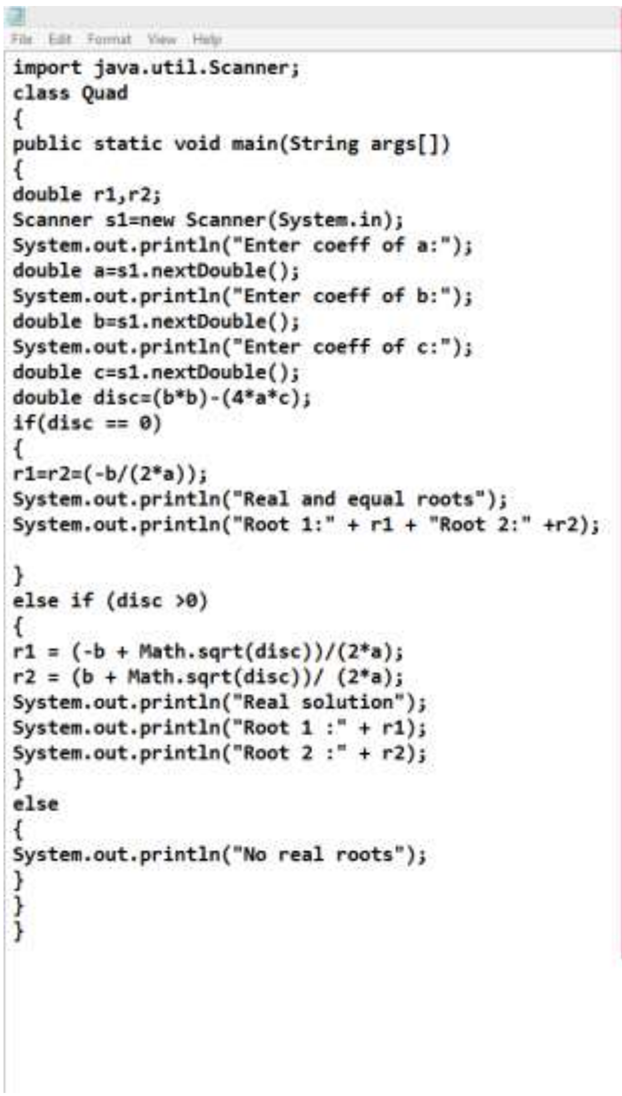
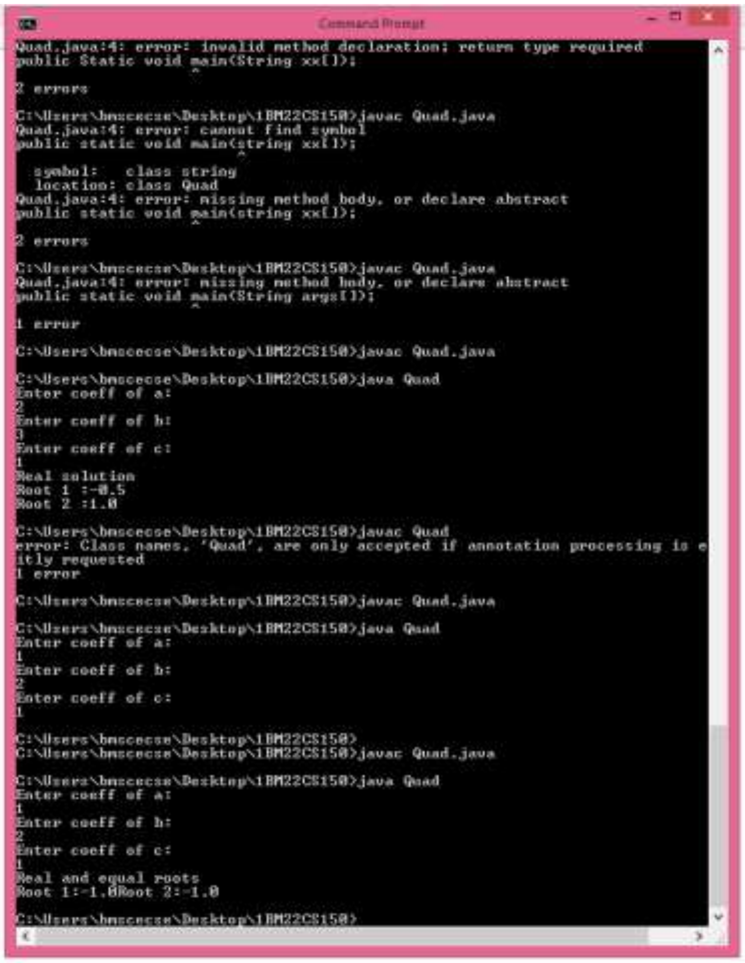
PROGRAM: Develop a Java program that prints all real solutions to the quadratic equation ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2-4ac is negative, display a message stating that there are no real solutions.

INPUT AND OUTPUT->



```java
import java.util.Scanner;
class Quad
{
public static void main(String args[])
{
double r1,r2;
Scanner s1=new Scanner(System.in);
System.out.println("Enter coeff of a:");
double a=s1.nextDouble();
System.out.println("Enter coeff of b:");
double b=s1.nextDouble();
System.out.println("Enter coeff of c:");
double c=s1.nextDouble();
double disc=(b*b)-(4*a*c);
if(disc == 0)
{
r1=r2=(-b/(2*a));
System.out.println("Real and equal roots");
System.out.println("Root 1:" + r1 + "Root 2:" +r2);

}
else if (disc >0)
{
r1 = (-b + Math.sqrt(disc))/(2*a);
r2 = (b + Math.sqrt(disc))/ (2*a);
System.out.println("Real solution");
System.out.println("Root 1 :" + r1);
System.out.println("Root 2 :" + r2);
}
else
{
System.out.println("No real roots");
}
}
}
```



```
Quad.java:4: error: invalid method declaration; return type required
public Static void main(String xx[]);
                   ^
2 errors

C:\Users\bmscecse\Desktop\1BM22CS150>javac Quad.java
Quad.java:4: error: cannot find symbol
public static void main(string xx[]);
                        ^
  symbol:   class string
  location: class Quad
Quad.java:4: error: missing method body, or declare abstract
public static void main(string xx[]);
                   ^
2 errors

C:\Users\bmscecse\Desktop\1BM22CS150>javac Quad.java
Quad.java:4: error: missing method body, or declare abstract
public static void main(String args[]);
                   ^
1 error

C:\Users\bmscecse\Desktop\1BM22CS150>javac Quad.java

C:\Users\bmscecse\Desktop\1BM22CS150>java Quad
Enter coeff of a:
2
Enter coeff of b:
3
Enter coeff of c:
1
Real solution
Root 1 :-0.5
Root 2 :1.0

C:\Users\bmscecse\Desktop\1BM22CS150>javac Quad
error: Class names, 'Quad', are only accepted if annotation processing is e
itly requested
1 error

C:\Users\bmscecse\Desktop\1BM22CS150>javac Quad.java

C:\Users\bmscecse\Desktop\1BM22CS150>java Quad
Enter coeff of a:
1
Enter coeff of b:
2
Enter coeff of c:
1

C:\Users\bmscecse\Desktop\1BM22CS150>
C:\Users\bmscecse\Desktop\1BM22CS150>javac Quad.java

C:\Users\bmscecse\Desktop\1BM22CS150>java Quad
Enter coeff of a:
1
Enter coeff of b:
2
Enter coeff of c:
1
Real and equal roots
Root 1:-1.0Root 2:-1.0

C:\Users\bmscecse\Desktop\1BM22CS150>
```

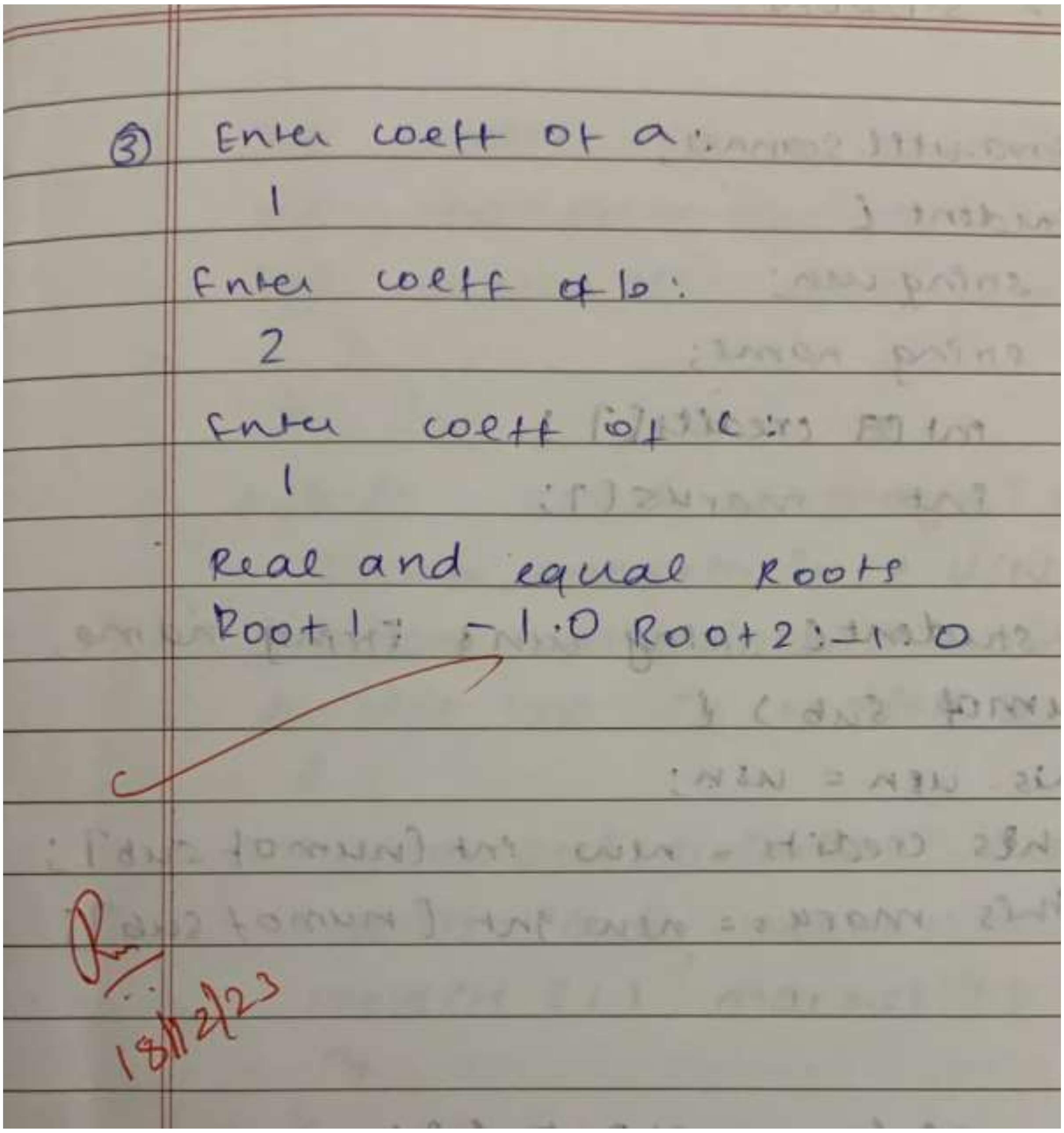
OBSERVATION->

2/1/23 PROGRAM1: QUADRATIC EQUATIONS.

```
import java.util.Scanner;
class Quad
{
public static void main (string args[])
{
double r1, r2;
scanner s1 = new. scanner (system.in);
system.out.printen ("Enter coeff of a:");
double a= s1.nextDouble ();
system.out.printen("Enter coeff of b:");
double b = s1.next Double ();
System.out.printen (" Enter coeff of c: ");
double c= s1. next Double();
double disc = (b*b) - (4*a*c);
If (disc < 0)
{
system.out.println ("No real roots ");
}
else if (disc > 0)
{
r1 = (-b+ Math.sqrt(disc)) / (2*a);
r2 =   (b + Math.sqrt(disc)) /(2*a);
system.out. println("Real solution ");
system. out.println ("Root 1: " + r1);
system. out.printen ("Root2: " + r2);
}
else
{
r1 = r2 = (-b/(2*a));
System.out.println("Real and equal");
System.out.println("Root 1: " + r1 +
"Root 2: " + r2);
}
}
}
```

output

```
① No Real Roots
  Root 1 : -0.5 + i 0.86602540
  Root 2 = -0.5 - i 0.86602540
② Enter coeff of a:
  2
  Enter coeff of b:
  3
  Enter coeff of c:
  1
  Real solution
  Root 1 : -0.6
  Root 2 : 1.0
```



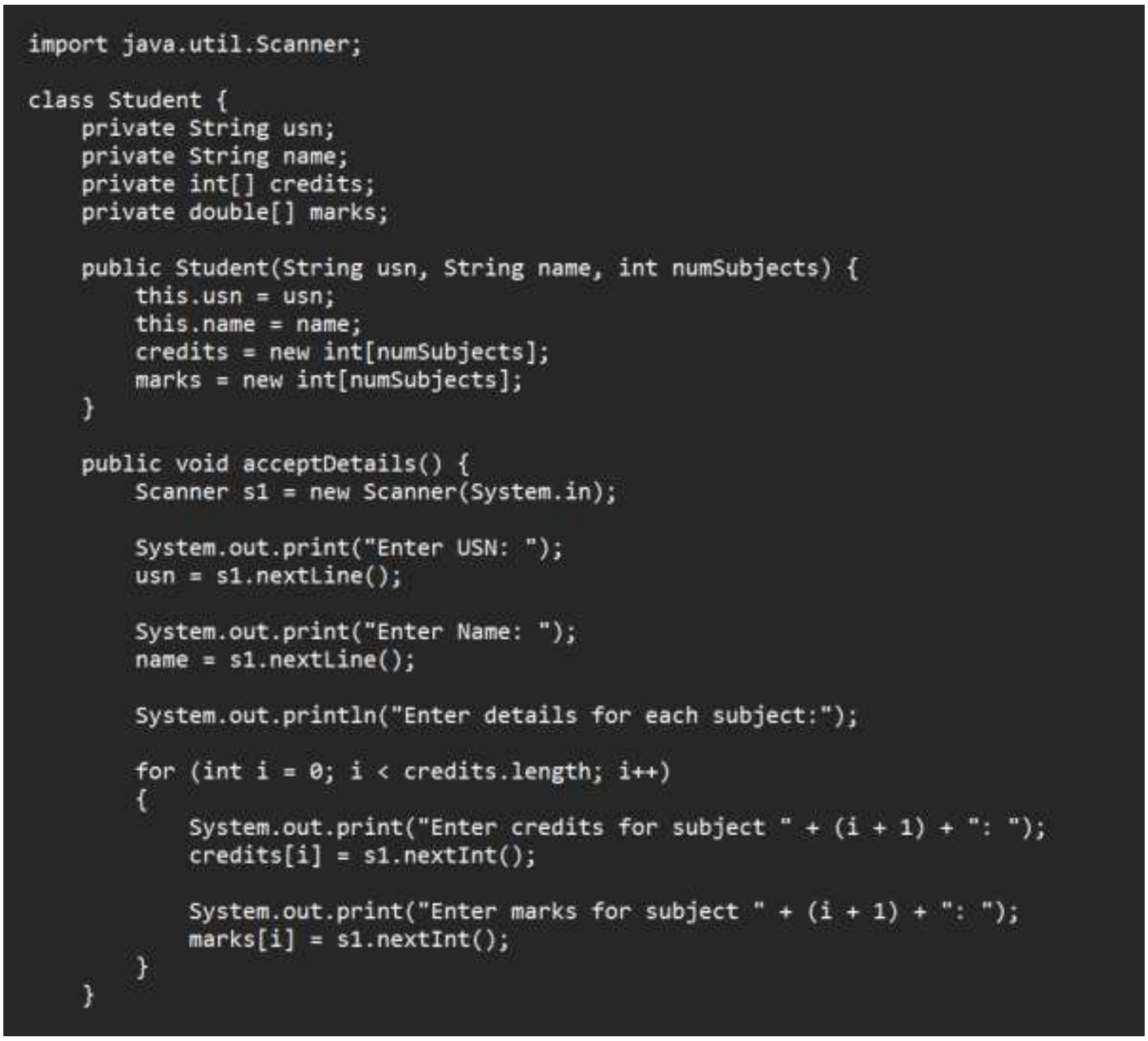
③
```
Enter coeff of a:
 1
Enter coeff of b:
 2
Enter coeff of c:
 1
Real and equal roots
Root1: -1.0 Root2:-1.0
```

[illegible]
18/12/23

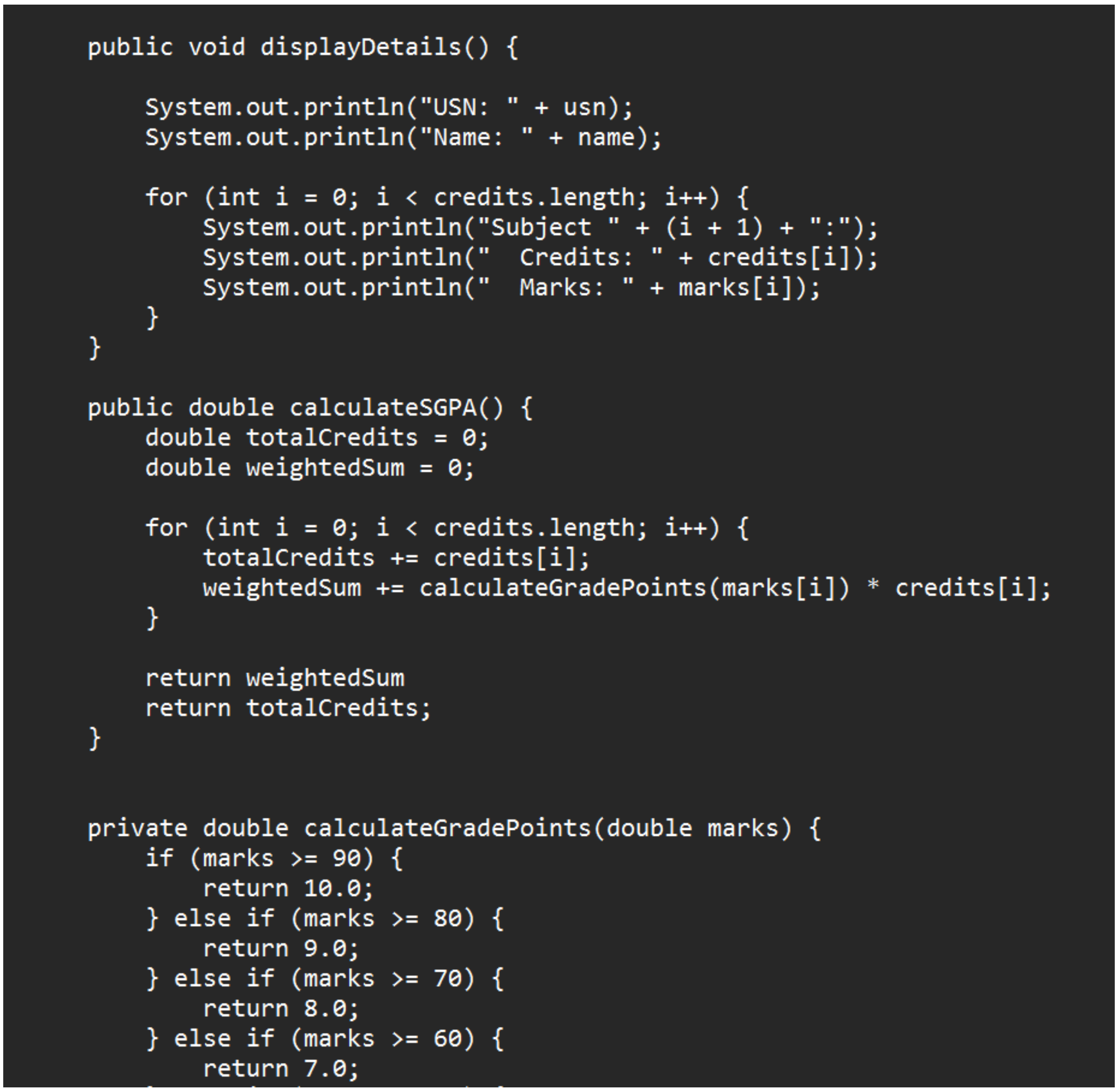
PROGRAM 2:

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.

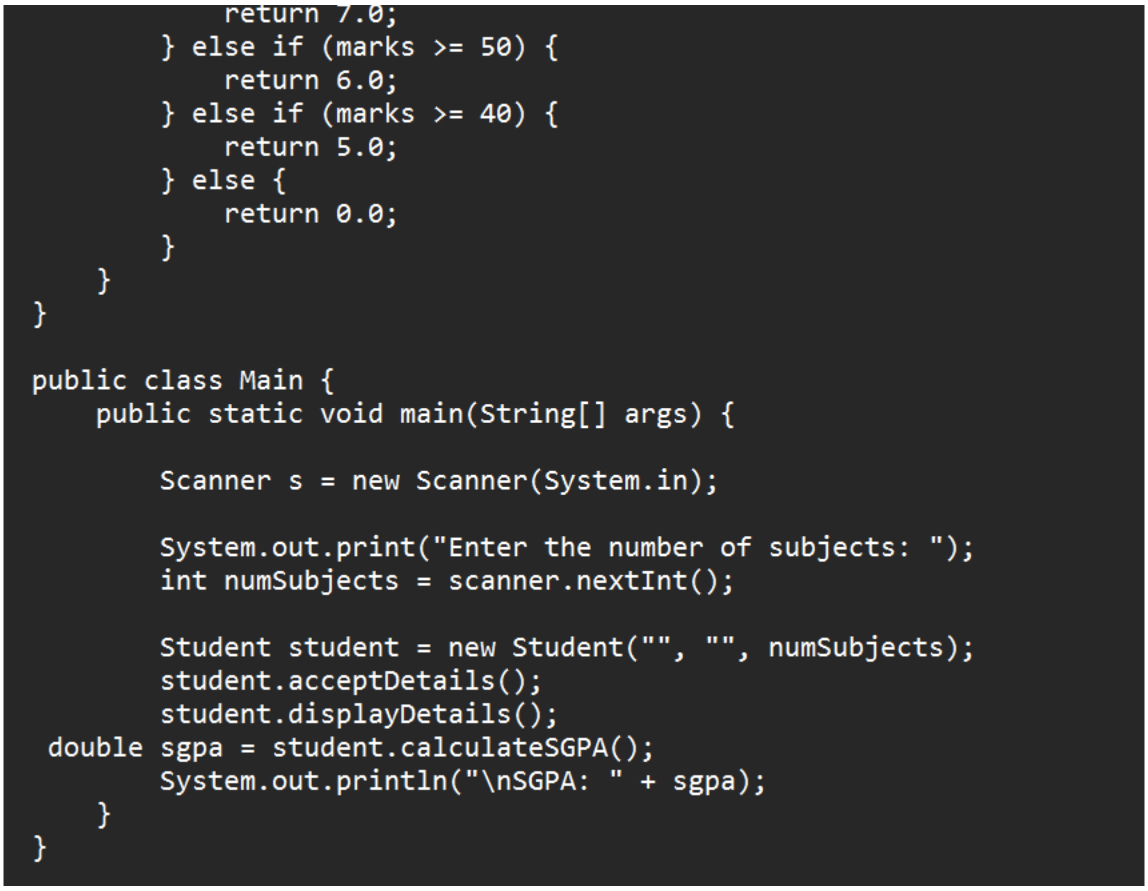
INPUT->



```
import java.util.Scanner;

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

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

    public void acceptDetails() {
        Scanner s1 = new Scanner(System.in);

        System.out.print("Enter USN: ");
        usn = s1.nextLine();

        System.out.print("Enter Name: ");
        name = s1.nextLine();

        System.out.println("Enter details for each subject:");

        for (int i = 0; i < credits.length; i++)
        {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = s1.nextInt();

            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = s1.nextInt();
        }
    }
```

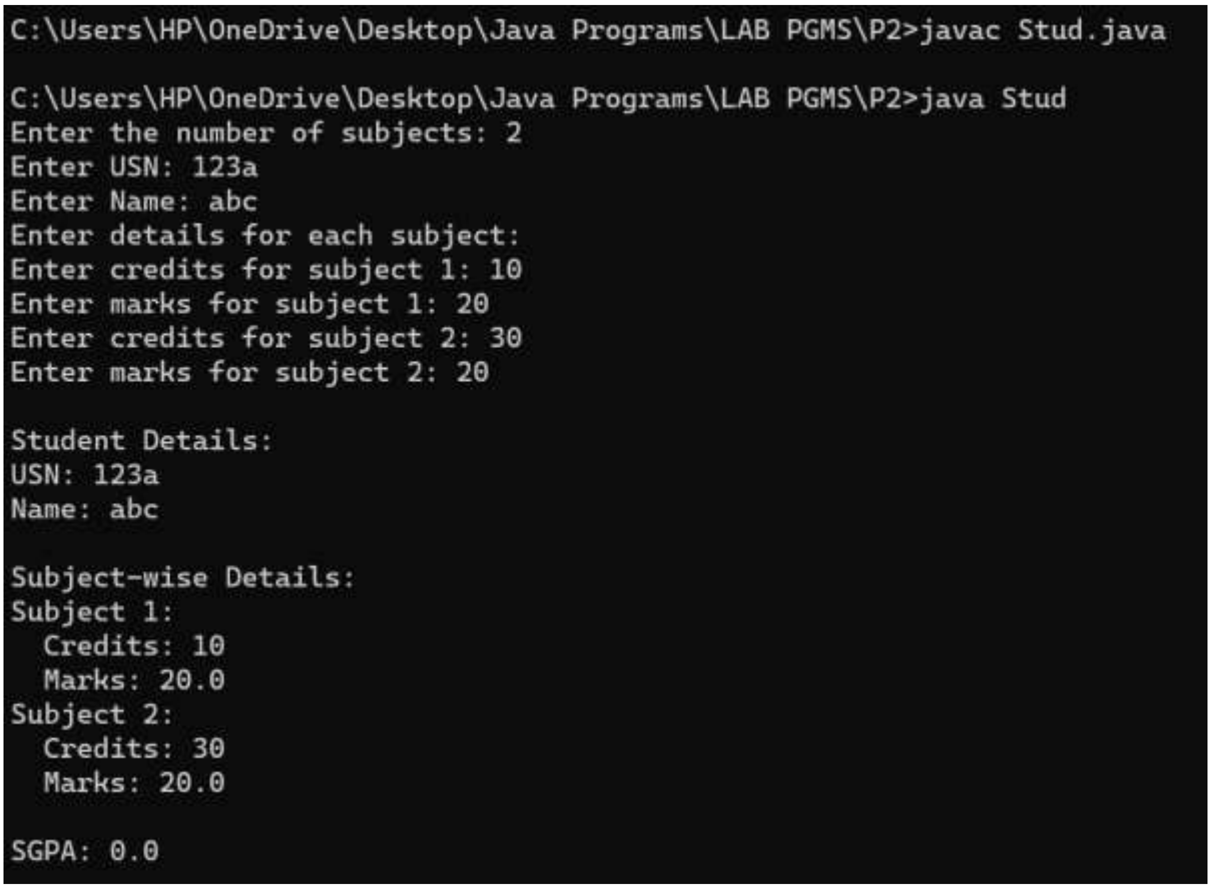


```java
    public void displayDetails() {

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

        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + ":");
            System.out.println("  Credits: " + credits[i]);
            System.out.println("  Marks: " + marks[i]);
        }
    }

    public double calculateSGPA() {
        double totalCredits = 0;
        double weightedSum = 0;

        for (int i = 0; i < credits.length; i++) {
            totalCredits += credits[i];
            weightedSum += calculateGradePoints(marks[i]) * credits[i];
        }

        return weightedSum
        return totalCredits;
    }


    private double calculateGradePoints(double marks) {
        if (marks >= 90) {
            return 10.0;
        } else if (marks >= 80) {
            return 9.0;
        } else if (marks >= 70) {
            return 8.0;
        } else if (marks >= 60) {
            return 7.0;
```



```java
            return 7.0;
        } else if (marks >= 50) {
            return 6.0;
        } else if (marks >= 40) {
            return 5.0;
        } else {
            return 0.0;
        }
    }
}

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

        Scanner s = new Scanner(System.in);

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

        Student student = new Student("", "", numSubjects);
        student.acceptDetails();
        student.displayDetails();
 double sgpa = student.calculateSGPA();
        System.out.println("\nSGPA: " + sgpa);
    }
}
```

OUTPUT->



```
C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P2>javac Stud.java

C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P2>java Stud
Enter the number of subjects: 2
Enter USN: 123a
Enter Name: abc
Enter details for each subject:
Enter credits for subject 1: 10
Enter marks for subject 1: 20
Enter credits for subject 2: 30
Enter marks for subject 2: 20

Student Details:
USN: 123a
Name: abc

Subject-wise Details:
Subject 1:
  Credits: 10
  Marks: 20.0
Subject 2:
  Credits: 30
  Marks: 20.0

SGPA: 0.0
```

OBSERVATION->

01/24 PROGRAM 2: STUDENT

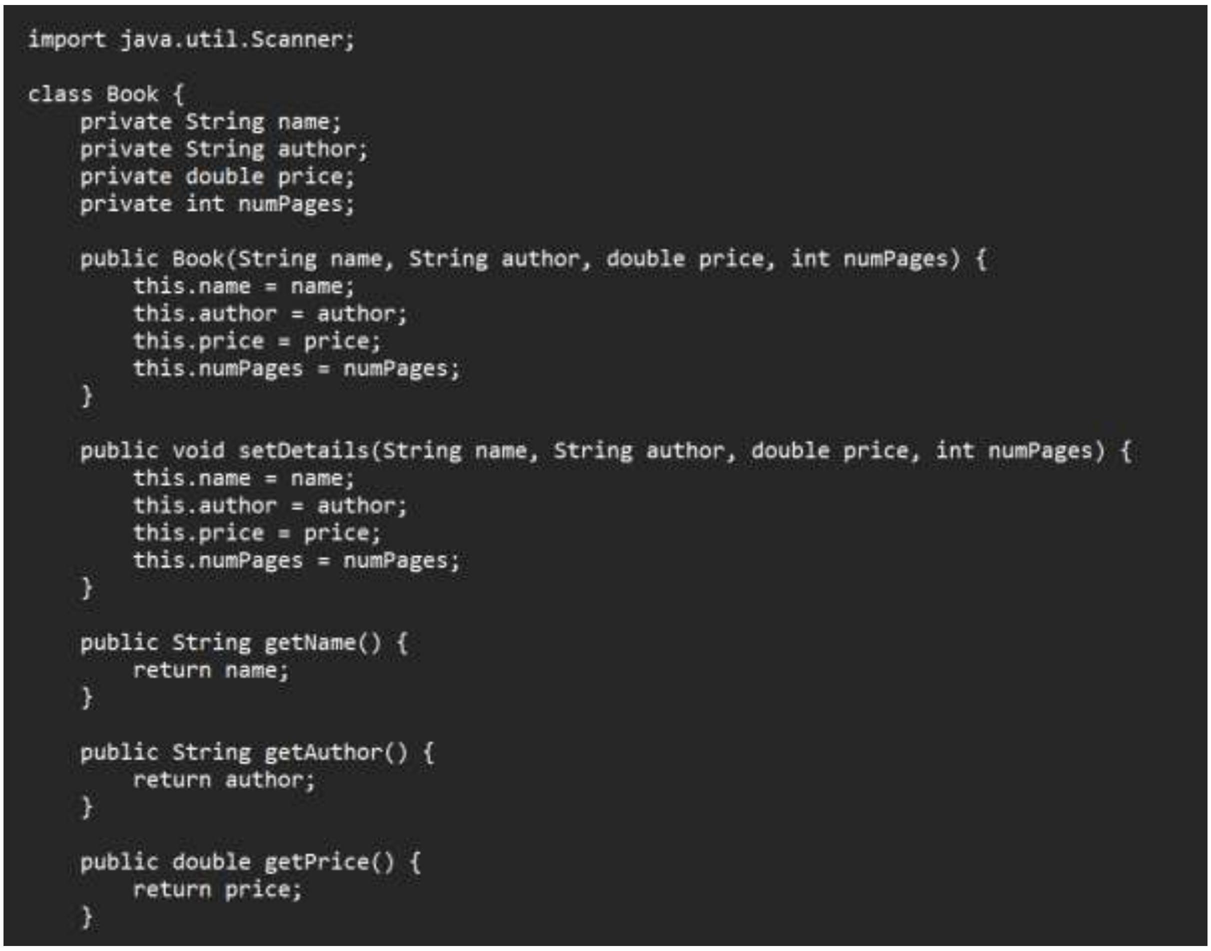
```
import java.util.scanner;
class student {
private string usn;
private string name;
private int credits[] ;
private int marks[];

public student( string usn, string name,
int numof sub ) {
        this. usn = usn;
         this. credits = new int [numof sub];
         this. marks= new int [numof sub];
       }

public void acceptDet (){
  scanner s1 = new scanner(system.in);
  system.out. print ("Enter USN:");
     ~~int credits [~~
           ~~credits~~ usn = ~~scanner~~ s1.nextLine();
    system.out. print (" Enter name ");
    name = s1 scanner.nextLine();
for (int i= 0 ; i< credits.length ; i++
   {
system.out.println(" Enter credits");
int credits [i] = ~~newscanner~~ s1. ~~nextIter~~ nextInt();
~~int ma~~ System.out.println("Enter marks:");
int marks [i] = ~~Scanners~~ s1.nextInt ();
  }
 }
public void display() {
 System.out.println(" USN: " + usn);
 System.out.println(" Name: " + name);
for (int i=0; i< credits.length; i++)
 {
 System.out.printf(" subject %d ,
credits: %d , Marks: %d ", i+1 ;
 credits [i], marks[i]) ;
  }
 }
public double calculate-SGPA(){
 int  total = 0;
 double  sum = 0.0;
for (int i=0; i<credits.length; i++){
        total+= credits [i];
        sum += calcGP (marks[i] *
                credits [i] ; }
        return sum;
        return total;
   }
```

```
private double calcGP() {
    ~~fn~~ if (marks >= 90) {
        return 10.0;
        }
      else if ( marks >= 80) {
        return 9.0;
        }
      else if (marks >= 70) {
        return 8.0;
        }
      else if (marks >= 60) {
          return 7.0;
        }
     else if  (marks >= 50) {
          return 6.0;
        }
    else {
        return 0.0;
        }
    };
  }
  class Stud {
public static void main (String args[])
 {
```

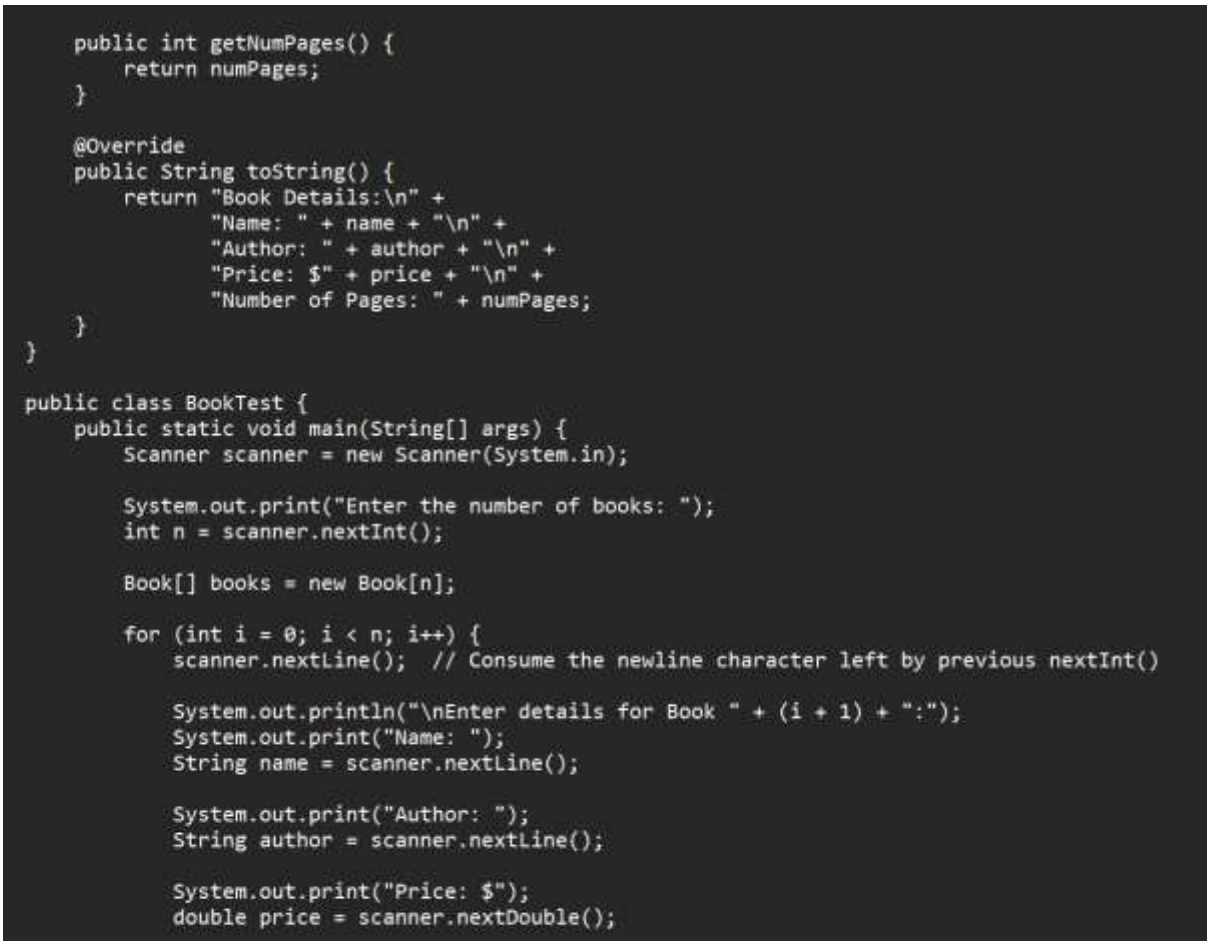
```
student student = new student("123",
"John Doe", 3);
   student.acceptDet();
   student.display();
  system.out.println("SGPA: "+ student.
   calculateSGPA());
   }
 }

OUTPUT :
Enter USN : 112
Enter Name : abc
Enter details for each subject:
 Enter credits for subject 1:
 10
Enter marks for subject 1: 90
Enter credits for subject 2: 9
                         2: 85
                         3: 8
                         3: 76
student Details :
 USN : 112
 Name : abc
 subject-wise details :
sub1 - credits - 10   Marks - 90
sub2 - credits - 9    Marks - 85
sub3 - credits - 8    Marks - 76
```

PROGRAM: 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.

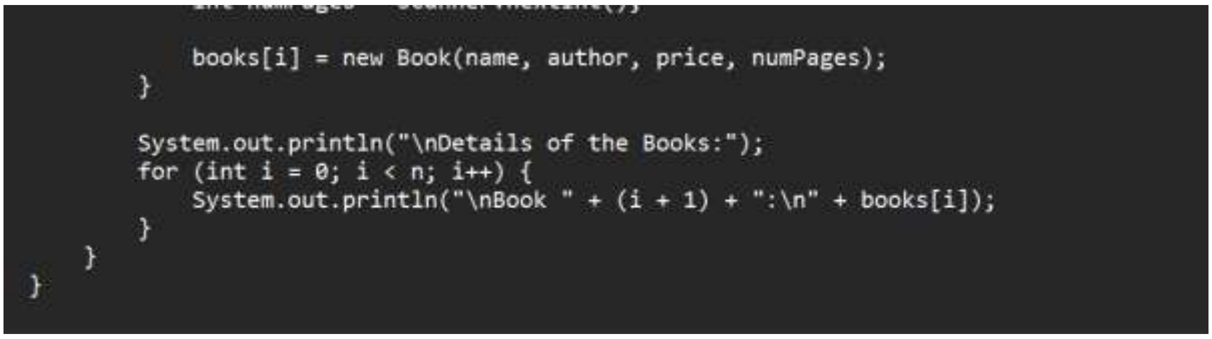
INPUT->



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

    public String getName() {
        return name;
    }

    public String getAuthor() {
        return author;
    }

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

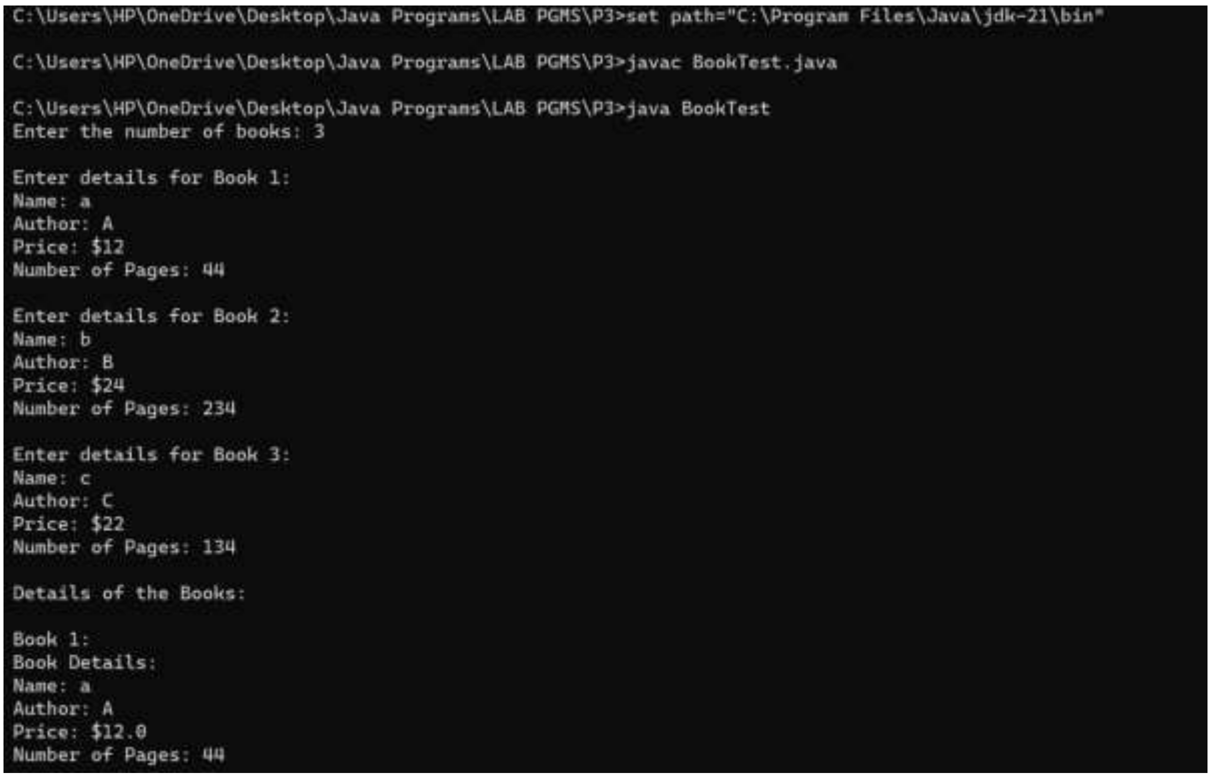


```java
    public int getNumPages() {
        return numPages;
    }

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

public class BookTest {
    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++) {
            scanner.nextLine();  // Consume the newline character left by previous nextInt()

            System.out.println("\nEnter details for Book " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = scanner.nextLine();

            System.out.print("Author: ");
            String author = scanner.nextLine();

            System.out.print("Price: $");
            double price = scanner.nextDouble();
```

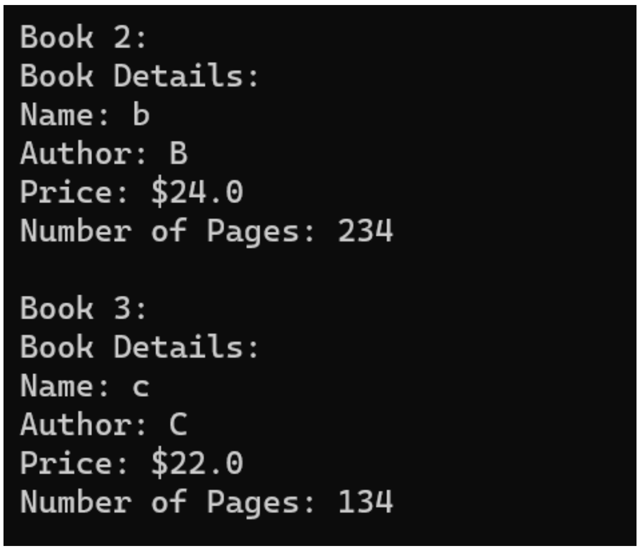


```java
            books[i] = new Book(name, author, price, numPages);
        }

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

OUTPUT->



```
C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P3>set path="C:\Program Files\Java\jdk-21\bin"

C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P3>javac BookTest.java

C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P3>java BookTest
Enter the number of books: 3

Enter details for Book 1:
Name: a
Author: A
Price: $12
Number of Pages: 44

Enter details for Book 2:
Name: b
Author: B
Price: $24
Number of Pages: 234

Enter details for Book 3:
Name: c
Author: C
Price: $22
Number of Pages: 134

Details of the Books:

Book 1:
Book Details:
Name: a
Author: A
Price: $12.0
Number of Pages: 44
```



```
Book 2:
Book Details:
Name: b
Author: B
Price: $24.0
Number of Pages: 234

Book 3:
Book Details:
Name: c
Author: C
Price: $22.0
Number of Pages: 134
```

OBSERVATION->

01/24 PROGRAM 3 : BOOK

```
import java.util.scanner;
 class Book {
   private String name;
   private String author;
   private ~~String~~ double price;
   private int num;

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

public void setDet (String name,
 String author, double price,
 int num) {
   this name
   this author
  this price
  this num
   }
```

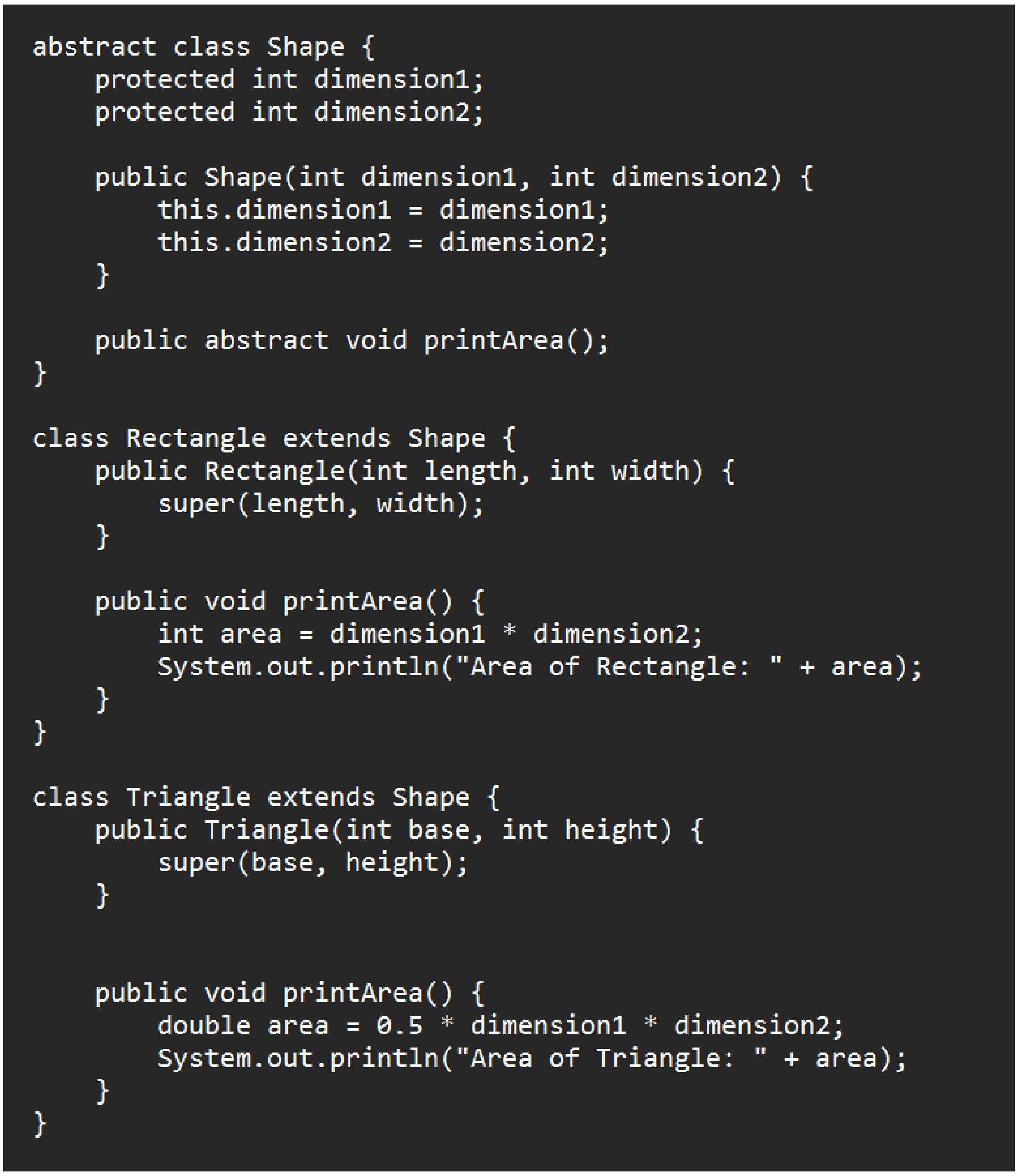
```
public String getDet () {
    return " Name:"+name+"\n author:"
+ author + "\n Price: $"+ price +
" \n Number of pages." + numpages;
  }

public string toString () {
    return getDet ();
  }
}

public class BookTest {
public static void main (String args[])
{
Scanner s1 = new Scanner(System.in);
System.out.println("Enter the number
of books:");
int n = sc. nextInt();
Book []b = new Book [n];
for (int i = 0; i < n; i++){
System.out.println("Enter the number
of books:");
int n = sc. nextInt())
Book []b = new Book [n];
for (int i=0 ; i<n; i++ ) {
System. out.println(" Enter the
details for book" + (i+1)+ " : " );
System.out.println(" Name:");
 String name= sc.next();
system.out.println("Author:");
 String author= sc.next();
 System. out. println ("price: $" );
 double price = sc.nextDouble;
System.out.println ("Number of Pages");
 int num = sc.nextInt();
b[i]= new Book(name, author,
price, num);
}
 System.out.println("\n Details of
books :");
 for(int i=0; i<n; (i++){
  System.out.println("\n book."
 +(i+1) + ": \n"+ b[i]);
 }
 }
}
```

```
OUTPUT :
Enter number of books : 2
Enter details of book 1 :
Name: abc
Author: aaa
Price: $ 19
Number of pages : 9
Enter details of book 2:
Name : qwe
Author: yyy
Price : $ 87
Number of pages : 30
Details of books:
books1 :
Name : abc
Author : aaa
Price : 19
Number of pages : 9
book 2:
Name : qwe
Author : yyy
Price : 87
Number of page : 30
```

8/11/2024 [illegible]

PROGRAM: 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.

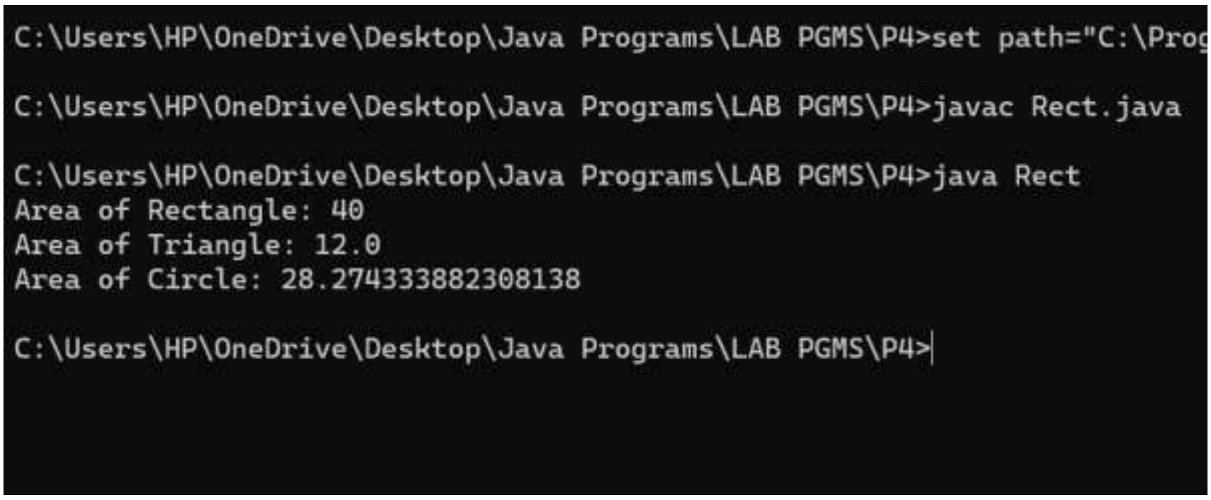
INPUT->



```java
abstract class Shape {
    protected int dimension1;
    protected int dimension2;

    public Shape(int dimension1, int dimension2) {
        this.dimension1 = dimension1;
        this.dimension2 = dimension2;
    }

    public abstract void printArea();
}

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

    public void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }


    public void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Area of Triangle: " + area);
    }
}
```

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


    public void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: " + area);
    }
}

public class Rect {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(5, 8);
        Triangle triangle = new Triangle(4, 6);
        Circle circle = new Circle(3);

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

OUTPUT->



```
C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P4>set path="C:\Prog

C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P4>javac Rect.java

C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P4>java Rect
Area of Rectangle: 40
Area of Triangle: 12.0
Area of Circle: 28.274333882308138

C:\Users\HP\OneDrive\Desktop\Java Programs\LAB PGMS\P4>
```

OBSERVATION->

4 PROGRAM4: ABSTRACT SHAPE

```
abstract class shape {
      protected int dimension1;
      protected int dimension2;

public shape (int dimension1, int dimension2)
 {   this. dimension1= dimension1;
     this. dimension2 = dimension2;
   }

public abstract void printArea ( );
 }

class Rect extends shape {
public Rect (int length, int width) {
      super (length, width);
      }
public void printArea ( ) {
       int area = dimension1 * dimension2;
system.out.println(" Area of Rectangle "
   + area);
   }
}
```

```
class Triangle extends Shape {
  public Triangle( int base, int height ){
      super (base, height );
  }
public void ~~Area~~ printArea (){
 double area = 0.5 * dimension1 *
                     dimension 2;
  System.out.println ("Area of Triangle: "
   + area);
  }
}
  class  Circle extends Shape {
   public Circle (int radius) {
      super(radius, 0 );
     }
 public void printArea () {
 double area = Math.PI * dimension1
  * dimension 1 ;
 system.out.println (" Area Of Circle:"
 + area);
  }
}
```

```
public class Rect {
public static void main (String args[])
 {
 Rect. rect = new rect ( 8, 9);
 Triangle triangle=new triangle (2, 6);
 Circle circle = new circle ( 10);

  rect. printArea ();
   triangle. printArea ();
    circle . printArea ();
  }

 }
```

Output

```
Area Of Rectangle : 72
Area Of Triangle : 6. 0
Area of circle : 314. 159265
```

[illegible] 8/1/2024

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:

a) Accept deposit from customer and update the balance.

b) Display the balance.

c) Compute and deposit interest

d) Permit withdrawal and update the balance

Check for the minimum balance, impose penalty if necessary and update the balance.

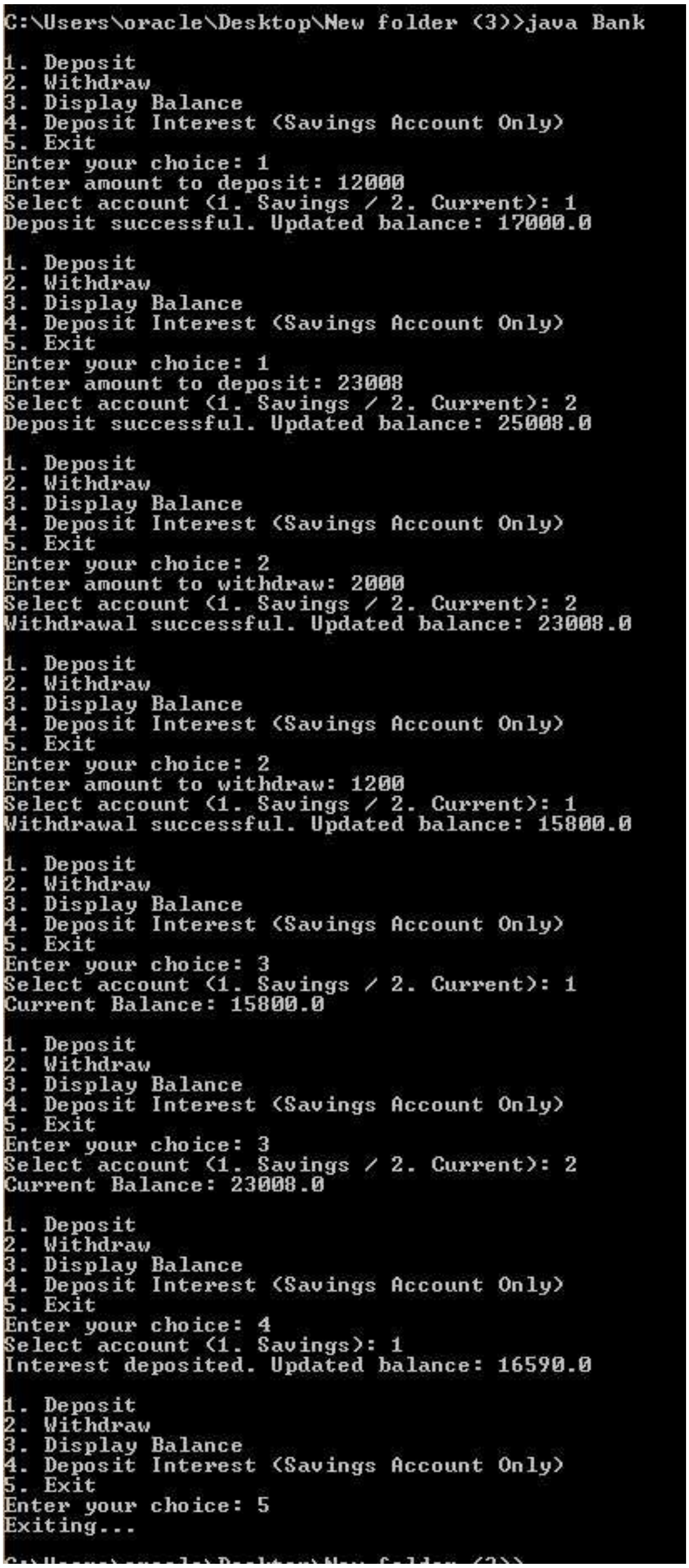
INPUT->

```java
import java.util.Scanner;

// Account class to store customer information
class Account {
   String customerName;
   int accountNumber;
   String accountType;
   double balance;

   // Constructor
   Account(String name, int accNo, String accType, double bal) {
      customerName = name;
      accountNumber = accNo;
      accountType = accType;
      balance = bal;
   }

   // Method to accept deposit
   void deposit(double amount) {
      balance += amount;
      System.out.println("Deposit successful. Updated balance: " + balance);
   }

   // Method to display balance
   void displayBalance() {
      System.out.println("Current Balance: " + balance);
   }
}

// Current Account class
class CurAcct extends Account {
   double minBalance = 1000; // Minimum balance for current account
   double serviceCharge = 50; // Service charge if balance falls below minimum

   // Constructor
   CurAcct(String name, int accNo, String accType, double bal) {
      super(name, accNo, accType, bal);
   }
```

```
    // Method to withdraw
    void withdraw(double amount) {
        if (balance - amount >= minBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. Updated balance: " + balance);
        } else {
            System.out.println("Insufficient balance. Service charge of $" + serviceCharge + " will be applied.");
            balance -= serviceCharge;
            System.out.println("Updated balance after service charge: " + balance);
        }
    }
}

// Savings Account class
class SavAcct extends Account {
    double interestRate = 0.05; // Interest rate for savings account

    // Constructor
    SavAcct(String name, int accNo, String accType, double bal) {
        super(name, accNo, accType, bal);
    }

    // Method to deposit interest
    void depositInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest deposited. Updated balance: " + balance);
    }

    // Method to withdraw
    void withdraw(double amount) {
        if (balance - amount >= 0) {
            balance -= amount;
            System.out.println("Withdrawal successful. Updated balance: " + balance);
        } else {
            System.out.println("Insufficient balance.");
        }
    }
}
```

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

    // Creating a savings account
    SavAcct savingsAccount = new SavAcct("John Doe", 1001, "Savings", 5000);
    // Creating a current account
    CurAcct currentAccount = new CurAcct("Jane Smith", 2001, "Current", 2000);

    // Menu
    int choice;
    do {
      System.out.println("\n1. Deposit");
      System.out.println("2. Withdraw");
      System.out.println("3. Display Balance");
      System.out.println("4. Deposit Interest (Savings Account 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();
          System.out.print("Select account (1. Savings / 2. Current): ");
          int accountType = scanner.nextInt();
          if (accountType == 1)
            savingsAccount.deposit(depositAmount);
          else if (accountType == 2)
            currentAccount.deposit(depositAmount);
          break;
        case 2:
          System.out.print("Enter amount to withdraw: ");
          double withdrawAmount = scanner.nextDouble();
          System.out.print("Select account (1. Savings / 2. Current): ");
          int accountTypeWithdraw = scanner.nextInt();
          if (accountTypeWithdraw == 1)
            savingsAccount.withdraw(withdrawAmount);
          else if (accountTypeWithdraw == 2)
            currentAccount.withdraw(withdrawAmount);
          break;
        case 3:
          System.out.print("Select account (1. Savings / 2. Current): ");
          int accountTypeDisplay = scanner.nextInt();
          if (accountTypeDisplay == 1)
            savingsAccount.displayBalance();
          else if (accountTypeDisplay == 2)
            currentAccount.displayBalance();
          break;
        case 4:
          System.out.print("Select account (1. Savings): ");
          int accountTypeInterest = scanner.nextInt();
          if (accountTypeInterest == 1)
            savingsAccount.depositInterest();
          else
            System.out.println("Invalid option.");
          break;
        case 5:
          System.out.println("Exiting...");
          break;
        default:
          System.out.println("Invalid option. Please try again.");
      }
    } while (choice != 5);

    scanner.close();
  }
}
```

OUTPUT->



```
C:\Users\oracle\Desktop\New folder (3)>java Bank

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 1
Enter amount to deposit: 12000
Select account (1. Savings / 2. Current): 1
Deposit successful. Updated balance: 17000.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 1
Enter amount to deposit: 23008
Select account (1. Savings / 2. Current): 2
Deposit successful. Updated balance: 25008.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 2
Enter amount to withdraw: 2000
Select account (1. Savings / 2. Current): 2
Withdrawal successful. Updated balance: 23008.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 2
Enter amount to withdraw: 1200
Select account (1. Savings / 2. Current): 1
Withdrawal successful. Updated balance: 15800.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 3
Select account (1. Savings / 2. Current): 1
Current Balance: 15800.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 3
Select account (1. Savings / 2. Current): 2
Current Balance: 23008.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 4
Select account (1. Savings): 1
Interest deposited. Updated balance: 16590.0

1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account Only)
5. Exit
Enter your choice: 5
Exiting...

C:\Users\oracle\Desktop\New folder (3)>
```

OBSERVATION->

9/02/24 PROGRAM 13: BANK

```
import java.util.scanner;
class Account {
 String customerName;
 int accountNamber;
 string accountType;
 double balance;

Account(String name; int accNO. String accType,
double bal)
{
  customerName = name;
  accountNumber = accNo;
  account Type  = accType;
   balance = bal;
}

void deposit (double amount)
{
    balance += amount;
 system.out.println("Deposit succesful.
Updated balance: "+ balance);
}

void displayBalance(){
System.out. println ("current balance:" +
                              balance);
                          }
```

```
}
class curAcct extends Account
{
 double minBalance = 1000;
 double serviceCharge = 50;
 curAcct (String name, int accNo,
 String accType, double bal)
 {
    super(name, accNo, acctype, bal);
 }
 void withdraw (double amount)
 {
 if(balance - amount >= minBalance)
 { ~~eff~~ balance -= amount;
 System.out.println("withdrawal
 successful. Updated balance:"+ balance);
  }
 else
 {
 System.out.println("Insufficient
 balance. Service charge of $ " +
 serviceCharge + "will be applied:");
 balance -= service charge;
 System.out.println("Updated balance
after service charge:"+ balance);
 }
}
}

class SavAcct extends Account
{
 double interestRate=0.05;
SavAcct (String name, int accNO, String
 accType, double bal)
  { super(name, accNO, accType, bal); }

void depositInterest()
{
 double interest = balance * interestRate;
 balance += interest;
System.out.println("Interest deposited.
 Updated balance:" +balance);
 }

 void withdraw (double amount)
 { if (balance-amount>=0)
   { balance -=amount;
System.out.println("Withdrawalsuccessful.
Updated balane : " + balance);
}
else
{
  system.out.printen ("Insufficient
  balance");
}
}
}

public class Bank
{ public static void main (String [] args)
 {
  scanner scanner = new scanner (system.in)
savAcct savingAccount = new savAcct ("Joh
Doe", 1001, "savings ", 5000);
curAcct  currentAccount = new curAcct (
"Jane Smith", 2001, "current", 200);

 int choice;
do {
      system.out.printen ("\n 1. Deposit ");
system. out. prenten (" 2. Withdraw");
system. out. printen (" 3. Display Balanc
system.out.printen("4. Deposit Interest
  (savings Account only)");
system.out.print(" Enter your choice: ");
 choice = scanner.nextInt();

switch (choice)
{ case 1:
system.out.printen("Enter amount to
deposit");
double depositAmount = scanner.nextDouble();
System.out.print(" select account (1.
 Savings / 2. Current ) : ");
 int accountType = scanner.nextInt();
     if (accountType = = 1)
savingsAccount.deposit (depositAmount);
 else if (accountType = = 2)
current Account.deposit (depositAmount);
 break;

case 2:
System.out.print(" Enter amount to
withdraw : ");
 double withdrawAmount = scanner.
   nextDouble ();
```

```
system.out.print("Enter amount to
withdraw");
double withdrawAmount = scanner.
nextDouble();
system.out.print("select account (1.Sav
|2.current):");
int accountTypewithdraw=scanner.
nextInt();
if (accountTypewithdraw = =1)
 savingsAccount.withdraw (withdraw
 Amount);
else if(accountTypewithdraw = = 2)
 currentAccount-withdraw (withdraw
 Account);
 break;
 case 3:
system.out.print ("select account
(1.savings |2.current): ");
int accountTypeDisplay =scanner.nextIn
if (account TypeDisplay = = 1)
savingsAccount.displayBalance();
else if (accountTypeDisplay = = 1)
 currentAccount.displayBalance();
break;
```

```
case 4:
System.out.print("select account(1.Savings)
");
int accountTypeInterest = scanner.nextInt();
if (accountTypeInterest == 1)
savingsAccount.depositInterest();
else
system.out.print("Invalid option");
 break;
 default:
 system.out.println("Invalid option.
please try again");
 }
} while(choice != 5);
Scanner.close();
 }
}
```

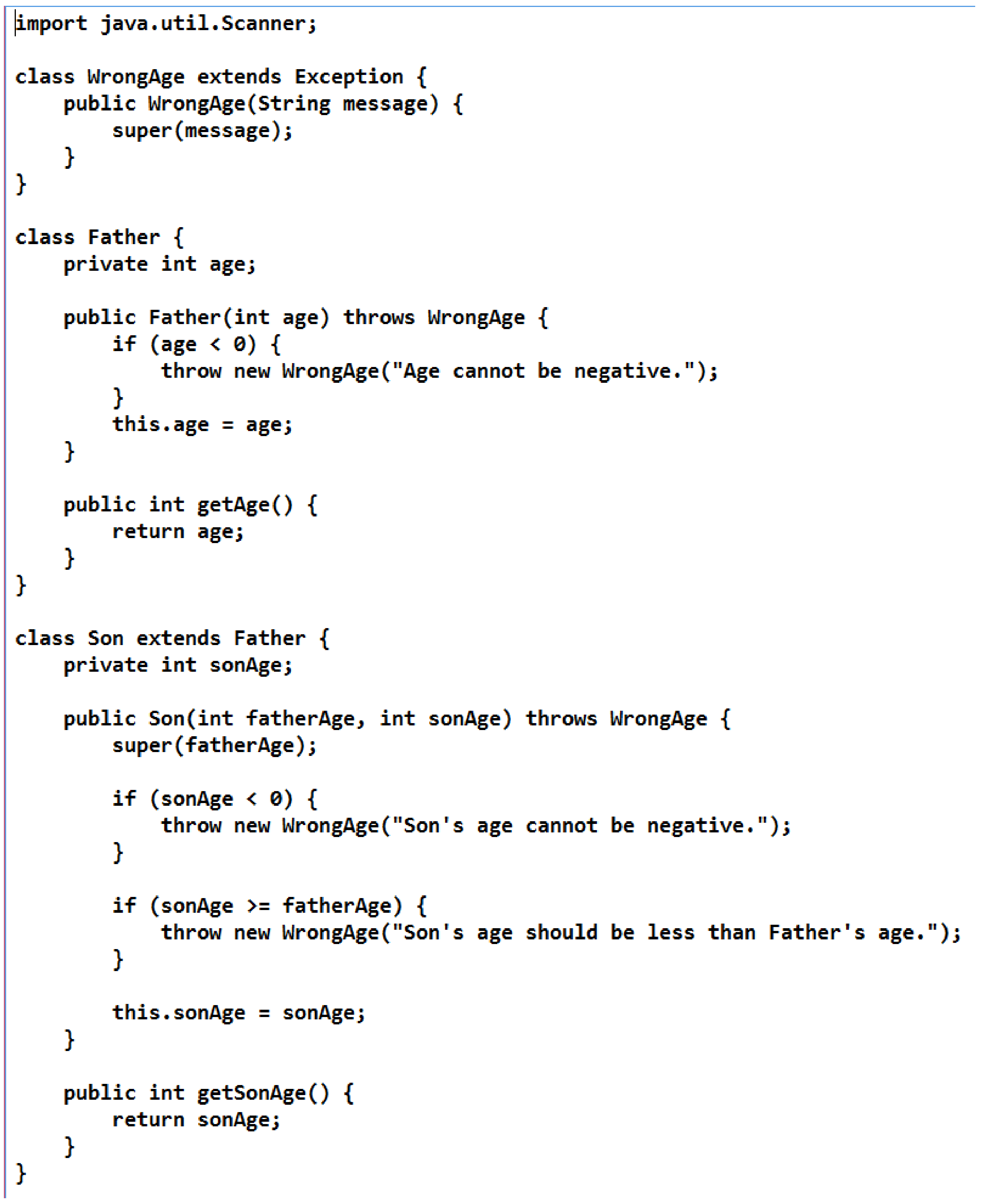
```
OUTPUT:
1. Deposit
2. Withdraw
3. Display Balance
4. Deposit Interest (Savings Account only)
5. Exit.

1.  Enteryour choice:1
    Enter amount : 12000
    Select account (1. savings | 2. current)
    1
    Deposit successful. Updated balance
    :17000

2.  Enter your choice:1
    Enter amount : 23008
    select account (1.savings | 2. current):2
    Deposit successful. Updated balance
    25008

3.  Enter your choice :2
    ~~select~~ enter amount : 2000
    select account (1.savings | 2.current):2
    withdrawal successful. updated
    balance 23008
```

```
4.  Enter your choice: 2
    Enter amount: 1200
    select account (1. savings / 2. current): 1.
    withdrawal successful updated
    balance : 15800

5.  Enter your choice: 3
    select account (1. savings / 2. curent): 1
    current Balance: 15800

6.  Enter your choice: 3
    select account (1. savings / 2. current): 2
    current Balance: 23008

7.  Enter your choice: 4
    select accoun (1. savings): 1
    Interest deposited. Updated balance:
    16590.

8.  Enter your choice: 5
    exiting....
```

[illegible]
19/2/2024

PROGRAM: 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 cases both father and son’s age and throws an exception if son’s age is >=father’s age.

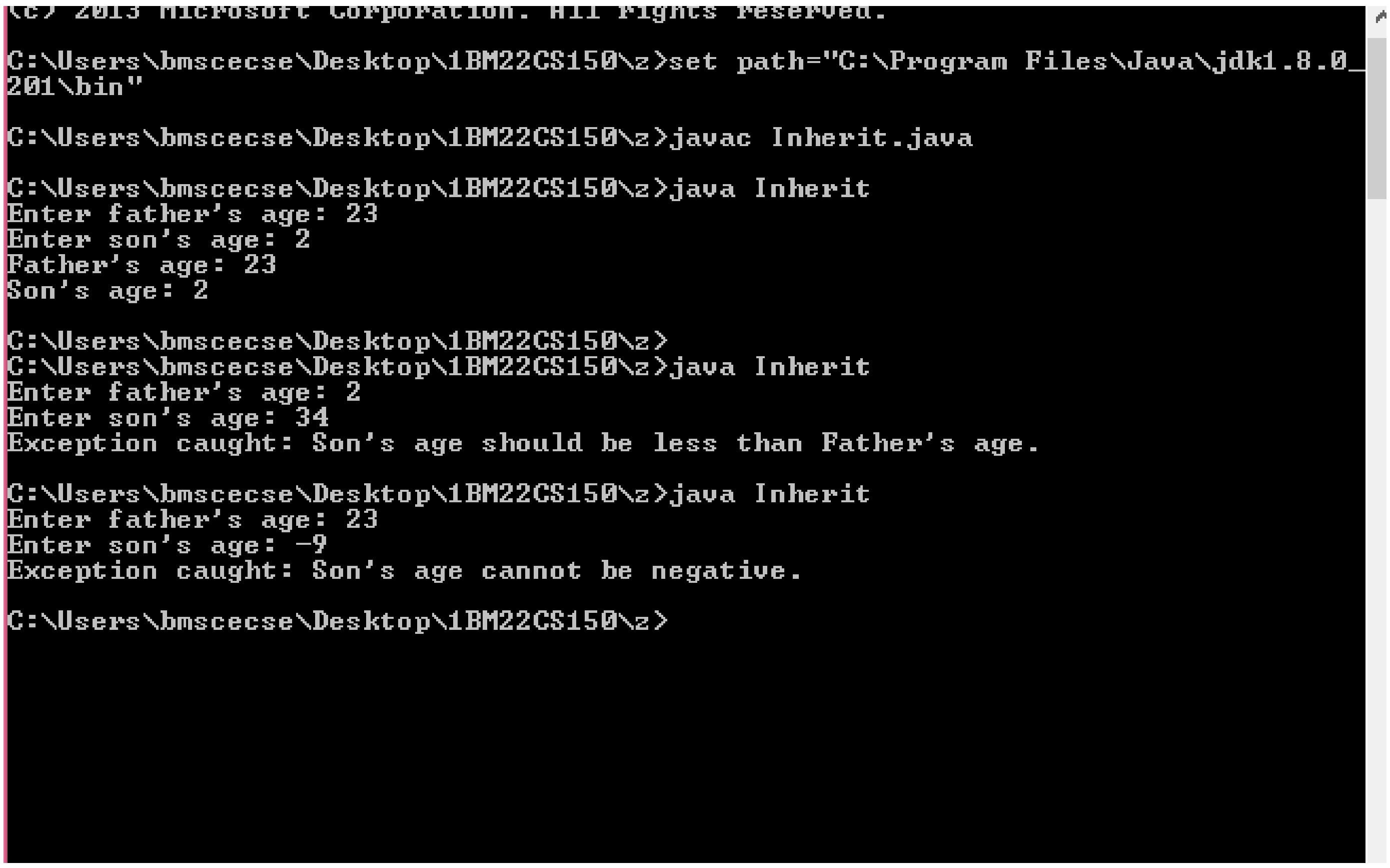
INPUT->



```java
import java.util.Scanner;

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

class Father {
    private int age;

    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("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 WrongAge {
        super(fatherAge);

        if (sonAge < 0) {
            throw new WrongAge("Son's age cannot be negative.");
        }

        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age should be less than Father's age.");
        }

        this.sonAge = sonAge;
    }

    public int getSonAge() {
        return sonAge;
    }
}
```

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

        try {
            System.out.print("Enter father's age: ");
            int fatherAge = scanner.nextInt();
            Father father = new Father(fatherAge);

            System.out.print("Enter son's age: ");
            int sonAge = scanner.nextInt();
            Son son = new Son(fatherAge, sonAge);

            System.out.println("Father's age: " + father.getAge());
            System.out.println("Son's age: " + son.getSonAge());

        } catch (WrongAge e) {
            System.out.println("Exception caught: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("Invalid input. Please enter valid ages.");
        } finally {
            scanner.close();
        }
    }
}
```

OUTPUT->



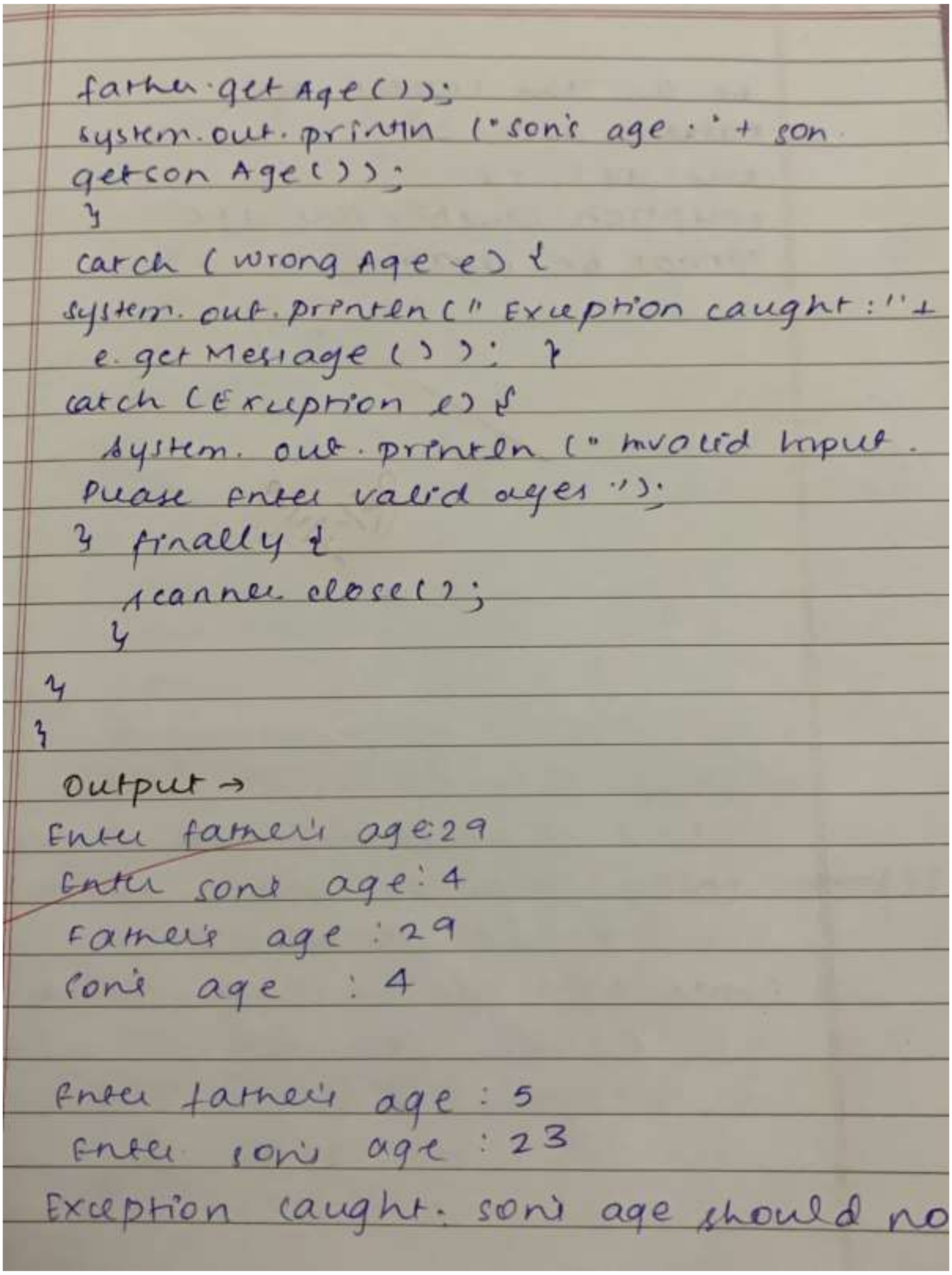
```
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\bmscecse\Desktop\1BM22CS150\z>set path="C:\Program Files\Java\jdk1.8.0_
201\bin"

C:\Users\bmscecse\Desktop\1BM22CS150\z>javac Inherit.java

C:\Users\bmscecse\Desktop\1BM22CS150\z>java Inherit
Enter father's age: 23
Enter son's age: 2
Father's age: 23
Son's age: 2

C:\Users\bmscecse\Desktop\1BM22CS150\z>
C:\Users\bmscecse\Desktop\1BM22CS150\z>java Inherit
Enter father's age: 2
Enter son's age: 34
Exception caught: Son's age should be less than Father's age.

C:\Users\bmscecse\Desktop\1BM22CS150\z>java Inherit
Enter father's age: 23
Enter son's age: -9
Exception caught: Son's age cannot be negative.

C:\Users\bmscecse\Desktop\1BM22CS150\z>
```

OBSERVATION->

01/24 PROGRAMS: EXCEPTION HANDLING

```
import java.util.Scanner;
class wrongAge extends Exception {
public wrongAge (String message){
      super (message);
  }
}
class Father {
 private int age;
public Father (int age) throws wrongAge {
    if (age < 0) {
  throw new wrongAge("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 wrongAge {
    super (fatherAge);
if (sonAge< 0) {
```

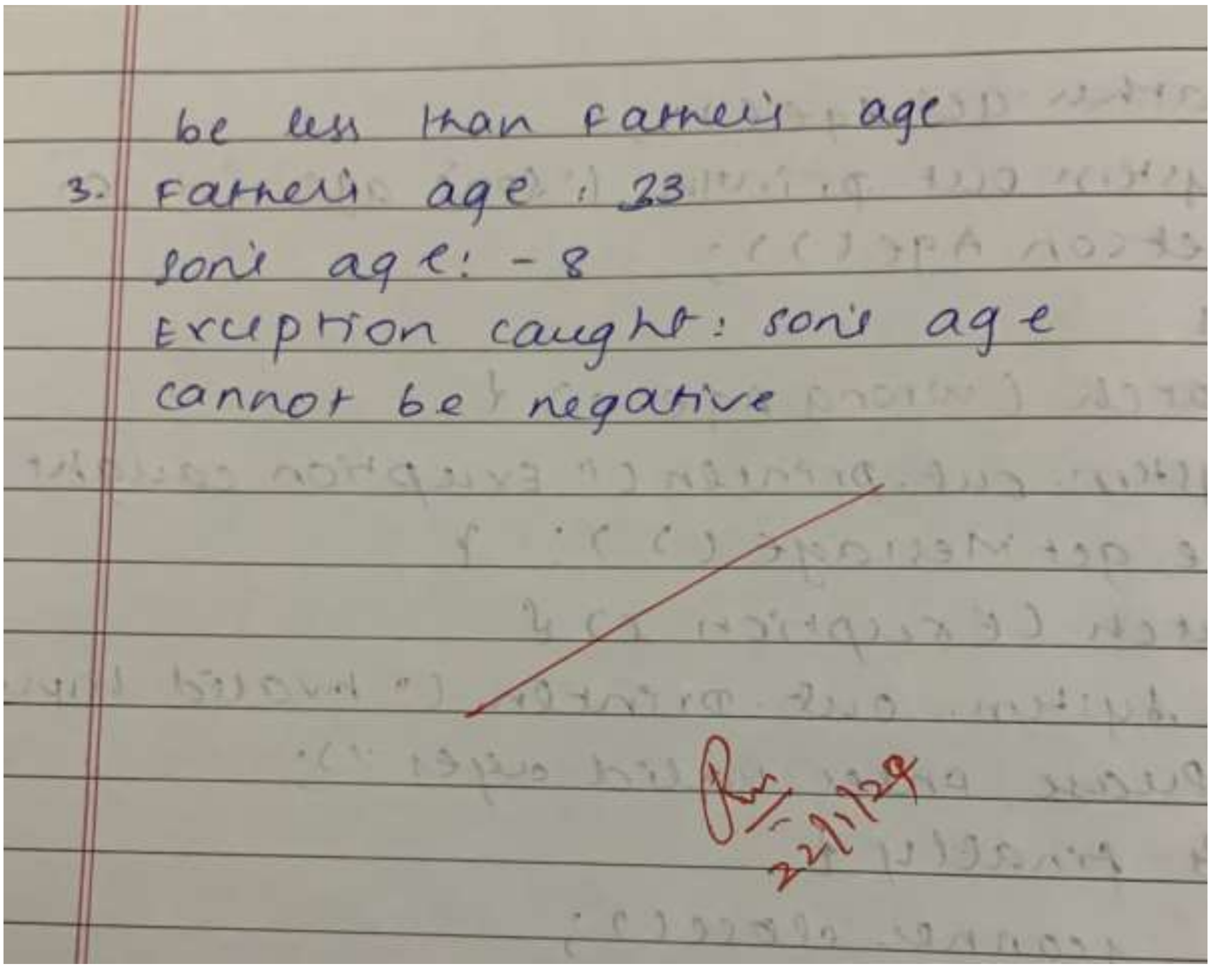
```
throw new WrongAge("son's age cannot
 negative");
 }
 if (sonAge >= fatherAge) {
 throw new WrongAge("son's age
should be less than Father's age.");
 }
 this. sonAge = sonAge;
 }
 public int getSonAge() {
     return sonAge;
 }

}
public class Inheritance {
public static void main (String args[]){
  Scanner scanner = new Scanner(System.in)
try {
 System.out.print("Enter Father's age:");
 int fatherAge = Scanner nextInt();
 Father father = new Father(fatherAge);
System.out.print(" Enter son's age: ");
 int sonAge = scanner.nextInt();
 son son = new Son (fatherAge, sonAge);
 system. out. println("Father's age :" +
```



```
father.getAge());
system.out.println ("son's age : " + son.
getsonAge());
}
catch (wrongAge e) {
system.out.println(" Exception caught :" +
 e.getMessage()); }
catch (Exception e) {
 system.out.println (" Invalid Input.
 Please enter valid ages ");
 } finally {
   scanner.close();
  }
}
}
```

Output →

```
Enter father's age:29
Enter son's age: 4
Father's age : 29
Son's age   : 4

Enter father's age : 5
 Enter son's age : 23
Exception caught. son's age should no
```



be less than father's age

3. Father's age : 23

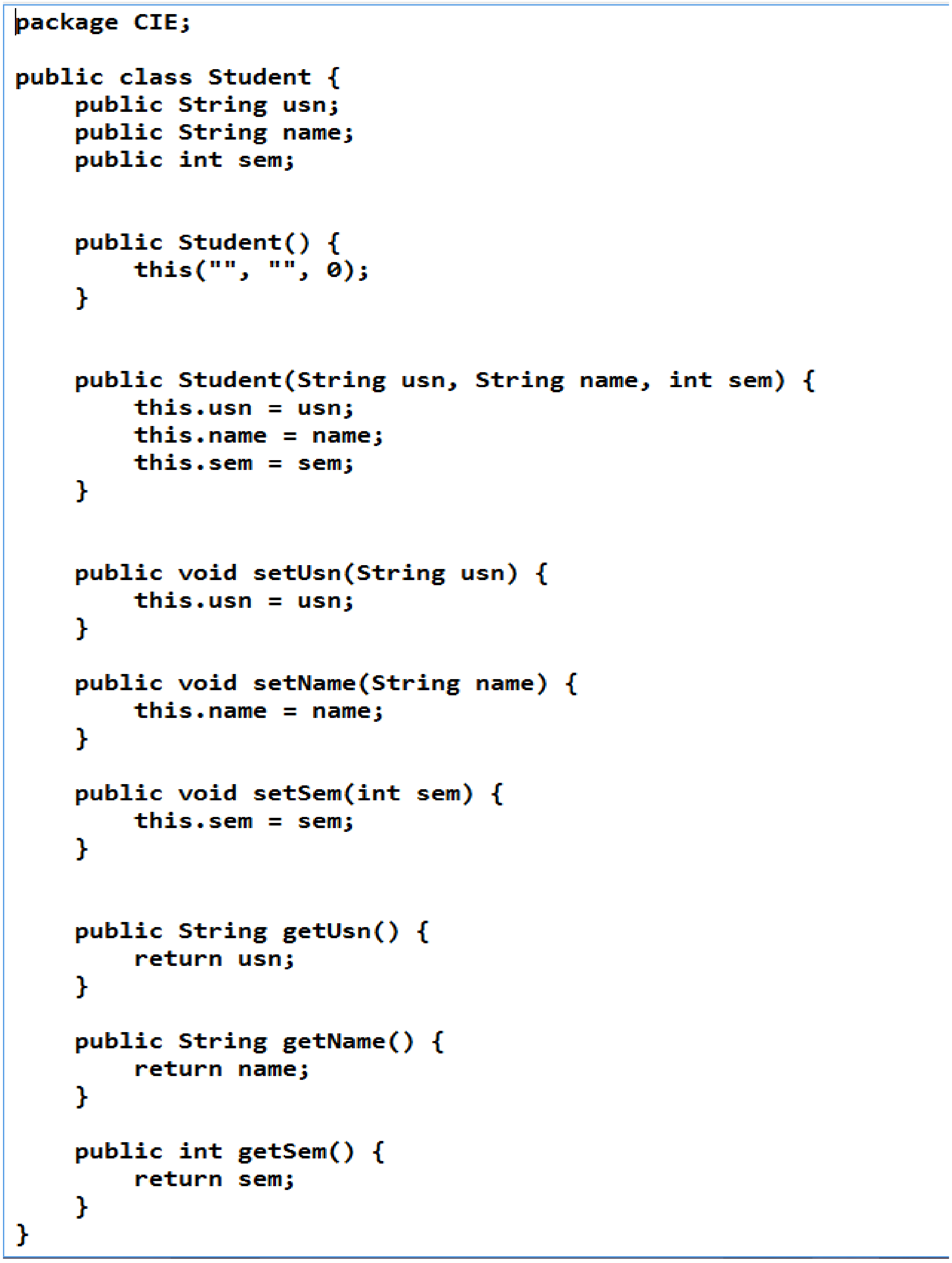
son's age: -8

Exception caught: son's age
cannot be negative

[illegible]

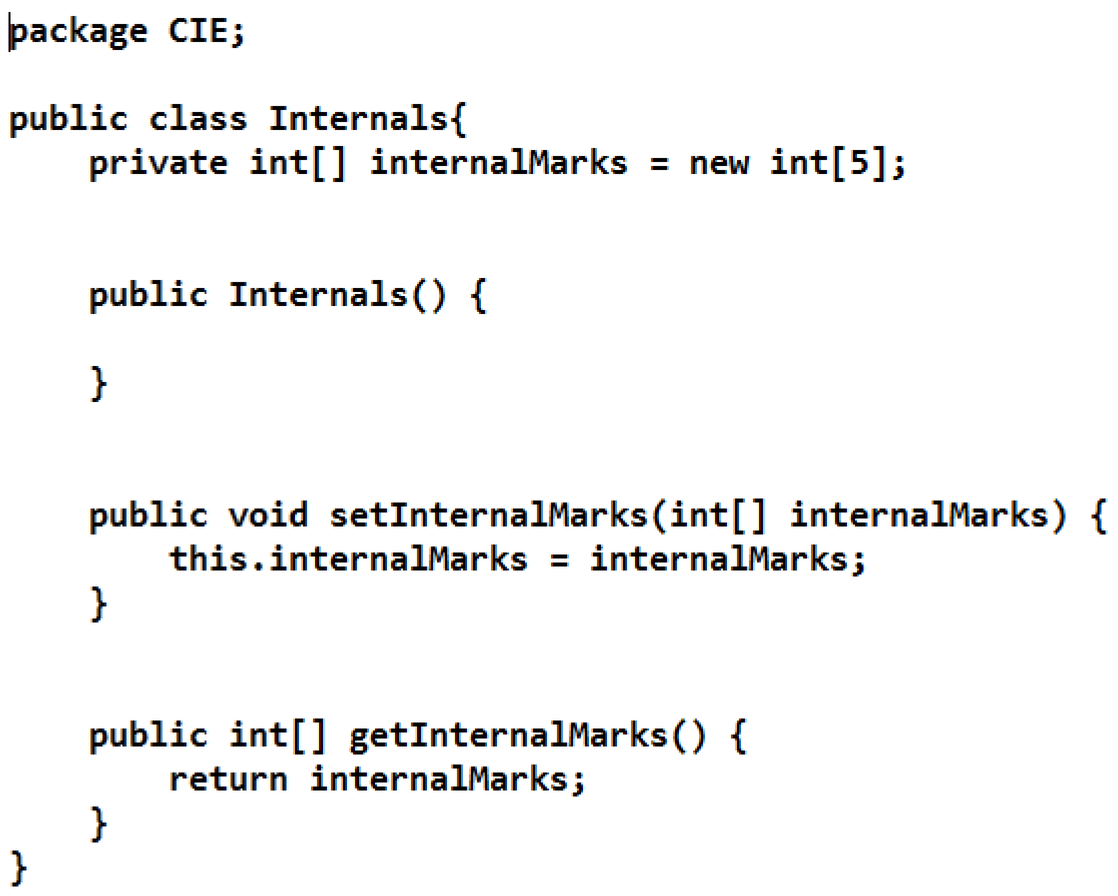
PROGRAM 6: Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class 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.

INPUT-> CIE.STUDENT



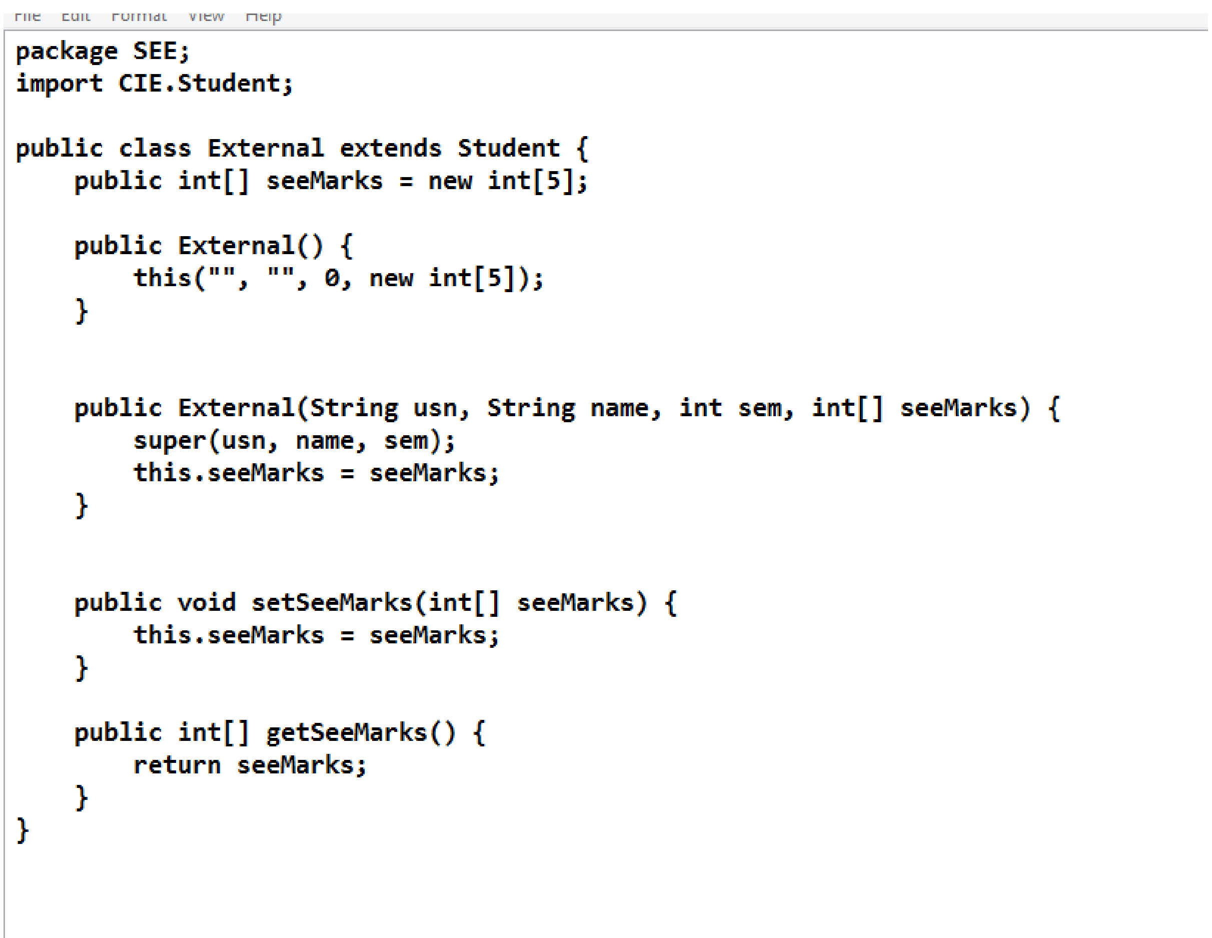
```java
package CIE;

public class Student {
    public String usn;
    public String name;
    public int sem;


    public Student() {
        this("", "", 0);
    }


    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }


    public void setUsn(String usn) {
        this.usn = usn;
    }

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

    public void setSem(int sem) {
        this.sem = sem;
    }


    public String getUsn() {
        return usn;
    }

    public String getName() {
        return name;
    }

    public int getSem() {
        return sem;
    }
}
```

CIE.INTERNALS



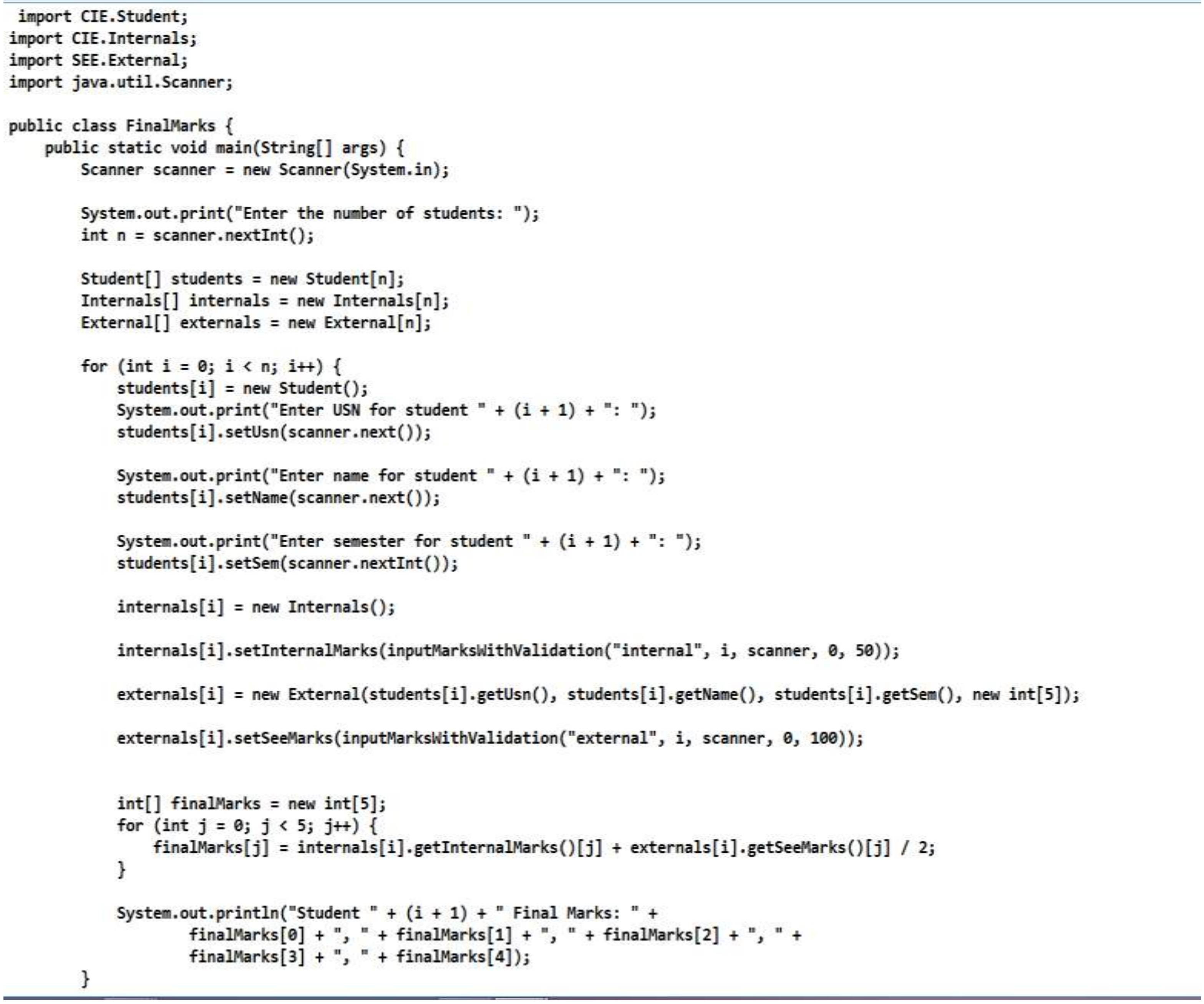
```
package CIE;

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


    public Internals() {

    }


    public void setInternalMarks(int[] internalMarks) {
        this.internalMarks = internalMarks;
    }


    public int[] getInternalMarks() {
        return internalMarks;
    }
}
```

SEE.EXTERNALS

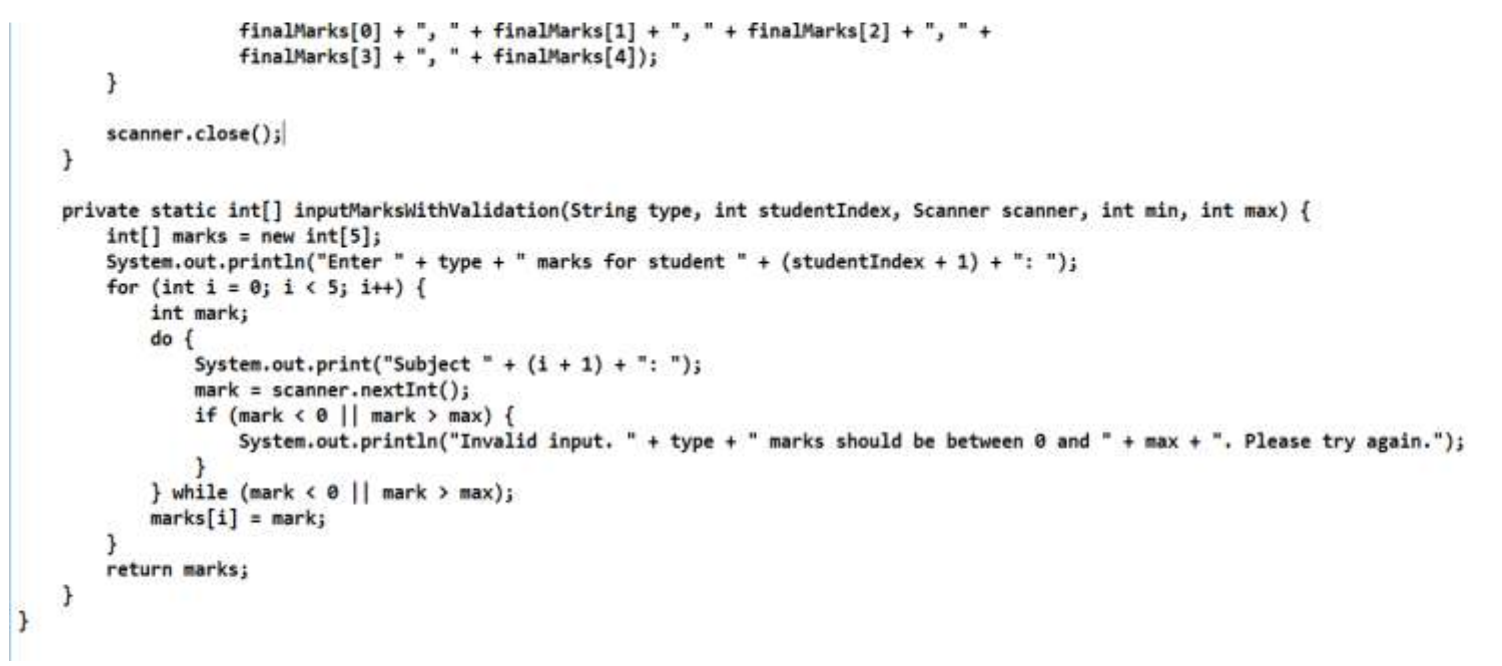


```
package SEE;
import CIE.Student;

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

    public External() {
        this("", "", 0, new int[5]);
    }


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


    public void setSeeMarks(int[] seeMarks) {
        this.seeMarks = seeMarks;
    }

    public int[] getSeeMarks() {
        return seeMarks;
    }
}
```

FINAL MARKS



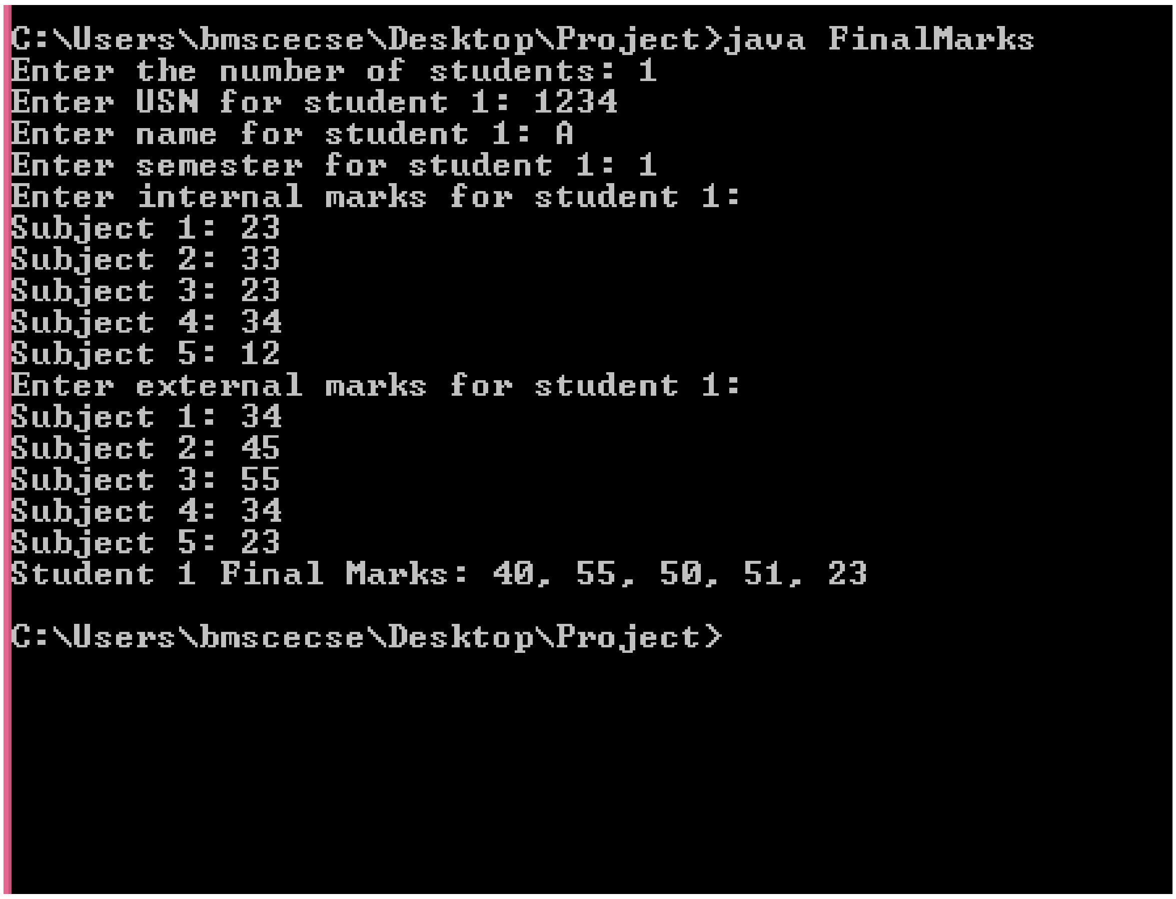
```java
 import CIE.Student;
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

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

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

        Student[] students = new Student[n];
        Internals[] internals = new Internals[n];
        External[] externals = new External[n];

        for (int i = 0; i < n; i++) {
            students[i] = new Student();
            System.out.print("Enter USN for student " + (i + 1) + ": ");
            students[i].setUsn(scanner.next());

            System.out.print("Enter name for student " + (i + 1) + ": ");
            students[i].setName(scanner.next());

            System.out.print("Enter semester for student " + (i + 1) + ": ");
            students[i].setSem(scanner.nextInt());

            internals[i] = new Internals();

            internals[i].setInternalMarks(inputMarksWithValidation("internal", i, scanner, 0, 50));

            externals[i] = new External(students[i].getUsn(), students[i].getName(), students[i].getSem(), new int[5]);

            externals[i].setSeeMarks(inputMarksWithValidation("external", i, scanner, 0, 100));


            int[] finalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                finalMarks[j] = internals[i].getInternalMarks()[j] + externals[i].getSeeMarks()[j] / 2;
            }

            System.out.println("Student " + (i + 1) + " Final Marks: " +
                    finalMarks[0] + ", " + finalMarks[1] + ", " + finalMarks[2] + ", " +
                    finalMarks[3] + ", " + finalMarks[4]);
        }
```



```java
                    finalMarks[0] + ", " + finalMarks[1] + ", " + finalMarks[2] + ", " +
                    finalMarks[3] + ", " + finalMarks[4]);
        }
```

```java

        scanner.close();
    }

    private static int[] inputMarksWithValidation(String type, int studentIndex, Scanner scanner, int min, int max) {
        int[] marks = new int[5];
        System.out.println("Enter " + type + " marks for student " + (studentIndex + 1) + ": ");
        for (int i = 0; i < 5; i++) {
            int mark;
            do {
                System.out.print("Subject " + (i + 1) + ": ");
                mark = scanner.nextInt();
                if (mark < 0 || mark > max) {
                    System.out.println("Invalid input. " + type + " marks should be between 0 and " + max + ". Please try again.");
                }
            } while (mark < 0 || mark > max);
            marks[i] = mark;
        }
        return marks;
    }
}
```

OUTPUT->



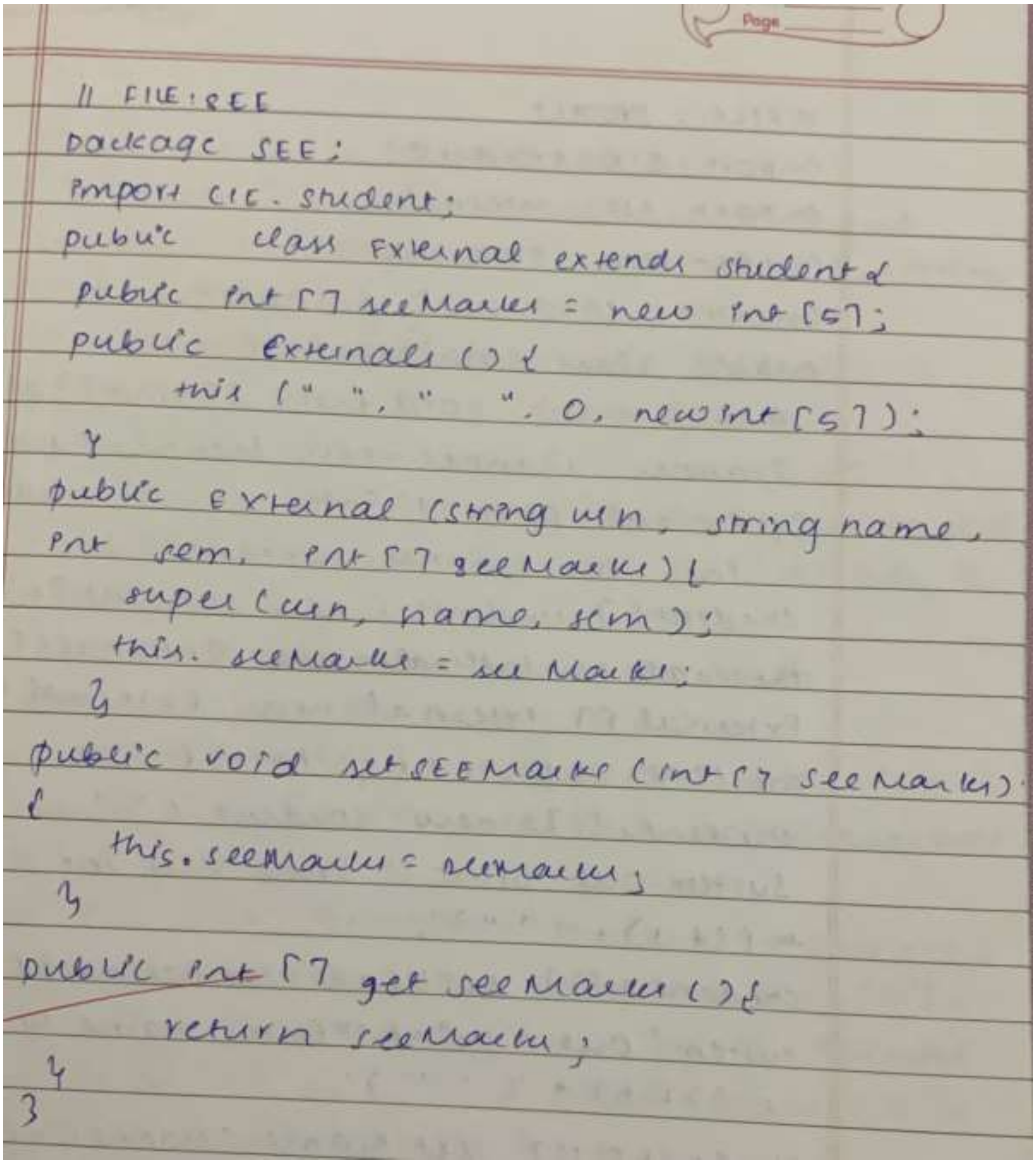
```
C:\Users\bmscecse\Desktop\Project>java FinalMarks
Enter the number of students: 1
Enter USN for student 1: 1234
Enter name for student 1: A
Enter semester for student 1: 1
Enter internal marks for student 1:
Subject 1: 23
Subject 2: 33
Subject 3: 23
Subject 4: 34
Subject 5: 12
Enter external marks for student 1:
Subject 1: 34
Subject 2: 45
Subject 3: 55
Subject 4: 34
Subject 5: 23
Student 1 Final Marks: 40, 55, 50, 51, 23

C:\Users\bmscecse\Desktop\Project>
```

OBSERVATION->

Date ______
Page ______

124 PROGRAM 6: PACKAGES

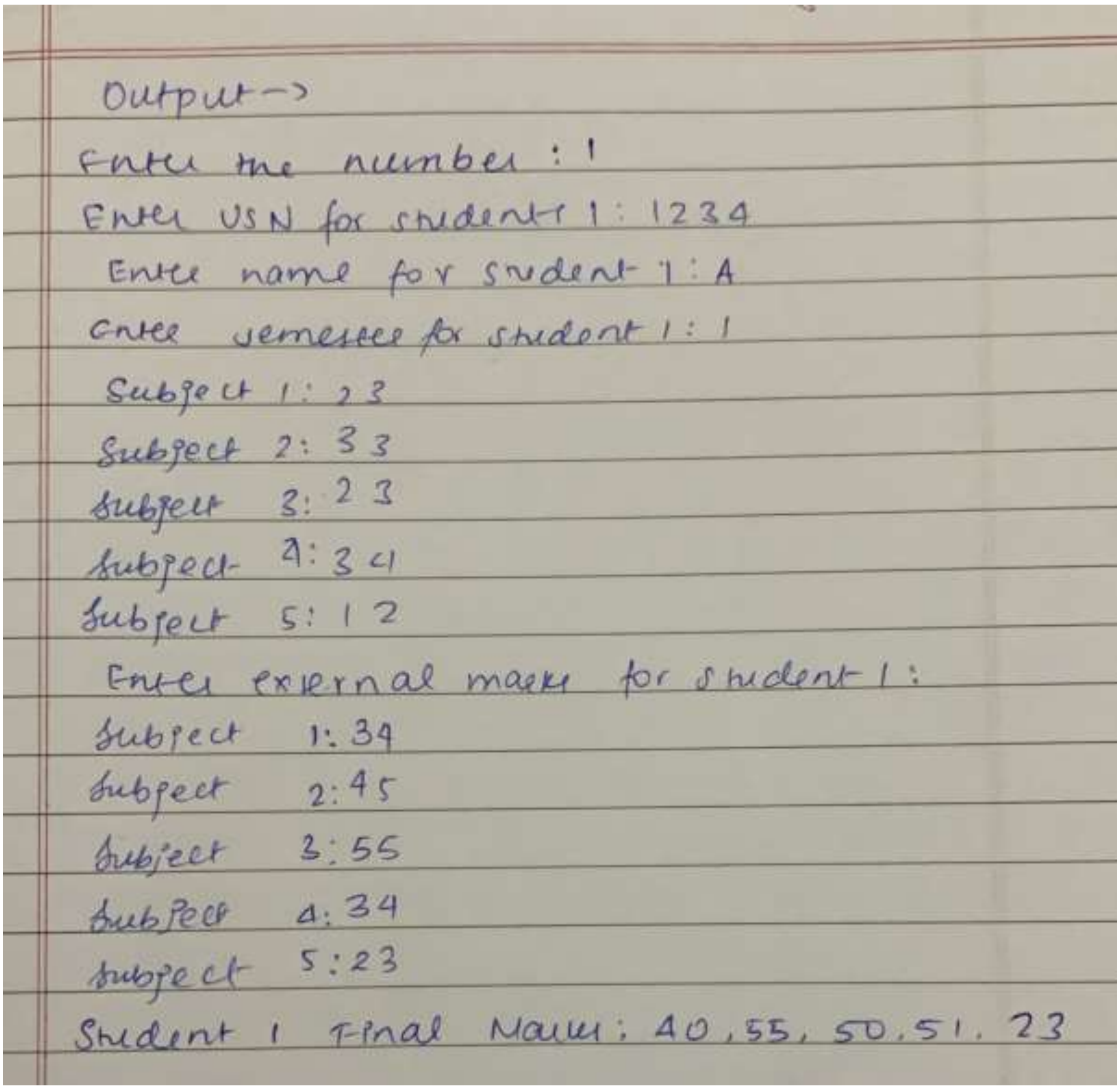
```
// FILE : CIE
package CIE;
public class student {
  public String usn;
  public string name;
  public int sem;
public student () {
   this (" ", "  ", 0);
 }
 public student (string usn, string name,
int sem) {
    this. usn = usn;
    this. name = name;
    this . sem = sem;
  }
 public void set Usn (string usn ){
        this.usn = usn; }
public void set Name ( string name){
       this.name = name; }
public void set sem (int sem ) {
          this. sem = sem ; }
 public string get Usn ( ) {
          return usn;
  }
public String getName(){
      return name;
 }
public int get sem(){
       return sem;
    }
 }


//FILE: CIE
package CIE;
public class Internals {
private int[] internalMarks = new int[5];
public Internals(){

 }
public void setInternalMarks(int[]
internalMarks){
   this. internalMarks = internalMarks;
  }
public int[] get-InternalMarks(){
   return internalMarks;
 }
}
```



```
// FILE:SEE
package SEE;
import CIE. student;
public   class External extends student {
public int [] seeMarks = new int [5];
public Externals () {
     this (" ", "   ". 0, new int [5]);
 }
public External (String usn, String name,
int sem, int [] seeMarks) {
   super (usn, name, sem);
   this. seeMarks = see Marks;
 }
public void set SEEMarks (int [] seeMarks)
{
   this.seeMarks = seeMarks;
 }
public int [] get seeMarks () {
     return seeMarks;
 }
}
```

```
// FILE : PROJECT
import CIE. student;
import CIE. internals;
import SEE. external;
import java. util. scanner;
public class FinalMarks {
 public static void main (String [] args) {
 Scanner scanner = new scanner(System.in
System.out. print ("Enter no. of students:
 int n = scanner. nextInt ();
student [] students = new student [n];
Internals [] internals = new Internals [n];
External [] external = new External [n];
for (int i=0; i<n; i++) {
students [i] = new student ();
System.out. print (" Enter USN for student
+ (i+1)+ " : ");
students [i]. setUsn ( scanner. next ());
system. out . print ("Enter name of stud "+
+ (i+1) + " : ");
students [i]. set Name (scanner. next());
system. out. print (" Enter sem of stud "+
i+1 ) + " : ");
students [i]. set sem (scanner. next()
internals [i] = newInternals ();
Internals [i]. setInternalMarks (input
Marks withvalidation ("internal", i, scanner
, 0, 50));
externals[i]= new Externals (students [i].
getUsn (), students[i]. getName,
students[i]. getSem (), new int[5]);
externals [i] . set seeMarks (inputMarksw
ithValidation ("external", i, scanner, 0,
100 ));
int [] finalMarks = new int [5];
for (int j = 0; j< 5; j++) {
  finalMarks [j] = internals [i] .getInternal
Marks () [j]+ externals [i]. getSeeMarks()
[j] / 2; }
system. out. println ("Student " + (i+1) +
"Final Marks : " + final Marks [0]+
" , " + finalMarks [1] + " , " + final
Marks [2] + " ," + final Marks [3]+
" , " + finalMarks [4] ));
}
scanner. close ();
}
```

```
private static int[] inputMarks with
Validation (String type, int studentInd
Scanner scanner, int min, int max) {
  int []marks = new int[5];
System.out.println ("Enter" + type + "
marks for student " + (studentIndex+1)
+ " : ");
for (int i = 0; i < 5; i++) {
  int mark;
 do {
 S.O.P ("Subject " + (i+1) + " : ");
 mark = scanner.nextInt();
 if (mark < 0 || mark > mark) {
 System.out.println ("invalid input"
+ type + " marks should be btwn 0 "
+ max + " please try again ");
}
} while (mark < 0 || mark > mark);
marks [i] = mark;
}
return marks;
}
}
```

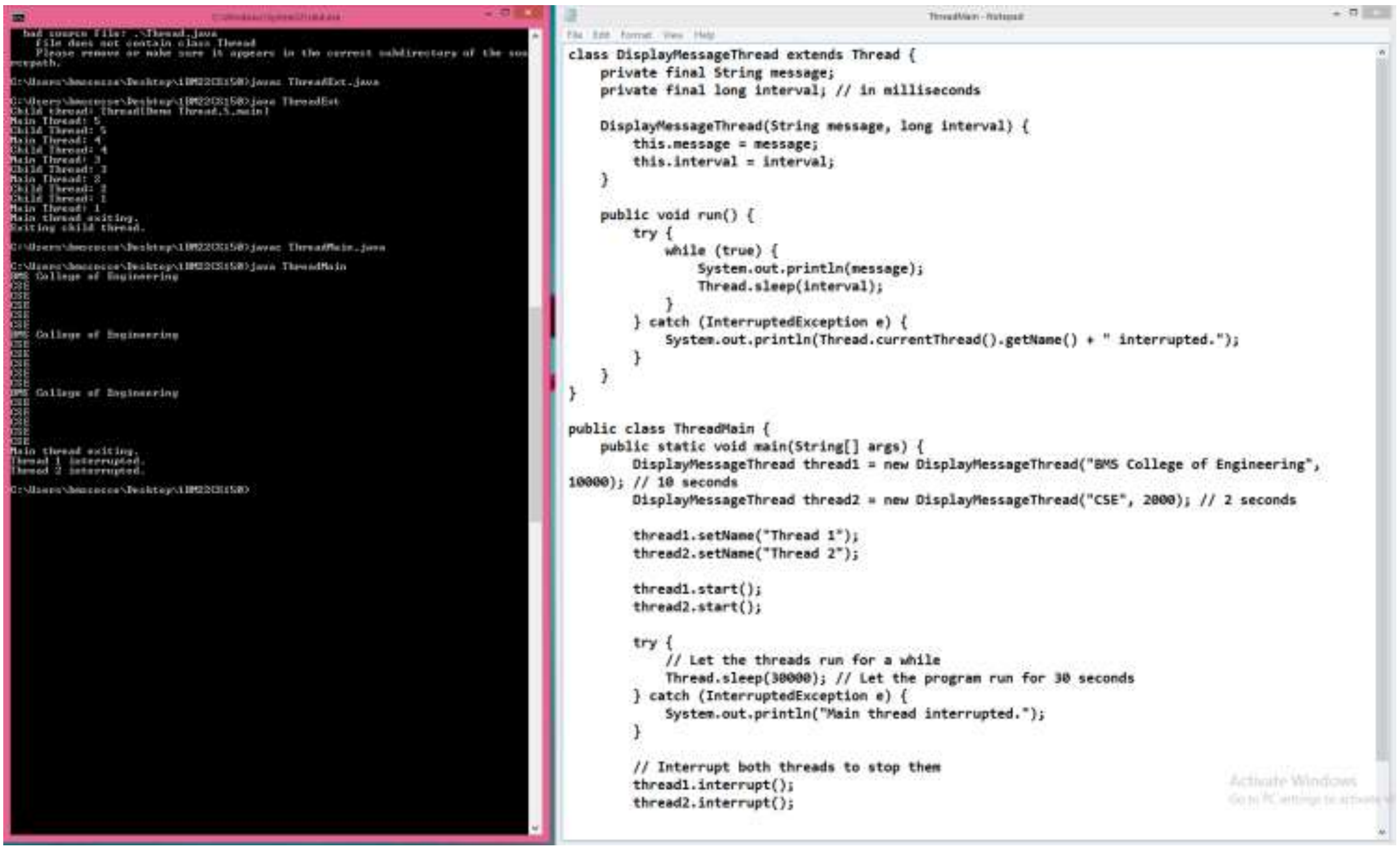


Output->

```
Enter the number : 1
Enter USN for student 1: 1234
 Enter name for student 1: A
Enter semester for student 1: 1
 Subject 1: 23
 Subject 2: 33
 Subject 3: 23
 Subject 4: 34
 Subject 5: 12
 Enter external marks for student 1:
 Subject   1: 34
 Subject   2: 45
 Subject   3: 55
 Subject   4: 34
 Subject  5:23
Student 1 Final Marks: 40, 55, 50, 51, 23
```

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.

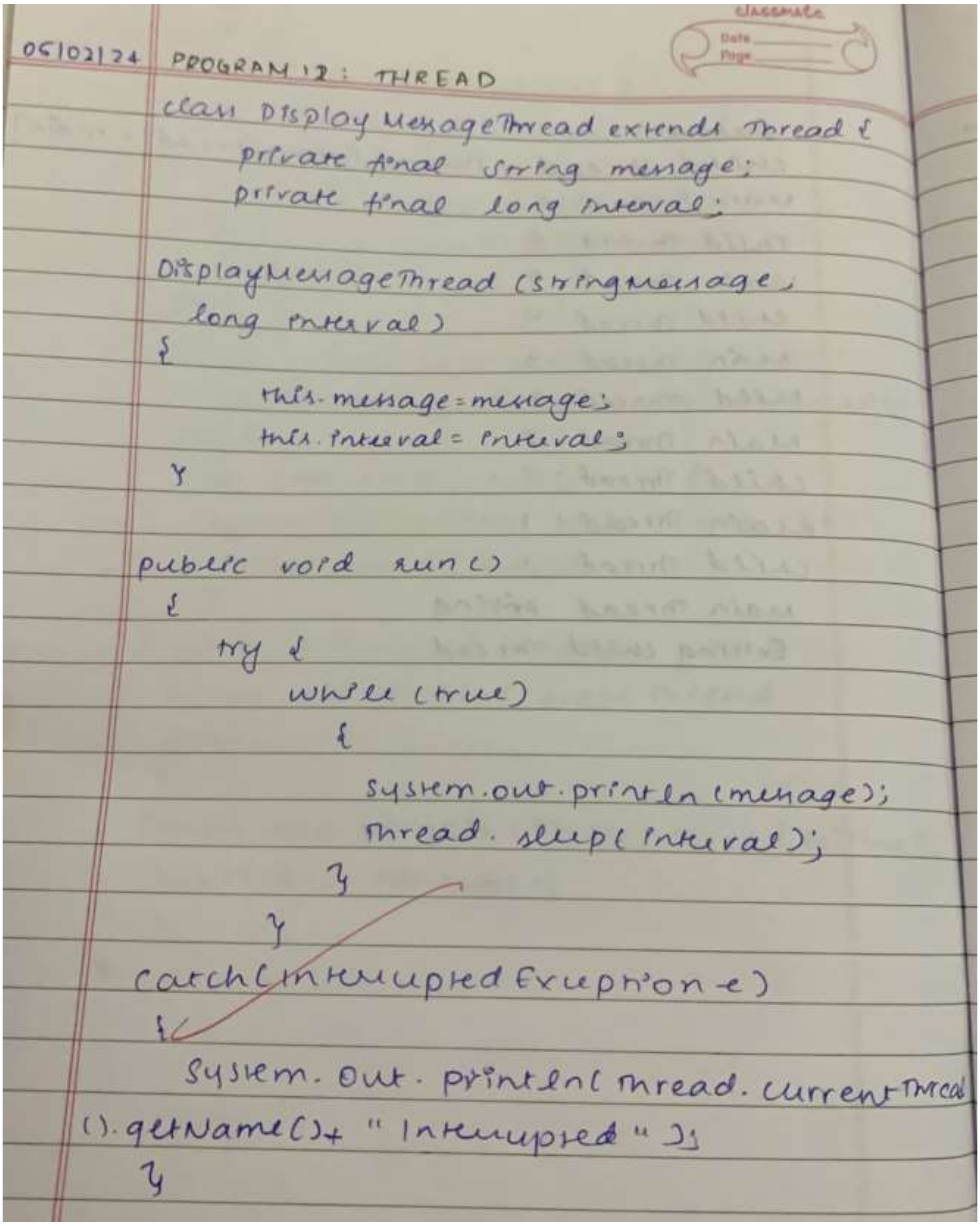
INPUT AND OUTPUT ->



```
  bad source file: .\Thread.java
    file does not contain class Thread
    Please remove or make sure it appears in the correct subdirectory of the sou
rcepath.

C:\Users\bmscecse\Desktop\1BM22CS150>javac ThreadExt.java

C:\Users\bmscecse\Desktop\1BM22CS150>java ThreadExt
Child thread: Thread[Demo Thread,5,main]
Main Thread: 5
Child Thread: 5
Main Thread: 4
Child Thread: 4
Main Thread: 3
Child Thread: 3
Main Thread: 2
Child Thread: 2
Child Thread: 1
Main Thread: 1
Main thread exiting.
Exiting child thread.

C:\Users\bmscecse\Desktop\1BM22CS150>javac ThreadMain.java

C:\Users\bmscecse\Desktop\1BM22CS150>java ThreadMain
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
Main thread exiting.
Thread 1 interrupted.
Thread 2 interrupted.

C:\Users\bmscecse\Desktop\1BM22CS150>
```

```java
class DisplayMessageThread extends Thread {
    private final String message;
    private final long interval; // in milliseconds

    DisplayMessageThread(String message, long interval) {
        this.message = message;
        this.interval = interval;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e) {
            System.out.println(Thread.currentThread().getName() + " interrupted.");
        }
    }
}

public class ThreadMain {
    public static void main(String[] args) {
        DisplayMessageThread thread1 = new DisplayMessageThread("BMS College of Engineering",
10000); // 10 seconds
        DisplayMessageThread thread2 = new DisplayMessageThread("CSE", 2000); // 2 seconds

        thread1.setName("Thread 1");
        thread2.setName("Thread 2");

        thread1.start();
        thread2.start();

        try {
            // Let the threads run for a while
            Thread.sleep(30000); // Let the program run for 30 seconds
        } catch (InterruptedException e) {
            System.out.println("Main thread interrupted.");
        }

        // Interrupt both threads to stop them
        thread1.interrupt();
        thread2.interrupt();
```

OBSERVATION->



05/02/24 PROGRAM 12: THREAD

```
class DisployMessageThread extends Thread {
        private final String message;
        private final long interval;

DisplayMessageThread (StringMessage,
  long interval)
{
        this.message=message;
        this.interval= interval;
}

public void run()
{
    try {
        while (true)
            {
                System.out.println(message);
                Thread. sleep(interval);
            }
        }
catch(InterruptedException-e)
{
   System. Out. println(Thread. currentThread
().getName()+ " Interrupted " );
}
```

```
    }
  }

public clas ThreadMain {
public static voidmain (string args[])
  {
    DisplayMenageThread thread1 = new
     DisplayMenageThread("BMS college of Engineerin
     1000);
  Display MenageThread thread2: new
DisplayMenageThread(" CSE ", 2000);
      thread 1. setName("Thread 1");
      thread 2. setName(" Thread 2");
    thread1. start();
    thread2. start();
    try {
         Thread. sleep (3000);
       }
   catch(InterruptedException e)
     {
       System.out.println("Main Thread Interupted
     }
    thread 1. interupt();
    thread 2. interupt();
  system.out.println("Main Thread exiting"
  }
}
```

~~Out~~

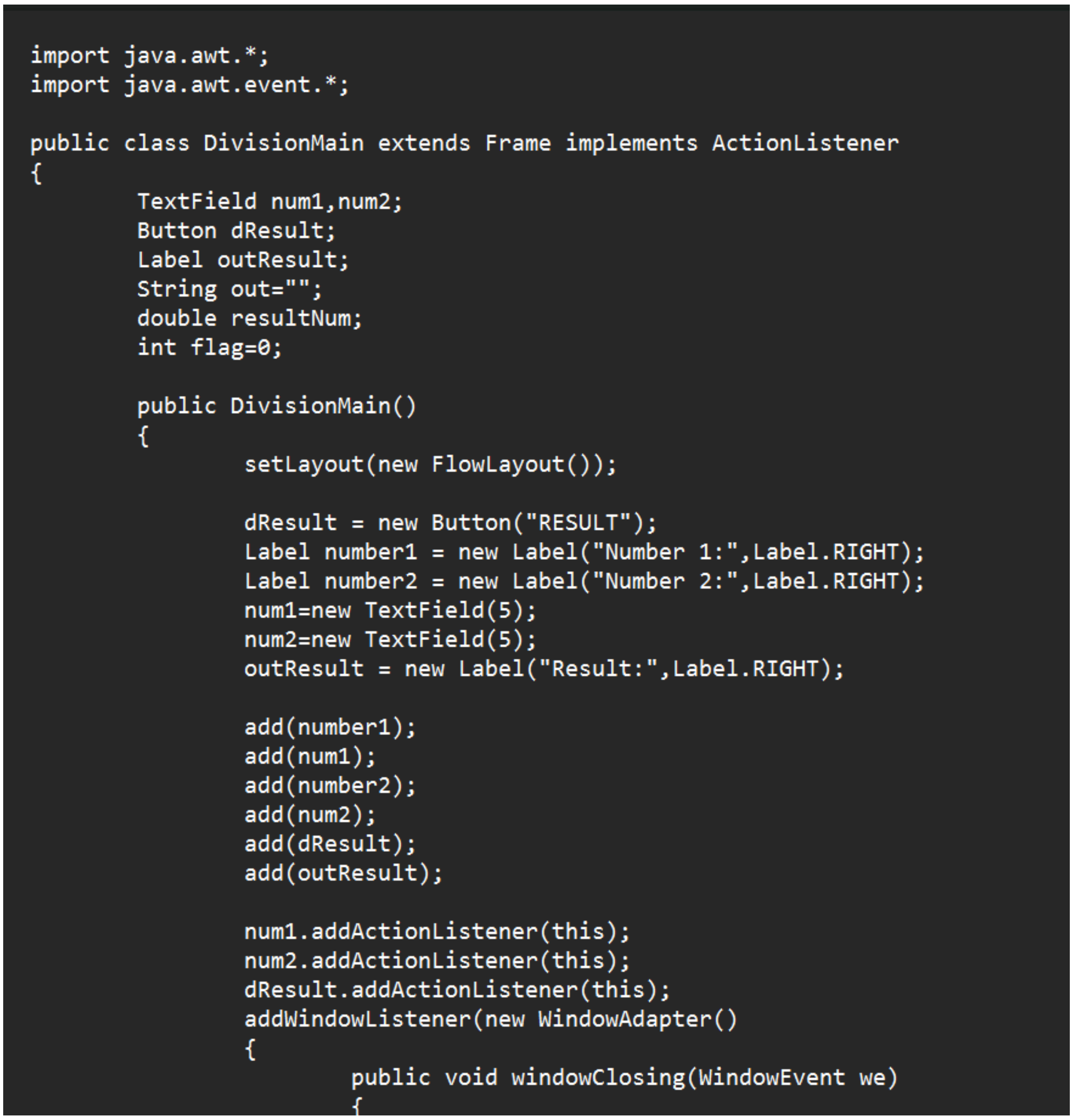
OUTPUT →

```
BMSCE
CSE
CSE
CSE
CSE
CSE
BMSCE
CSE
CSE
CSE
CSE
CSE
Main Thread exiting
Thread 1 interrupted
Thread 2 interrupted.
```

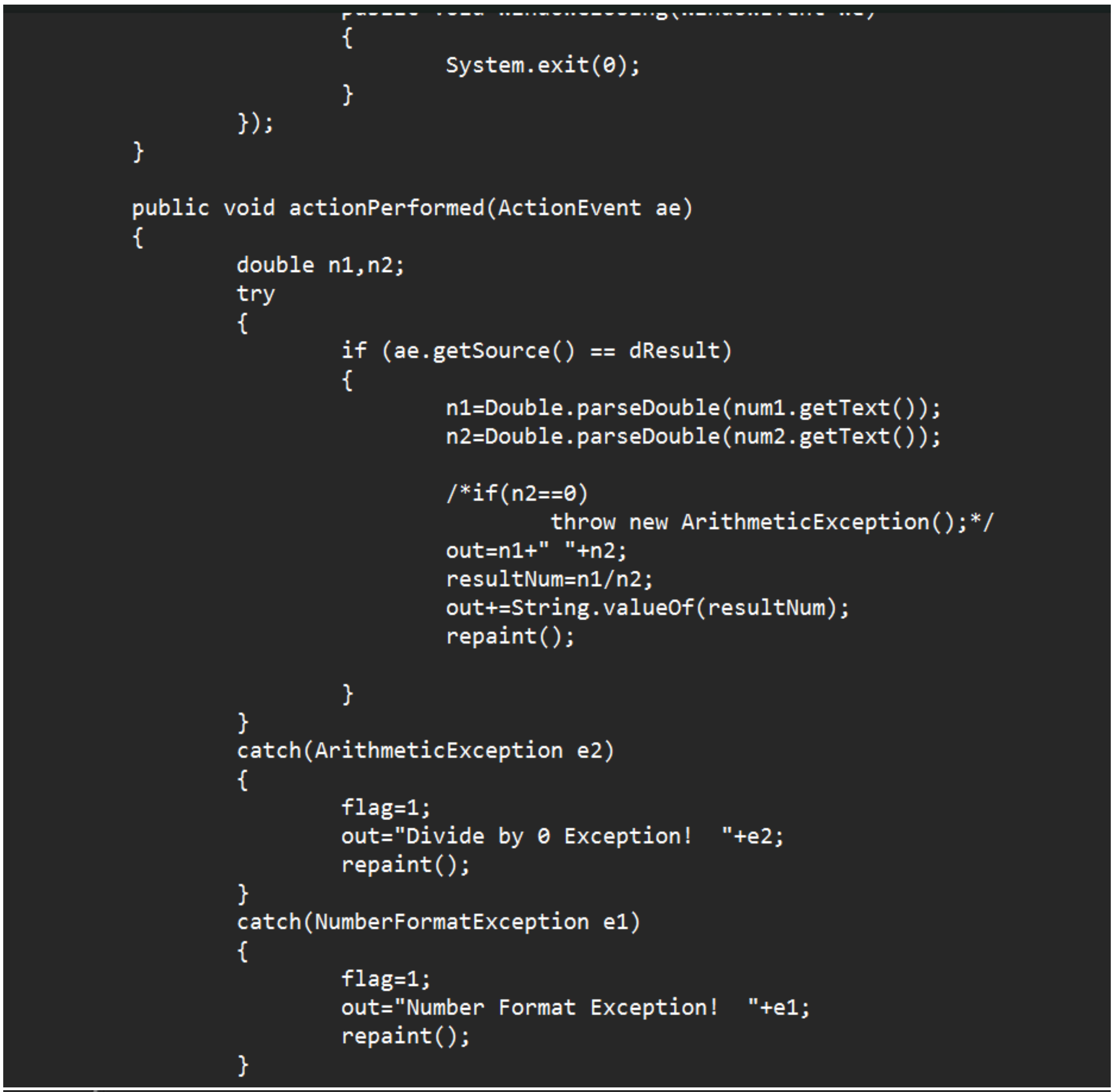
5/2/2024

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.

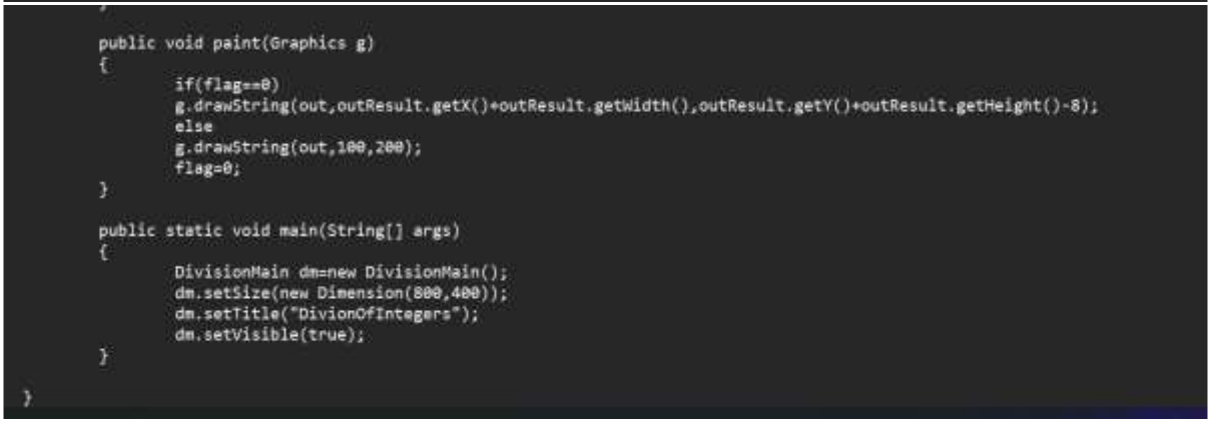
INPUT->



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

public class DivisionMain extends Frame implements ActionListener
{
        TextField num1,num2;
        Button dResult;
        Label outResult;
        String out="";
        double resultNum;
        int flag=0;

        public DivisionMain()
        {
                setLayout(new FlowLayout());

                dResult = new Button("RESULT");
                Label number1 = new Label("Number 1:",Label.RIGHT);
                Label number2 = new Label("Number 2:",Label.RIGHT);
                num1=new TextField(5);
                num2=new TextField(5);
                outResult = new Label("Result:",Label.RIGHT);

                add(number1);
                add(num1);
                add(number2);
                add(num2);
                add(dResult);
                add(outResult);

                num1.addActionListener(this);
                num2.addActionListener(this);
                dResult.addActionListener(this);
                addWindowListener(new WindowAdapter()
                {
                        public void windowClosing(WindowEvent we)
                        {
```

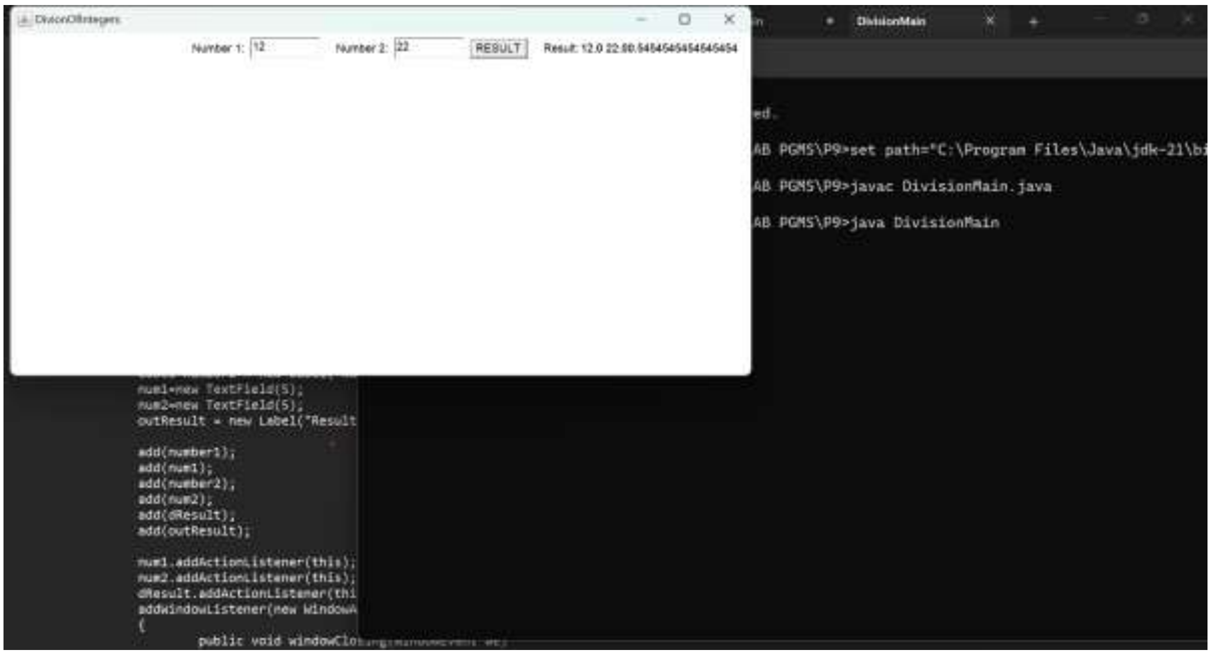


```
                {
                        System.exit(0);
                }
        });
}

public void actionPerformed(ActionEvent ae)
{
        double n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        n1=Double.parseDouble(num1.getText());
                        n2=Double.parseDouble(num2.getText());

                        /*if(n2==0)
                                throw new ArithmeticException();*/
                        out=n1+" "+n2;
                        resultNum=n1/n2;
                        out+=String.valueOf(resultNum);
                        repaint();

                }
        }
        catch(ArithmeticException e2)
        {
                flag=1;
                out="Divide by 0 Exception!  "+e2;
                repaint();
        }
        catch(NumberFormatException e1)
        {
                flag=1;
                out="Number Format Exception!  "+e1;
                repaint();
        }
```



```
        public void paint(Graphics g)
        {
                if(flag==0)
                g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.getY()+outResult.getHeight()-8);
                else
                g.drawString(out,100,200);
                flag=0;
        }

        public static void main(String[] args)
        {
                DivisionMain dm=new DivisionMain();
                dm.setSize(new Dimension(800,400));
                dm.setTitle("DivionOfIntegers");
                dm.setVisible(true);
        }

}
```

OUTPUT->

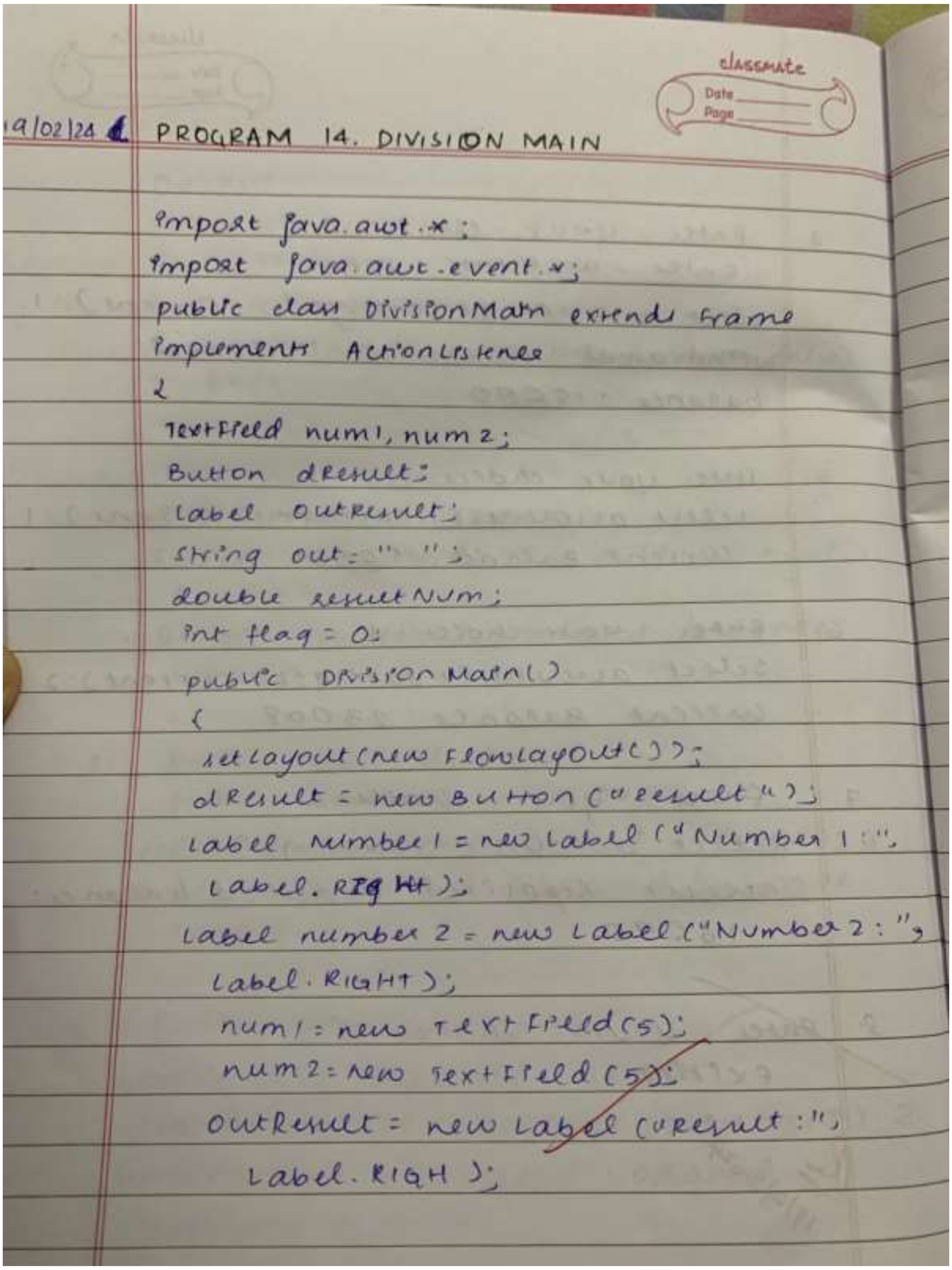


```
Number 1: 12    Number 2: 22    RESULT    Result: 12.0 22.00.5454545454545454
```

```
ed.

AB PGMS\P9>set path="C:\Program Files\Java\jdk-21\bi

AB PGMS\P9>javac DivisionMain.java

AB PGMS\P9>java DivisionMain
```

```
num1=new TextField(5);
num2=new TextField(5);
outResult = new Label("Result

add(number1);
add(num1);
add(number2);
add(num2);
add(dResult);
add(outResult);

num1.addActionListener(this);
num2.addActionListener(this);
dResult.addActionListener(thi
addWindowListener(new WindowA
{
        public void windowClo
```

OBSERVATION->



classmate

Date
Page

19/02/24 PROGRAM 14. DIVISION MAIN

```
import java.awt.*;
import java.awt.event.*;
public class DivisionMain extends Frame
implements ActionListener
{
 TextField num1, num2;
 Button dResult;
 Label outResult;
 String out=" ";
 double resultNum;
 int flag=0;
 public DivisionMain()
 {
  setLayout(new FlowLayout());
  dResult = new Button("result");
  Label Number1 = new Label("Number 1:",
   Label.RIGHt);
  Label number2 = new Label("Number2:",
   Label.RIGHT);
   num1 = new TextField(5);
   num2 = new TextField(5);
  outResult = new Label("Result:",
   Label.RIGH);
   add (number1);
   add (num1);
   add (number2);
   add (num2);
   add (dResult);
   add( outResult);
   num1.add ActionListener (this);
   num2 add ActionListener (this);
   dResult. addAction Listener (this);
   addWindow Listener (new WindowAdapter()
  {   public void windowclosing (
        WindowEvent  we)
       {
         System. exit(0);
         }
   });
}
public void actionPerformed (ActionEvent
  ae)
 {
   double n1, n2;
   try
   { if (ae.getSource()== dResult )
   {
n1= Double.parseDouble (num1.getText());
n2 =Double.parseDouble(num2.getText());
        out = n1 + "   " + n2;
    resultNum = n1/n2;
        out + = String.valueOf (resultNum);
        repaint ();
    }
}
catch(ArithmeticException  e2)
{   flag= 1;
    out = " Divide by 0 Exception ! "+e2.
    repaint ();
}
catch (NumberFormat Exception  e1)
{
      flag =1;
   out = "Number Format Exception! "+e1;
   repaint ();
}
}
public void paint (Graphics g)
{ if (flag = = 0)
g. drawString (out, outResult.getx() +
outResult .getwidth(), outResult.getY() +
outResult  .getHeight() - 8);
  else
```

```
g.drawString(out, 100, 200);
   flag = 0;
}
public static void main(String[] args)
{
 DivisionMain dm = new DivisionMain();
dm.setSize(new Dimension(800, 400));
dm.setTitle("Division of Integers");
dm.setVisible(true);
}
}
```

OUTPUT:

```
Number 1: [12]  Number 2: [22]

[Result]        Result: 12.0 22.0 0.545454
```