate/100);
        balance += interest;
        S.o.p ("Interest of $" + interest + " computed &
            deposited.");
    }

    class CurrentAccount extends Account
    {
        double minBalance;
        double serviceCharge;

        public CurrentAccount (String customerName, int
            accountNumber, String accountType, double
            balance, double minBalance, double serviceCharge)
        {
            super (customerName, accountNumber, accountType,
                balance);
            this.minBalance = minBalance;
            this.serviceCharge = serviceCharge;
        }

        public void withdraw (double amount)
        {
            if (balance - amount >= minBalance)
            {
                balance -= amount;
                S.o.p ("Withdrawal of $" + amount + " successful.");
            }
            else
            {
                S.o.p ("Withdrawal failed due to insufficient balance
                    or minimum balance constraint.");
            }
        }
    }
}

```

```

public void checkMinimumBalance ()
{
    if (balance < minBalance)
    {
        balance -= serviceCharge;
        S.o.p ("Service charge of $" + serviceCharge + "
            imposed due to falling below minimum
            balance.");
    }
}

public class Bank
{
    public static void main (String[] args)
    {
        Scanner scanner = new Scanner (System.in);

        S.o.p ("Welcome to the bank.");
        S.o.p ("Enter customer name:");
        String customerName = scanner.nextLine();

        S.o.p ("Enter account number:");
        int accountNumber = scanner.nextInt();
        scanner.nextLine();

        S.o.p ("Enter account type (Savings/Current):");
        String accountType = scanner.nextLine();
        S.o.p ("Enter initial balance:");
        double initialBalance = scanner.nextDouble();

        Account account;
        if (accountType.equals (ignoreCase ("Savings")))
        {
            S.o.p ("Enter interest Rate:");
            double interestRate = scanner.nextDouble();
            account = new SavingsAccount (customerName,
                accountNumber, accountType, initialBalance,
                interestRate);
        }
        else if (accountType.equals (ignoreCase ("Current")))
        {
            S.o.p ("Enter minimum balance:");
            double minBalance = scanner.nextDouble();
            S.o.p ("Enter service charge:");
            double serviceCharge = scanner.nextDouble();

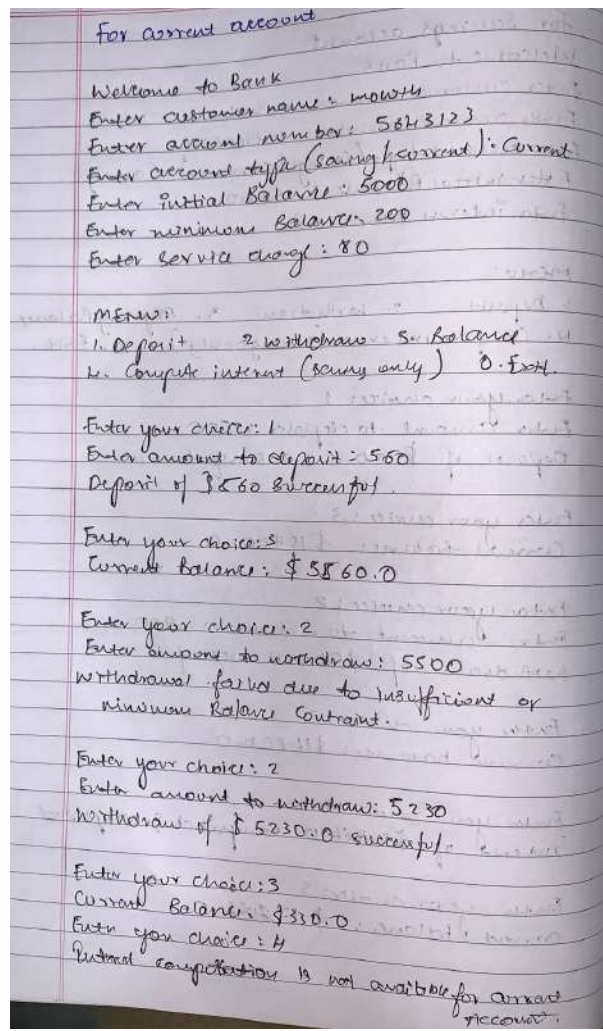
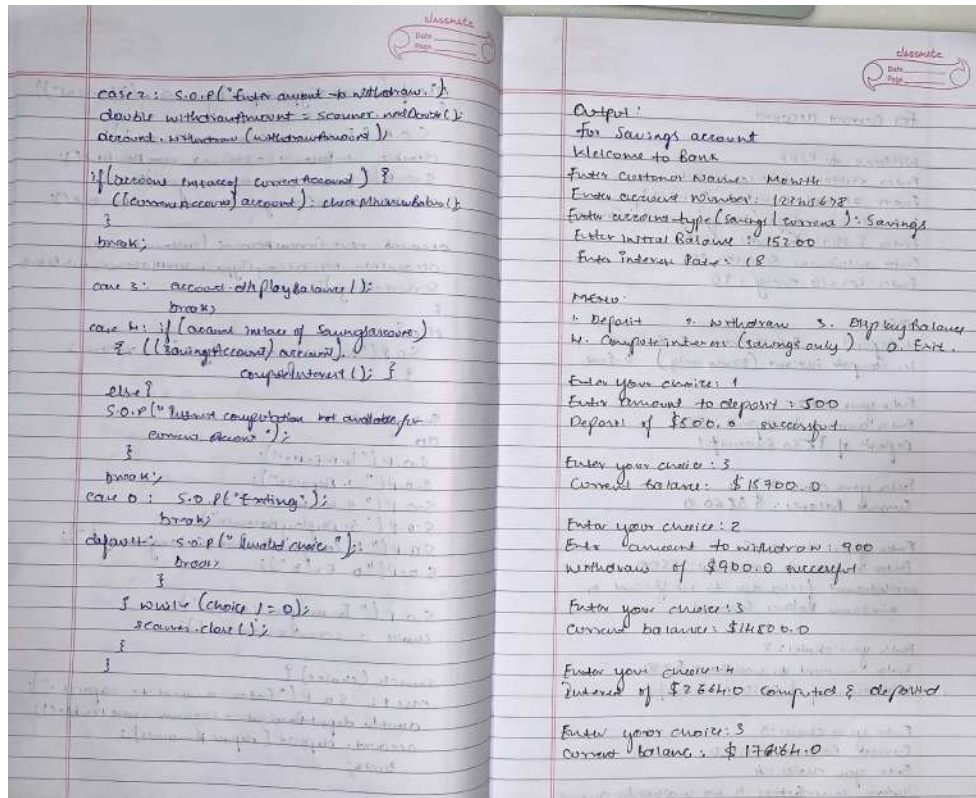
            account = new CurrentAccount (customerName,
                accountNumber, accountType, initialBalance, minBalance,
                serviceCharge);
        }
        else
        {
            S.o.p ("Invalid account type.");
            return;
        }

        int choice;
        do
        {
            S.o.p ("In MENU");
            S.o.p ("1. Deposit");
            S.o.p ("2. Withdrawal");
            S.o.p ("3. Display Balance");
            S.o.p ("4. Compute Interest (Savings only)");
            S.o.p ("0. Exit");

            S.o.p ("Enter your choice:");
            choice = scanner.nextInt();

            switch (choice)
            {
                case 1: S.o.p ("Enter amount to deposit:");
                    double depositAmount = scanner.nextDouble();
                    account.deposit (depositAmount);
                    break;
            }
        }
    }
}

```





## Output:

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements: https://aka.ms/PSWindows

PS C:\Users\umsc\Desktop\IDM2C5162> javac Bank.java
PS C:\Users\umsc\Desktop\IDM2C5162> java Bank
Welcome to the Bank!
Enter customer name: Mohith
Enter account number: 125623019
Enter account type (Savings/Current): Savings
Enter initial balance: 15200
Enter interest rate: 18

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 1
Current balance: $15200.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 1
Enter amount to deposit: 500
Deposit of $500.0 successful.

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 3
Current balance: $15700.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 2
Enter amount to withdraw: 900
Withdrawal of $900.0 successful.

```

```

Bank.java
import java.util.Scanner;

class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    public Account(String customerName, int accountNumber, String accountType, double balance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance = amount;
        System.out.println("Amount of $" + amount + " deposited successfully.");
    }

    public void withdraw(double amount) {
        System.out.println("Current balance: $" + balance);

        if (amount >= balance) {
            balance = amount;
            System.out.println("Withdrawal of $" + amount + " successful.");
        } else {
            System.out.println("Insufficient funds.");
        }
    }

    class SavingsAccount extends Account {
        double interestRate;

        public SavingsAccount(String customerName, int accountNumber, String accountType, double balance, double interestRate) {
            super(customerName, accountNumber, accountType, balance);
            this.interestRate = interestRate;
        }

        public void computeInterest() {
            double interest = balance * (interestRate / 100);
            balance = interest;
            System.out.println("Interest of $" + interest + " computed and deposited.");
        }
    }

    class CurrentAccount extends Account {
        double serviceCharge;

        public CurrentAccount(String customerName, int accountNumber, String accountType, double balance, double serviceCharge) {
            super(customerName, accountNumber, accountType, balance);
            this.serviceCharge = serviceCharge;
        }

        public void withdraw(double amount) {
            if (balance - amount <= 0) {
                balance = amount;
                System.out.println("Withdrawal of $" + amount + " successful.");
            } else {
                System.out.println("Withdrawal failed due to insufficient funds or service charge constraint.");
            }
        }

        public void computeServiceCharge() {
            if (balance < 0) {
                balance = serviceCharge;
                System.out.println("Service charge of $" + serviceCharge + " applied due to falling below minimum balance.");
            }
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Welcome to the Bank!");
        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account number: ");
        int accountNumber = scanner.nextInt();

        System.out.println("Enter account type (Savings/Current): ");
        String accountType = scanner.nextLine();

        System.out.print("Enter initial balance: ");
        double initialBalance = scanner.nextDouble();

        Account account;
        if (accountType.equals("Savings")) {
            System.out.print("Enter interest rate: ");
            double interestRate = scanner.nextDouble();
            account = new SavingsAccount(customerName, accountNumber, accountType, initialBalance, interestRate);
        } else if (accountType.equals("Current")) {
            System.out.print("Enter service charge: ");
            double serviceCharge = scanner.nextDouble();
            account = new CurrentAccount(customerName, accountNumber, accountType, initialBalance, serviceCharge);
        } else {
            System.out.println("Invalid account type.");
            return;
        }

        int choice;
        do {
            System.out.println("Menu:");
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Display Balance");
            System.out.println("4. Compute Interest (Savings only)");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter amount to deposit: ");
                    double depositAmount = scanner.nextDouble();
                    account.deposit(depositAmount);
                    break;
                case 2:
                    System.out.print("Enter amount to withdraw: ");
                    double withdrawAmount = scanner.nextDouble();
                    account.withdraw(withdrawAmount);
                    if (choice == 2) {
                        System.out.println("Withdrawal failed due to insufficient funds or service charge constraint.");
                    }
                    break;
                case 3:
                    account.displayBalance();
                    break;
                case 4:
                    if (account instanceof SavingsAccount) {
                        ((SavingsAccount) account).computeInterest();
                    } else {
                        System.out.println("Interest computation is not available for Current accounts.");
                    }
                    break;
                case 5:
                    System.out.print("Would you like to continue? (yes/no): ");
                    String choiceStr = scanner.nextLine();
                    if (choiceStr.equalsIgnoreCase("no")) {
                        break;
                    }
                    break;
            }
        } while (choice != 5);
    }
}

```

```
Windows PowerShell
Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 3
Current balance: $14800.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 4
Interest of $2664.0 computed and deposited.

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 3
Current balance: $17464.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 2
Enter amount to withdraw: 17000
Withdrawal of $17000.0 successful.

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 3
Current balance: $464.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
```



```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements: https://aka.ms/PSWindows

PS C:\Users\bmsce\Desktop\1BM22CS162> java Bank
Welcome to the Bank!
Enter customer name: mohith
Enter account number: 5641123
Enter account type (Savings/Current): Current
Enter initial balance: 5000
Enter minimum balance: 200
Enter service charge: 80

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 1
Current balance: $5000.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 1
Enter amount to deposit: 500
Deposit of $500.0 successful.

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 3
Current balance: $5500.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 2
Enter amount to withdraw: 5000
Withdrawal failed due to insufficient funds or minimum balance constraint.

Activate Windows
Go to Settings to activate Windows.

```

```

Windows PowerShell

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 3
Current balance: $5500.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 2
Enter amount to withdraw: 5230
Withdrawal of $5230.0 successful.

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 3
Current balance: $330.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 4
Interest computation is not available for Current accounts.

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
0. Exit
Enter your choice: 0
Exiting...
PS C:\Users\bmsce\Desktop\1BM22CS162> |

```

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

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

Class WrongAge extends Exception

{

WrongAge (String message)

{

super (message);

}

}

// Base class father

class Father

{

private int age;

public Father (int age) throws WrongAge

{

if (age < 0)

{

throw new WrongAge ("Age can't be negative");

}

this.age = age;

}

public int getAge() {

return age;

}

}

// Derived class Son

```

class Son extends Father
{
    private int sonAge;
    public Son (int father's age, int sonAge) throws
        WrongAge
    {
        super (age);
        if (sonAge > father's age)
        {
            throw new WrongAge ("Son's age can't be
            greater than or equal to father's age");
        }
        this.sonAge = sonAge;
    }

    public int getSonAge() {
        return sonAge;
    }
}

public class ExceptionSubstance {
    public static void main (String[] args)
    {
        try
        {
            Son son = new Son (10, 20);
            S.o.p ("Father's age: " + son.getAge());
            S.o.p ("Son's age: " + son.getSonAge());
        }
        catch (WrongAge e)
        {
            S.o.p ("Exception: " + e.getMessage());
        }
    }
}

```

son invalid son = new Son(30, 35); <sup>Exception</sup>

S.o.p ("This line will not be reached");

catch (WrongAge e)

S.o.p ("Exception: " + e.getMessage());

Output:

Father's age: 10

Son's age: 20

Exception: Son's age can't be greater than or equal to father's age.

**Output:**

```

Sgpa.java BookTest.java ShapePRG.java Bookrun.java studentrun.java ExceptionInheritance.java
File Edit View

class WrongAge extends Exception{
    WrongAge(String message)
    {
        super(message);
    }
}

class Father{
    private int age;
    public Father(int age)throws WrongAge
    {
        if(age<0)
        {
            throw new WrongAge("Age can't be negative");
        }
        this.age=age;
    }

    public int getAge()
    {
        return age;
    }
}

class Son extends Father
{
    private int sonAge;

    public Son(int age,int sonAge)throws WrongAge
    {
        super(age);

```

```

        if(sonAge>=age)
        {
            throw new WrongAge("Son's age can't be greater than or equal to
father's age");
        }
        this.sonAge=sonAge;
    }

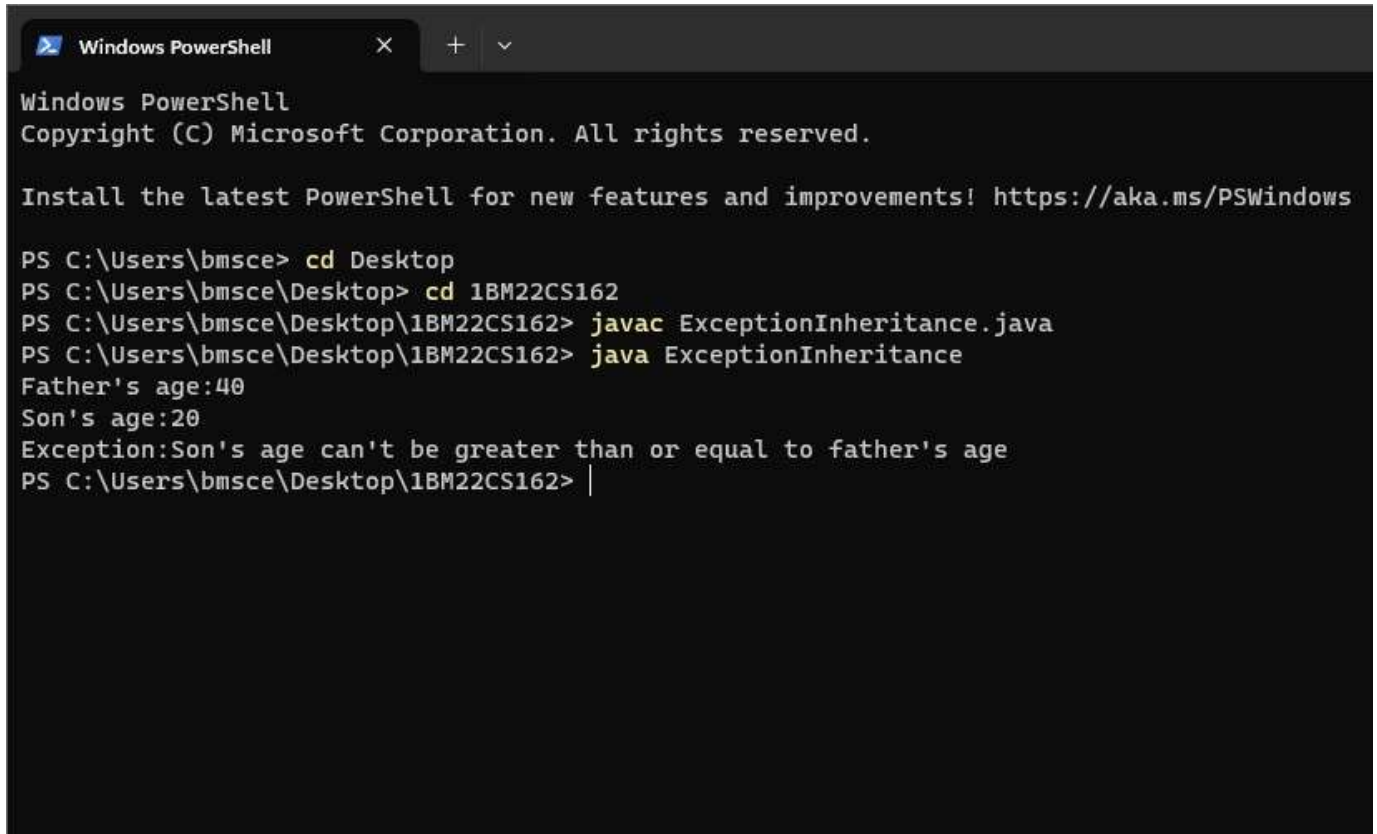
    public int getsonAge()
    {
        return sonAge;
    }
}

public class ExceptionInheritance{
    public static void main(String[]args)
    {
        try
        {
            Son son=new Son(40,20);
            System.out.println("Father's age:"+son.getAge());
            System.out.println("Son's age:"+son.getsonAge());

            Son invalidson =new Son(30,35);
            System.out.println("This line will not be reached");
        }
        catch(WrongAge e)
        {
            System.out.println("Exception:"+ e.getMessage());
        }
    }
}

```





```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\bmsce> cd Desktop
PS C:\Users\bmsce\Desktop> cd 1BM22CS162
PS C:\Users\bmsce\Desktop\1BM22CS162> javac ExceptionInheritance.java
PS C:\Users\bmsce\Desktop\1BM22CS162> java ExceptionInheritance
Father's age:40
Son's age:20
Exception:Son's age can't be greater than or equal to father's age
PS C:\Users\bmsce\Desktop\1BM22CS162> |
```

7..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.

Lab Program

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

```

class NewThread implements Runnable
{
    Thread t;
    NewThread()
    {
        t = new Thread(this, "CollegeName");
        System.out.println("Name: " + t);
        t.start();
    }
    public void run()
    {
        try
        {
            for(int n=2; n>0; n--)
            {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        }
        catch (InterruptedException ie)
        {
            System.out.println("Child thread interrupted");
        }
    }
}

class CollegeRun
{
    public static void main (String ss[])
    {
        new NewThread();
    }
}

```

Date \_\_\_\_\_  
Page \_\_\_\_\_

```

System.out.println("College & Department");
System.out.println("Printing below");

try {
    for(int n=2; n>0; n--)
    {
        System.out.println("BMS College of Engineering");
        Thread.sleep(10000);
    }
}
catch (InterruptedException e)
{
    System.out.println("Main Thread Interrupted");
}

```

Output:

Name: Thread [H20, College Name, 5, main]

College & Department

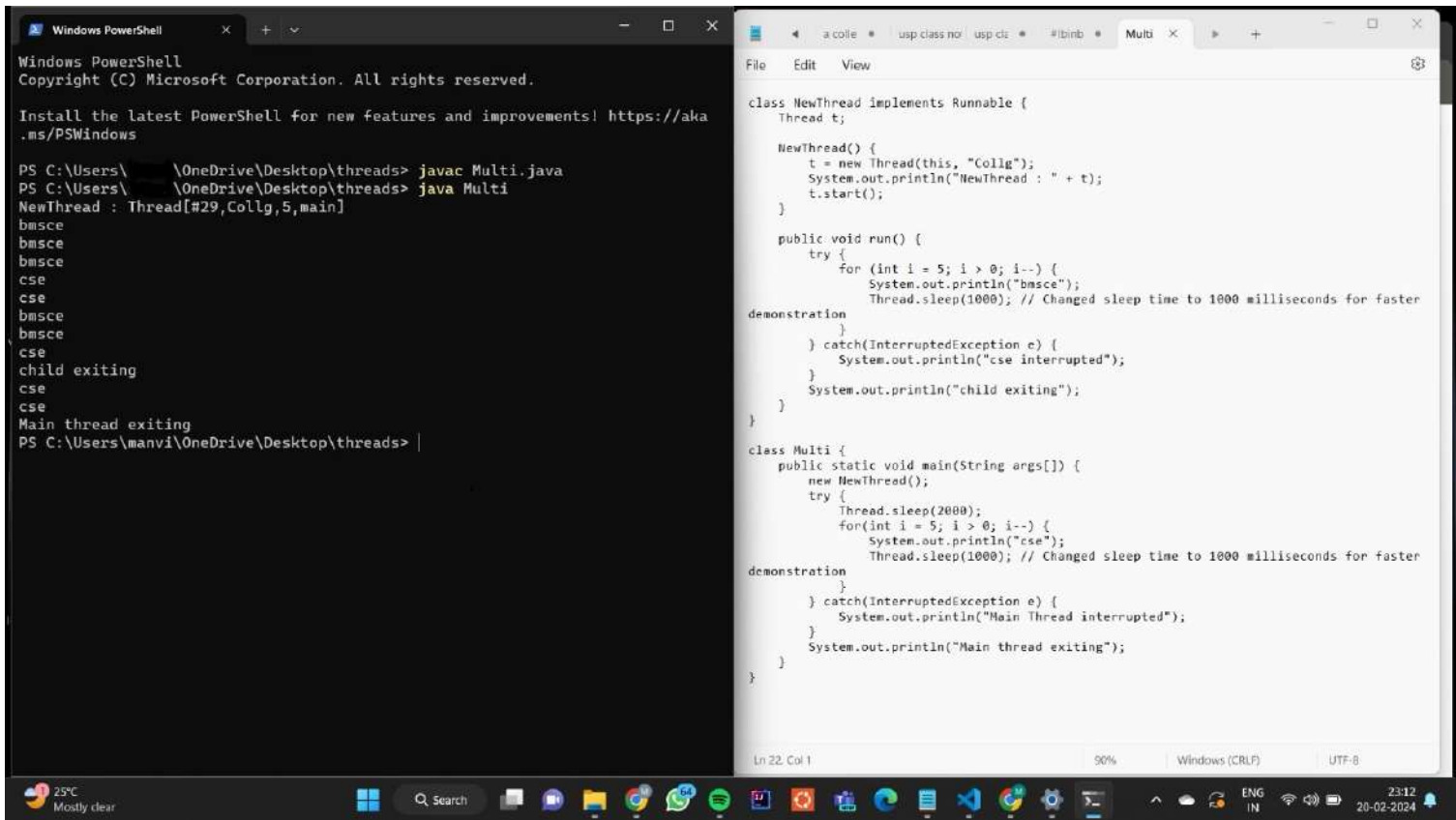
Printing below

BMS College of Engineering

CSE

CSE in 7 seconds

BMS College of Engineering in 10 seconds



The image shows a Windows desktop environment with two windows open. The left window is a Windows PowerShell terminal, and the right window is a code editor displaying Java code.

**Windows PowerShell Window:**

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\manvi\OneDrive\Desktop\threads> javac Multi.java
PS C:\Users\manvi\OneDrive\Desktop\threads> java Multi
NewThread : Thread[#29,Collg,5,main]
bmsce
bmsce
bmsce
cse
cse
bmsce
bmsce
cse
child exiting
cse
cse
Main thread exiting
PS C:\Users\manvi\OneDrive\Desktop\threads>
```

**Code Editor Window:**

```
File Edit View

class NewThread implements Runnable {
    Thread t;

    NewThread() {
        t = new Thread(this, "Collg");
        System.out.println("NewThread : " + t);
        t.start();
    }

    public void run() {
        try {
            for (int i = 5; i > 0; i--) {
                System.out.println("bmsce");
                Thread.sleep(1000); // Changed sleep time to 1000 milliseconds for faster demonstration
            }
        } catch (InterruptedException e) {
            System.out.println("cse interrupted");
        }
        System.out.println("child exiting");
    }
}

class Multi {
    public static void main(String args[]) {
        new NewThread();
        try {
            Thread.sleep(2000);
            for (int i = 5; i > 0; i--) {
                System.out.println("cse");
                Thread.sleep(1000); // Changed sleep time to 1000 milliseconds for faster demonstration
            }
        } catch (InterruptedException e) {
            System.out.println("Main Thread interrupted");
        }
        System.out.println("Main thread exiting");
    }
}
```

The taskbar at the bottom shows the system clock as 23:12 on 20-02-2024, with a temperature of 25°C and weather 'Mostly clear'.



8. 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 `ArithmeticException`. Display the exception in a message dialog box.

9. Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 & Num2. The division of Num1 & 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 `ArithmeticException`. Display the exception in a message dialog box.

```
import java.awt.*;
```

```
import java.awt.event.*;
```

```
public class IntegerDivisionAWT extends Frame
implements ActionListener
{
```

```
    private TextField num1Field, num2Field, resultField;
    private Button divideButton;
```

```
    public IntegerDivisionAWT() {
```

```
        setTitle("Integer Division");
```

```
        setSize(300, 200);
```

```
        setLayout(new FlowLayout());
```

```
        addWindowListener(new WindowAdapter() {
```

```
            public void windowClosing(WindowEvent e)
```

```
            { dispose(); }
```

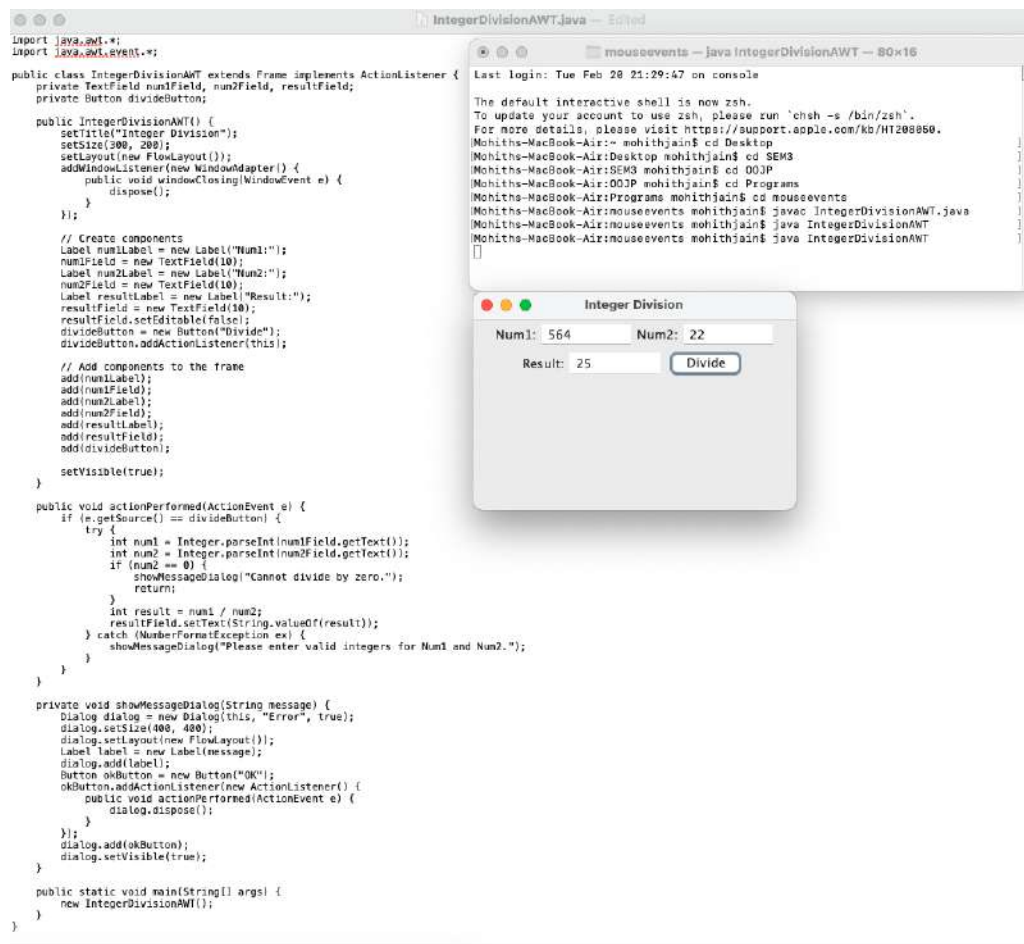
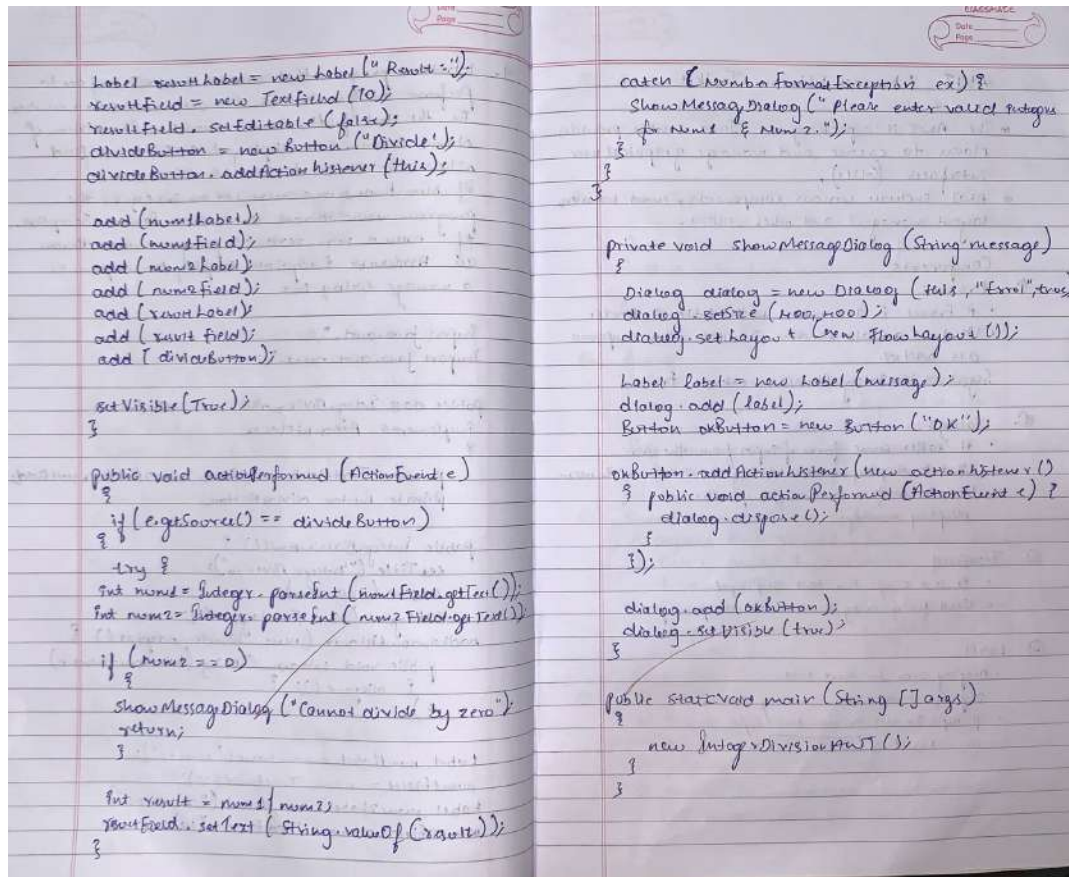
```
        });
```

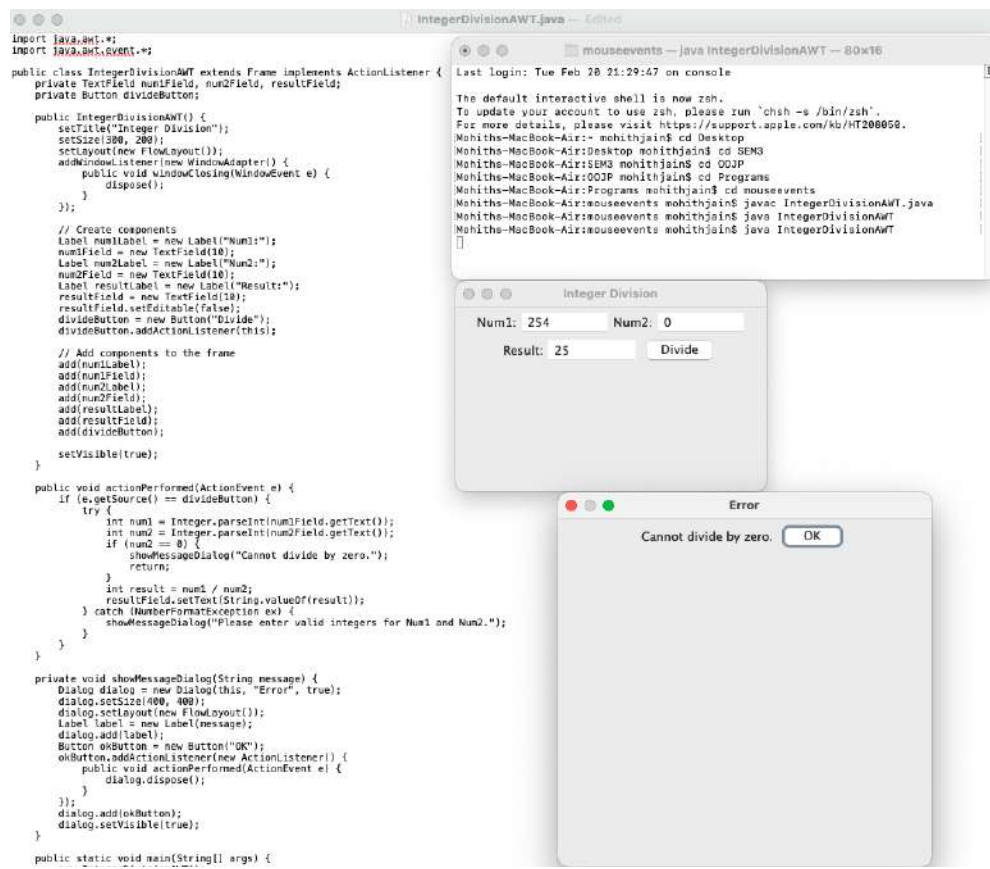
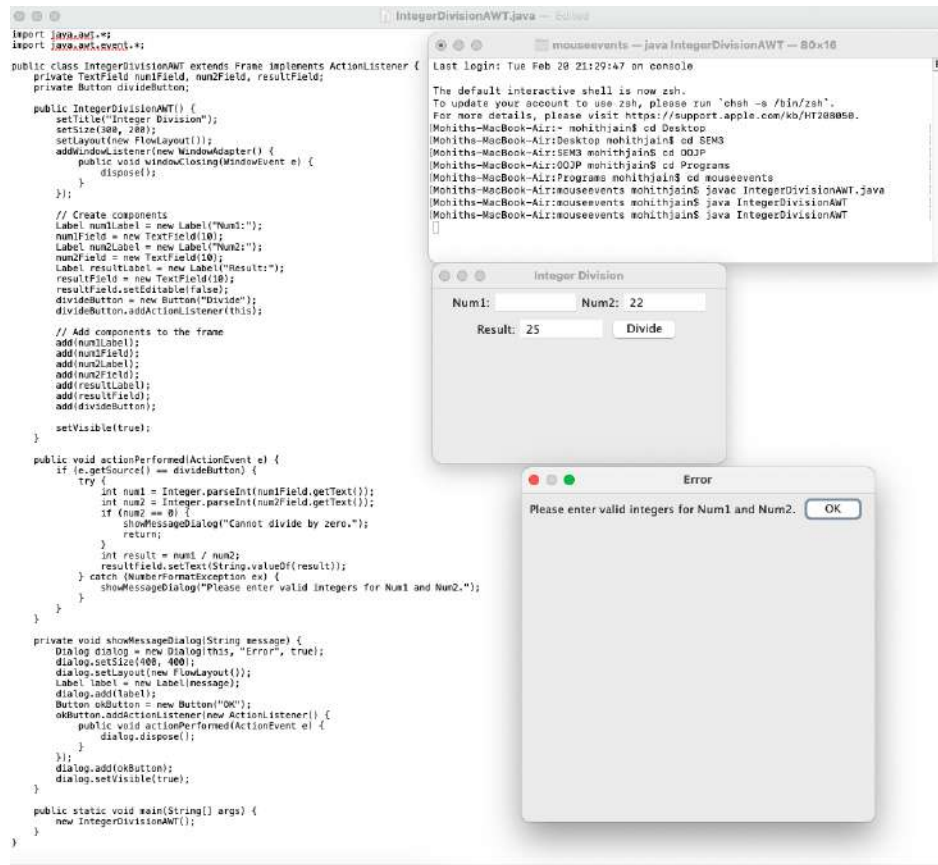
```
        Label num1Label = new Label("Num1:");
```

```
        num1Field = new TextField(10);
```

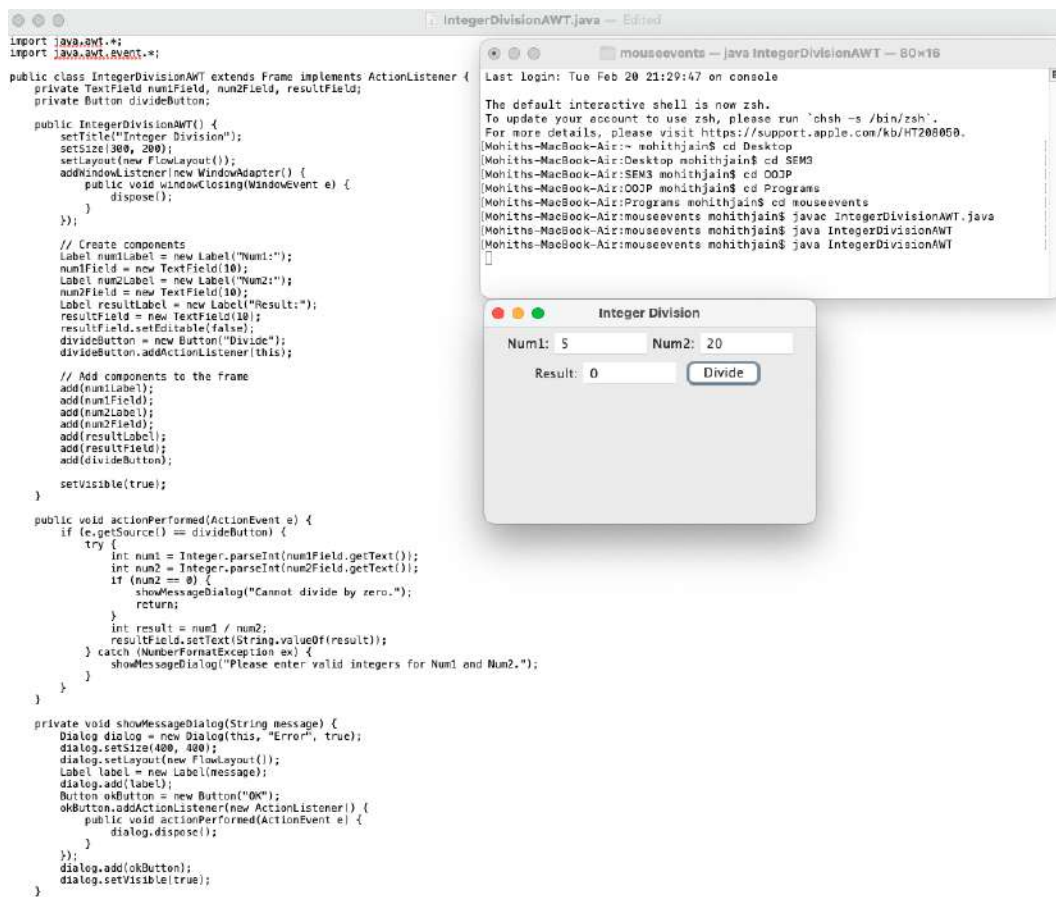
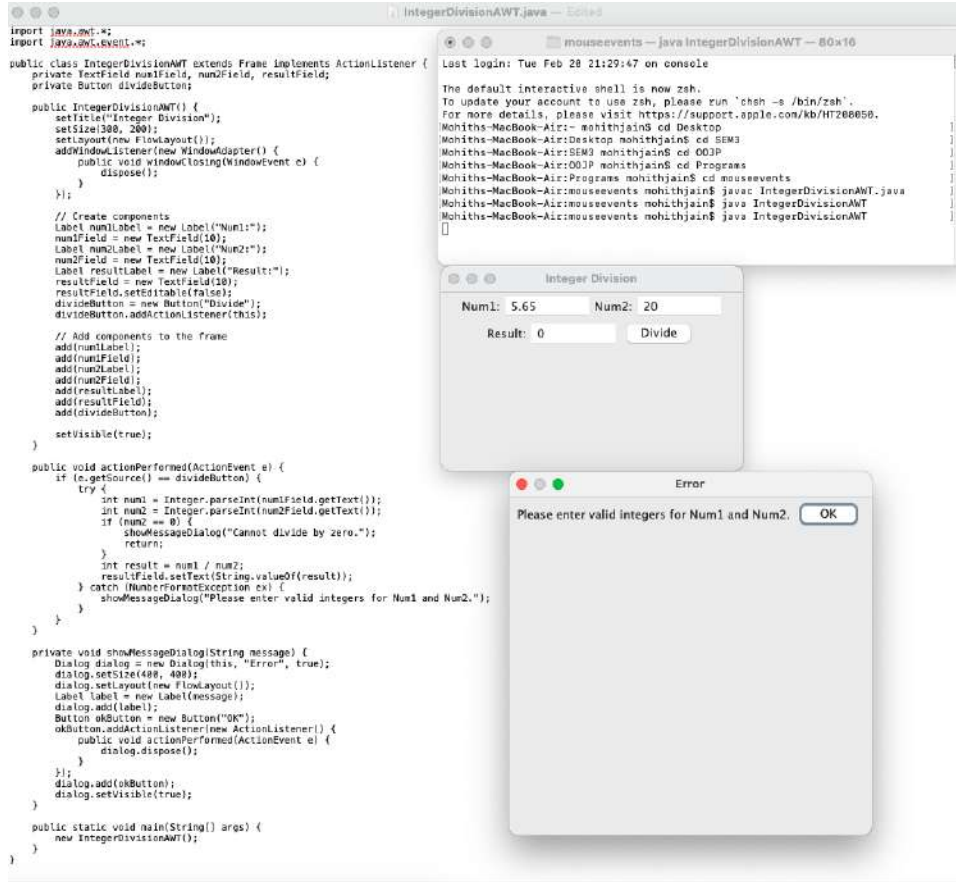
```
        Label num2Label = new Label("Num2:");
```

```
        num2Field = new TextField(10);
```











**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 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.**

CIE - Student, Internals  
SEE - External (Student)

classmate  
Date 22/1/24  
Page

5. Create a package CIE which has two classes - Student and Internals. The class Student 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 two packages in a file that declares the final marks of n students in a five courses.

```
CIE.java
import java.util.Scanner;
package CIE;
public class student {
    String name;
    String usn;
    int sem;
    public student (String name, String usn, int sem)
    {
        this.name = name;
        this.usn = usn;
        this.sem = sem;
    }

    class Internals extends student {
        int [] InternalMarks;
        public Internals (String name, String usn, int sem,
            int [] InternalMarks)
        {
            super (name, usn, sem);
            this.InternalMarks = InternalMarks;
        }
    }
}
```

SEE.java

```

package SEE;
import CIE.Student;
class Externalss extends CIE.Student {
    int[] Seemarks;

    public Externalss(String name, String usn,
        int sem, int[] Seemarks)
    {
        super(name, usn, sem);
        this.Seemarks = Seemarks;
    }
}

```

New file

```

public class Findmarks {
    public static void main (String [] args)
    {
        Scanner s1 = new Scanner(System.in);
        // int n = nextInt();
        System.out.println("Enter no. of students");
        int n = s1.nextInt();

        String[] name = new String[n];
        String[] usn = new String[n];
        int[] sem = new int[n];

        int[][] Internalmarks = new int[n][5];
        int[][] Seemarks = new int[n][5];
    }
}

```

```

for (int i=0; i<5; i++) {
    System.out.println("Enter details of student"
        + (i+1) + ":");
    S.o.p("Name:");
    name[i] = s1.next();
    S.o.p("Usn:");
    usn[i] = s1.next();
    S.o.p("Sem:");
    sem[i] = s1.nextInt();
}

```

```

S.o.p("Enter internal marks for 5 courses:");
for (int j=0; j<5; j++) {
    S.o.p("Course " + (j+1) + ":");
    Internalmarks[i][j] = s1.nextInt();
}
S.o.p("Enter Seemarks for 5 courses:");
for (int j=0; j<5; j++) {
    S.o.p("Course " + (j+1) + ":");
    Seemarks[i][j] = s1.nextInt();
}
}

```

Array to store final Marks

```

int[][] finalMarks = new int[n][5];
for (int i=0; i<n; i++) {
    Internalss I1 = new Internalss(name[i], usn[i],
        sem[i], Internalmarks[i]);
}

```

```

Externalss E1 = new Externalss(name[i], usn[i],
    sem[i], Seemarks[i]);

```

```

for (int j=0; j<5; j++) {
    finalMarks[i][j] = I1.Internalmarks[j] +
        E1.Seemarks[j];
}
}

```

```

S.o.p("Final marks for " + n + " Students in
    5 courses:");

```

```

for (int i=0; i<n; i++) {
    S.o.p("marks [i] + ":");
}

```

```

for (int j=0; j<5; j++) {
}

```

```

S.o.p("finalmarks [i][j] + ":");
}

```

```

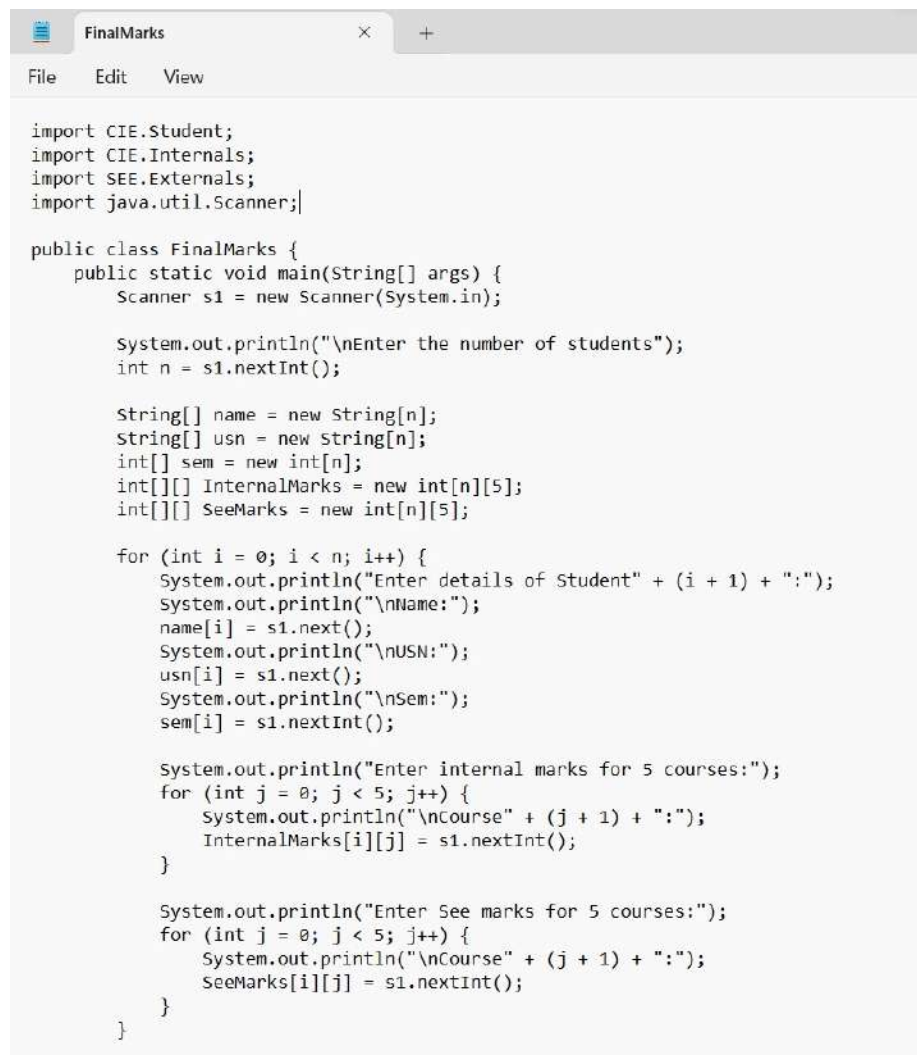
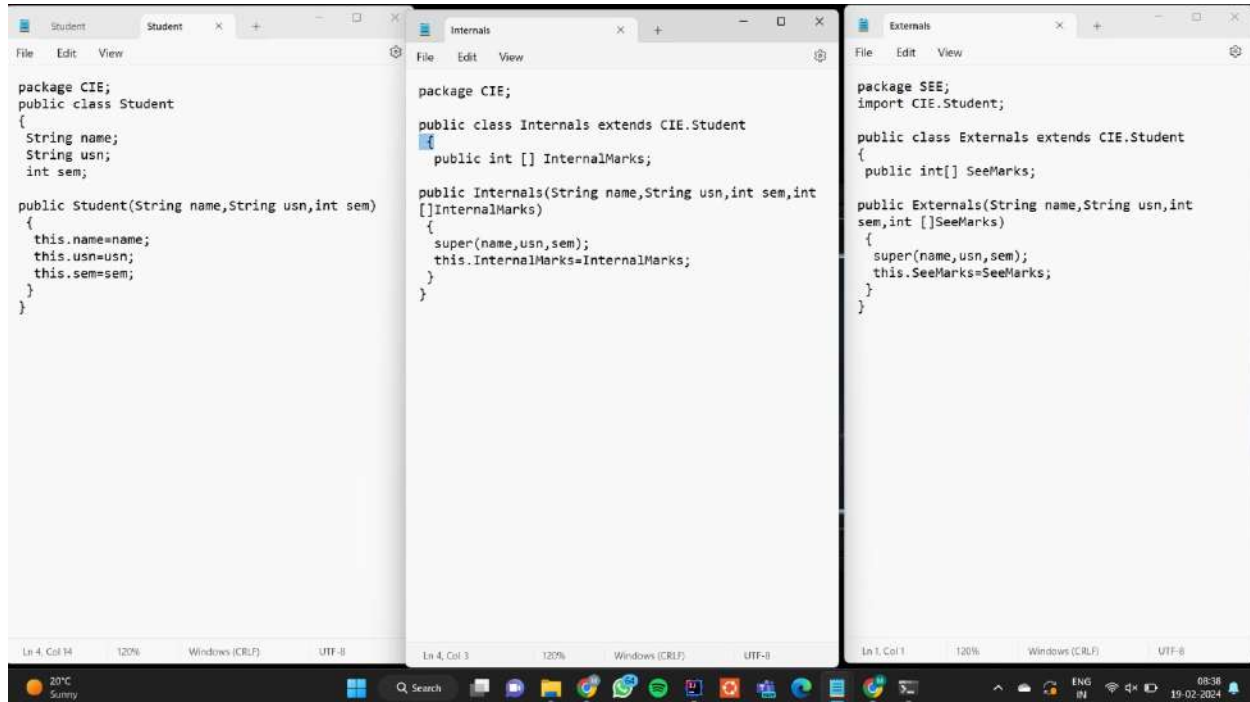
S.o.p();
}

```

```

s1.close();
}
}

```

**Output:**



```

        System.out.println("Enter See marks for 5 courses:");
        for (int j = 0; j < 5; j++) {
            System.out.println("\nCourse" + (j + 1) + ":");
            SeeMarks[i][j] = s1.nextInt();
        }
    }

    int[][] FinalMarks = new int[n][5];
    for (int i = 0; i < n; i++) {
        Internals I1 = new Internals(name[i], usn[i], sem[i], InternalMarks[i]);
        Externals E1 = new Externals(name[i], usn[i], sem[i], SeeMarks[i]);
        int sum=0;
        for (int j = 0; j < 5; j++) {
            FinalMarks[i][j] = I1.InternalMarks[i] + E1.SeeMarks[j];
        }
        System.out.println("\n\nFinal Marks for " + n + " Student in 5 courses:");
        for (int k = 0; k < n; k++) {
            System.out.println(name[k] + ":");
            for (int j = 0; j < 5; j++) {
                sum += FinalMarks[i][j];
            }
            System.out.println(sum);
        }
    }
    s1.close();
}
}

```



pkg — -bash — 80x63

```
External.java:8: error: cannot find symbol
public class External extends Student {
    ^
    symbol: class Student
3 errors
Mohiths-MacBook-Air:StudentMarks mohithjain$
[Restored 20-Feb-2024 at 11:42:57 PM]
Last login: Tue Feb 20 21:59:02 on ttys000
Restored session: Tue Feb 20 22:03:47 IST 2024
```

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Mohiths-MacBook-Air:StudentMarks mohithjain$ clear
```

```
Mohiths-MacBook-Air:StudentMarks mohithjain$ cd
Mohiths-MacBook-Air:~ mohithjain$ cd Desktop
Mohiths-MacBook-Air:Desktop mohithjain$ cd pkg
Mohiths-MacBook-Air:pkg mohithjain$ javac FinalMarks.java
Mohiths-MacBook-Air:pkg mohithjain$ java FinalMarks
```

```
Enter the number of students
2
Enter details of Student1:

Name:
Mohith

USN:
1bm22cs162

Sem:
3
Enter internal marks for 5 courses:
```

```
Course1:
48

Course2:
41

Course3:
43

Course4:
45

Course5:
39
Enter See marks for 5 courses:
```

```
Course1:
96

Course2:
97

Course3:
94

Course4:
89
```

pkg — -bash — 80x63

```
Course5:
92
Enter details of Student2:
```

```
Name:
John

USN:
1bm22cs140
```

```
Sem:
2
Enter internal marks for 5 courses:
```

```
Course1:
49

Course2:
45

Course3:
39

Course4:
41
```

```
Course5:
42
Enter See marks for 5 courses:
```

```
Course1:
89

Course2:
91

Course3:
90

Course4:
97

Course5:
95
```

```
Final Marks for 2 Student in 5 courses:
Mohith:
708
John:
1416
```

```
Final Marks for 2 Student in 5 courses:
Mohith:
687
John:
1374
Mohiths-MacBook-Air:pkg mohithjain$
```

## 10. Abstract Window Toolkit - AWT programs report

### AWT Abstract Window Toolkit

- \* The AWT is a package in Java which provides classes to create and manage graphical user interfaces (GUIs).
- \* AWT includes various components, event handlers, layout managers and other utilities.

#### Components

##### ①. Frame:

- A Frame is a window with title & border.
- Used as a main window in which other components are added.

Syntax: `class IntegerDivision extends Frame`

##### ②. Dialog:

- It takes some form of input from the user.
- Used to display modal dialog to interact with user.
- Used to confirm actions, prompt the user & display messages.

##### ③. TextField:

- It is a single-line text displayed.
- Used for guiding the user to accept input.

##### ④. Label

- Display area for short text.
- Displays static text.
- Prompts the user to enter text / input.
- eg: `Num1 =`



### ⑤. Button:

- ) It triggers the action when clicked
- ) Used for performing specific action
- ) Used for submitting form, confirming, etc..

### ⑥. CheckBox

- ) Component which can be checked or unchecked.
- ) Used to enable or disable
- ) Used in the form with multiple choices.

## Event Handling

### ①. Event:

- ) Events are generated by user actions
- ) Used for handling user input, such as mouse-clicks, key press or window events.
- ) Events are processed by event listeners attached to the relevant components.

### ②. Event Listener:

- ) Objects that receives & handles events.
- ) This is for the implementation of event driven behaviour in GUI's.
- ) Event listeners are registered with specific components to listen for particular types of events.

### ③. Action Event

- ) Event that indicates that a component-defined action occurred.
- ) Used for handling user actions
- ) Action generated when user clicks on components like button, check box, etc..

#### ④. Action listener

- It receives action events.
- It responds to the user actions.

#### ⑤. Window Event

- Event indicating change to the state of a window.
- Handles window-related events such as window opening, closing, resizing etc.

#### ⑥. Window listener

- It receives window events.
- Used to implement methods to handle window-related events.

### Layout Management

#### ①. Flow layout:

- Arranges components in a sequential flow one after another.
- Used for arranging rows or columns with equal spacing between them.
- Used for creating forms etc.

#### ②. Border layout

- Layout manager that divides the container into 5 regions: north, south, east, west & center.
- Used for arranging components at the edges & corner of the container.

#### ③. Layout manager

- Responsible for arranging components in container.
- Controls size & position of components.

\*\*\*\*\*THANK YOU\*\*\*\*\*