WEEK 1:

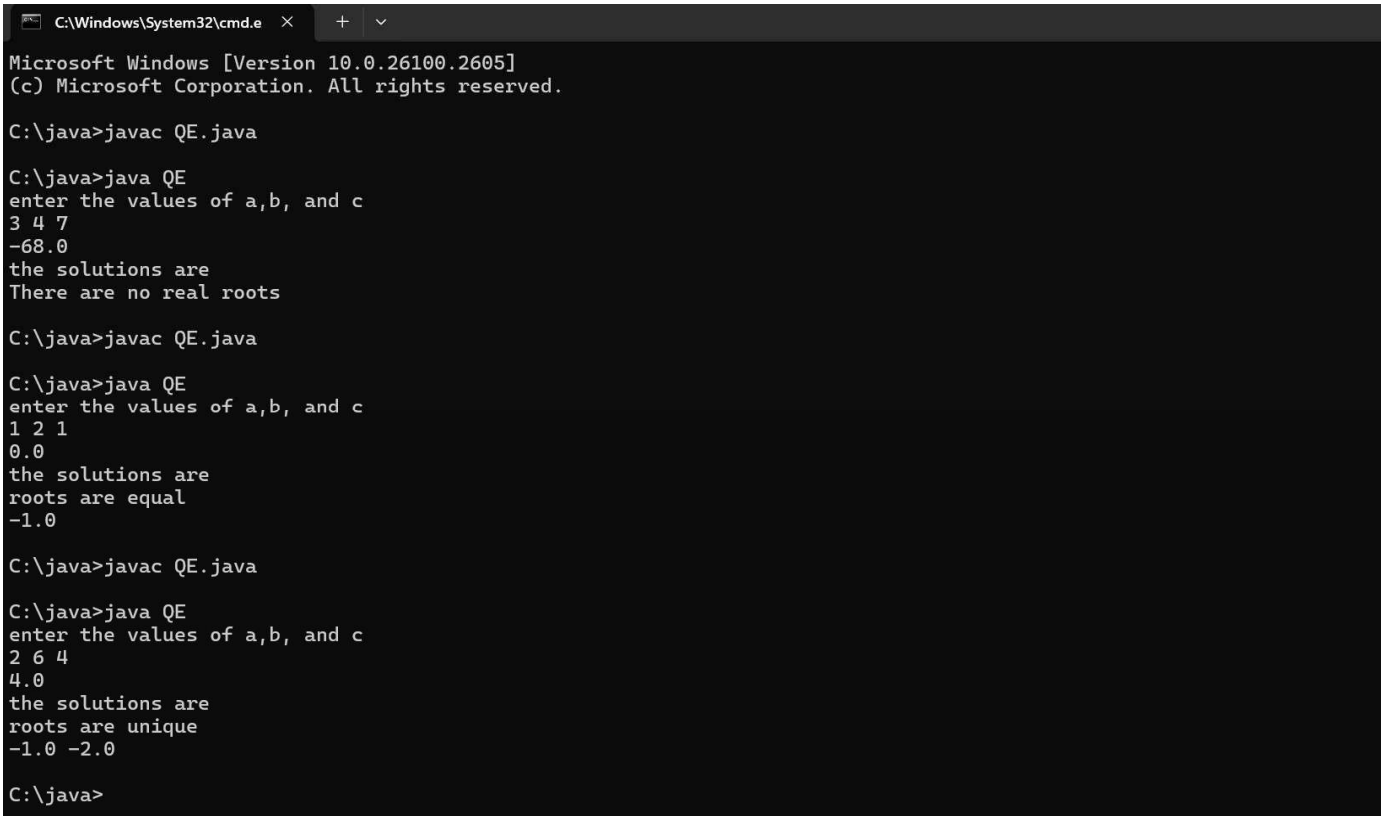
Develop a Java program that prints all real solu ons to the quadra c equa on ax2+bx+c=0. Read in a, b, c and use the quadra c formula. If the discriminate b2-4ac is nega ve, display a message sta ng that there are no real solu ons.

Source Code:

```
 import java.util.Scanner;

class Quadratic {     float d;
   Scanner sc = new Scanner(System.in);
    void solver()


   {
       System.out.println("enter the values of a,b, and c");        int a = sc.nextInt();
int b = sc.nextInt();        int c = sc.nextInt();



      if (a == 0) {
         System.out.println("invalid equation");
      }        else{
         d= b*b - 4*a*c;
         System.out.println(d);
         System.out.println("the solutions are");            if(d>0){
            System.out.println("roots are unique ");               double r1 = (-
b+Math.sqrt(d))/(2*a);               double r2 = (-b-Math.sqrt(d))/(2*a);
            System.out.println(r1 +" " + r2);
         }
         if(d==0){
            System.out.println("roots are equal ");               double r = -b/(2*a);
            System.out.println(r);
         }           if(d<0){
            System.out.println("There are no real roots" );
         }
      }
```

```
   }


}



public class QE {
   public static void main(String[] args) {        Quadratic q1 = new Quadratic();         q1.solver();

  }
}
```

OUTPUT:



Pgm 1 25/09/24

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

```
import java.util.Scanner;
public class QuadraticEquation {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter coefficient a: ");
        double a = sc.nextDouble();
        System.out.println("Enter coefficient b: ");
        double b = sc.nextDouble();
        System.out.println("Enter coefficient c: ");
        double c = sc.nextDouble();
        double discriminant = b*b - 4*a*c;

        if(discriminant > 0){
            double root1 = (-b + Math.sqrt(discriminant))/(2*a);
            double root2 = (-b - Math.sqrt(discriminant))/(2*a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root 1 = " + root1);
            System.out.println("Root 2 = " + root2);
        }
        else if(discriminant == 0) {
```

```
            double root = -b/(2*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root = " + root);
        }
    else {
            System.out.println("There are no real solutions");
        }
    }
}
```

⇒ Enter coefficient a:
1
Enter coefficient b:
2
Enter coefficient c:
1
Roots are real and equal
Root = -1.0

⇒ Enter coefficient a:
1
Enter coefficient b:
1
Enter coefficient c:
1
There are no real solutions.

⇒ 1, 2, 3.
no real solutions.

WEEK 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.

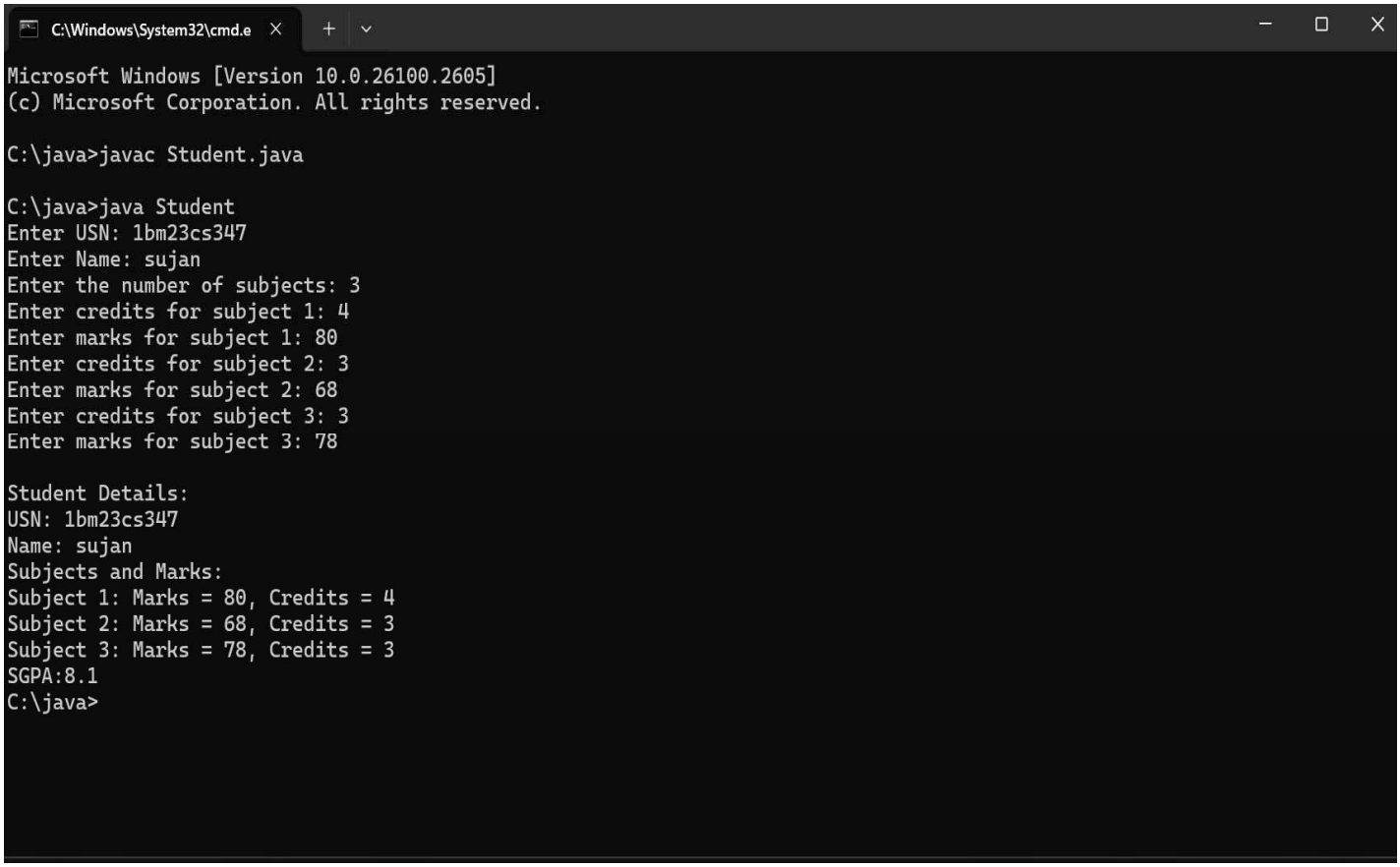
Source Code:

```
import java.util.Scanner; class
Student {    String usn;    String
name;    int numSubjects;    int[]
credits;    int[] marks;    double
sgpa;


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

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

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

      System.out.print("Enter the number of subjects: "); numSubjects = sc.nextInt();
       credits = new int[numSubjects]; marks =
new int[numSubjects];

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

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

   public void displayDetails() {
      System.out.println("\nStudent Details:");
```

```
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Subjects and Marks:");

        for (int i = 0; i < numSubjects; i++) {
            System.out.println("Subject " + (i + 1) + ": Marks = " + marks[i]
+ ", Credits = " + credits[i]);
        }
    }
         public void calculateSGPA() {         int
totalCredits = 0;         int totalGradePoints =
0;
         for (int i = 0; i < numSubjects; i++) {              int grade =
calculateGrade(marks[i]);              totalGradePoints += grade *
credits[i];              totalCredits += credits[i];
        }
         sgpa = (double) totalGradePoints / totalCredits;
    }     private int calculateGrade(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 void displaySGPA() {
        System.out.printf("SGPA:" + sgpa);
    }
```

```
   public static void main(String[] args) {        Student student = new Student();        student.acceptDetails();
student.displayDetails();        student.calculateSGPA();        student.displaySGPA();
   }
}
```

OUTPUT :



OBSERVATION:

Prgm 2 03/10/24

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

```
import java.util.Scanner;
public class SGPA_cal {
public static void main(String[] args){
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter name, USN ");
    System.out.println("Enter the number of subjects : ");
    int num = scanner.nextInt();

    double[] credits = new double[num];
    double[] grades = new double[num];

    for(int i=0; i<num; i++){
        System.out.println("Enter credit hours "+(i+1)+":");
        credits[i] = scanner.nextDouble();
        System.out.println("Enter grade "+(i+1)+" 0 to 10":);
        grades[i] = scanner.nextDouble();
    }
double totalcredits=0;
double gradesum=0;
```

```
for(int i=0; i<num; i++) {
        gradesum += grades[i] * credits[i];
        totalcredits += credits[i];
    }

double sgpa = gradesum/totalcredits;
System.out.println("SGPA is %.2f %n", sgpa);

scanner.close()
}
}
```

```
Enter USN: 1BM23CS292
Enter name: Sameksha
Enter the number of subjects: 4
Enter the credits for subject 1: 4
Enter the marks for subject 1: 10
Enter the credits for subject 2: 3
Enter the marks for subject 2: 9
Enter the credits for subject 3: 2
Enter the marks for subject 3: 9
Enter the credits for subject 4: 1
Enter the marks for subject 4: 10

SGPA is 9.50
```

WEEK 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.

Source Code:

```
import java.util.Scanner;
 class Book {     int
price;     String author;
String name;     int
pages;
    public Book(int price, String author, String name, int pages) {        this.price = price;
this.author = author;        this.name = name;        this.pages = pages;
   }
   public void setter() {
      System.out.println("enter the price,author,name and pages of the book");        Scanner sc = new
Scanner(System.in);        this.price=sc.nextInt();        this.author= sc.next();        this.name=sc.next();
this.pages=sc.nextInt();
   }
   public void getter() {
      System.out.println("Book Details:");
      System.out.println("Price:"+price);
      System.out.println("Author:"+author);
      System.out.println("Name:"+name);
      System.out.println("Pages:"+pages);
   }     public String toString() {              return "these are book
details";
   }
}
```

```
public class Pro {
   public static void main(String[] args) {        Scanner s1 = new
Scanner(System.in);
       System.out.println("enter the number of books");        int n = s1.nextInt();

      Book []b1 = new Book[n];
              for(int i=0;i<n;i++){            b1[i] = new Book(200,"sachin","The
Pride",111);            b1[i].getter();            b1[i].setter();            b1[i].getter();
       System.out.println(b1[i]);

      }
   }
}
```

OUTPUT:

```
enter the number of books
1
Book Details:
Price:200
Author:sachin
Name:The Pride
Pages:111
enter the price,author,name and pages of the book
150
virat
TheCentury
120
Book Details:
Price:150
Author:virat
Name:TheCentury
Pages:120
these are book details
```

OBSERVATION:

Prgm 3 19/10/24

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.

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
class Book {
    String name;
    String author;
    double price;
    int numPages;

    Book(String name, String author, double price,
        int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
void setName(String name) {
    this.name = name;
}
void setAuthor(String author) {
    this.author = author;
}
```

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

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

String getName () {
    return Name ;
}
String getAuthor () {
    return Author ;
}
int getNumPages () {
    returnnumPages ;
}

double getPrice () {
    return price;
}

Public String toString () {
    return " Book { " +
        " Name = " + name + ' \" ' +
        " Author = " + author + ' | " +
        " Price = " + price +
        " Number of pages = " + numPages
        '}'
    }
}
```

```
class BookStore {
    public static void main (String [] args ) {
        Scanner sc = new Scanner ( System.in ) ;
        List <Book> books = new ArrayList < > ( ) ;
        System.out.println (" Enter the no of books : ");
        int n = sc.nextInt ( ) ;
        sc.nextLine ( ) ;

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

        Book book = new Book ( name, author, price,
                               numPages ) ;
        books.add ( book ) ;
    }
System.out.println (" \n Book Details : " ) ;
    for ( Book book : books ) {
            System.out.println ( book ) ;
    }
sc.close ( ) ;
}
}
```

Enter the number of books : 2
Enter details for book 1:
Name : No time for goodbye
Author : Lindwood Barclay
Price : 450
No of Pages : 290

Enter details for book 2:
Name : Holes
Author : Louis Sachar
Price : 200
No of Pages : 250

Book Details :
Book { Name = "No time for goodbye", Author = "Lindwood Barclay", Price = 450.0, No of pages = 290 }

Book { Name = "Holes", Author = "Louis Sachar", Price = 200.0, No of Pages = 250 }

WEEK 4 :

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

Source Code:

```
abstract class Shape {     int
dim1;     int dim2;
       abstract void printArea();
}  class Rectangle extends Shape {     public Rectangle(int
length, int width) {         this.dim1 = length;         this.dim2 =
width;
    }
        void printArea() {
       int area = dim1 * dim2;
       System.out.println("Area of Rectangle: " + area);
    }
}
class Triangle extends Shape {
    public Triangle(int base, int height) {         this.dim1 =
base;         this.dim2 = height;
    }      void printArea() {
       double area = 0.5 * dim1 * dim2;
       System.out.println("Area of Triangle: " + area);
    }
}  class Circle extends Shape {
```

```
    public Circle(int radius) {        this.dim1 =
radius;        this.dim2 = 0;
    }
        void printArea() {
        double area = Math.PI * dim1 * dim1;
        System.out.println("Area of Circle: " + area);
    }
}
public class Main {
    public static void main(String[] args) {        Shape rectangle =
new Rectangle(8,9);
        Shape triangle = new Triangle(8, 6);
        Shape circle = new Circle(14);
         rectangle.printArea();
triangle.printArea();
circle.printArea();
    }
}
```

OUTPUT:

```
Area of Rectangle: 72
Area of Triangle: 24.0
Area of Circle: 615.7521601035994
PS C:\Users\satis\OneDrive\Documents\ooj_lab>
```

OBSERVATION:

Pgm: 4

24/10/24

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 contain only the method printArea() that prints the area of the given shape.

```
class Main{
    public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    System.out.print("choose a shape to calculate
                the area: (1: rectangle, 2: triangle,
                        3: circle): ");
    int choice = sc.nextInt();
    Shape shape = null;
    switch(choice){
    case 1:
    System.out.println("Enter width of the rectangle:");
    int width = sc.nextInt();
    System.out.println("Enter height of the rectangle");
    int height = sc.nextInt();
    shape = new Rectangle(width, height);
    break;

    case 2:
    System.out.println("Enter base of the triangle:
```

```
        int base=sc.nextInt();
        System.out.println("Enter height of the Triangle:");
        int height=sc.nextInt();
        shape=new Triangle(base, height);
        break;
      case 3:
        System.out.println("Enter radius of the circle:");
        int radius=sc.nextInt();
        shape=new Circle(radius);
        break;
      default
        System.out.println("Invalid choice");
        break;
    }
    if(shape!=null){
        shape.printArea();
    }
    scanner.close();
}
```

Choose a shape to calculate the area (1: Rectangle, 2: Triangle, 3: Circle): 1

Enter width of the rectangle: 12
Enter height of the rectangle: 44
Area of rectangle: 528

WEEK 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 facili es 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.

Source Code :

```
import java.util.Scanner;

class Account {
    String customerName;      int
accountNumber;      String
accountType;      double balance;
    public Account(String customerName, int accountNumber, String accountType) {
this.customerName = customerName;        this.accountNumber = accountNumber;        this.accountType =
accountType;        this.balance = 0.0;
    }
   public void deposit(double amount) {        if
(amount > 0) {            balance += amount;
          System.out.println("Amount deposited: " + amount);          System.out.println("Updated balance: " +
balance);
       } else {
          System.out.println("Invalid deposit amount!");
```

```
        }
    }
    public void displayBalance() {
        System.out.println("Balance: " + balance);
    }
}  class SavAcct extends Account {
private double interestRate;
    public SavAcct(String customerName, int accountNumber, double interestRate) {        super(customerName,
accountNumber, "Savings");         this.interestRate = interestRate;
    }      public void computeAndDepositInterest() {         double interest =
balance * (interestRate / 100);         balance += interest;
        System.out.println("Interest added: " + interest);         System.out.println("Updated balance: " + balance);
    }      public void withdraw(double amount) {         if
(amount <= balance) {             balance -= amount;
            System.out.println("Amount withdrawn: " + amount);             System.out.println("Updated balance: " +
balance);
        } else {
            System.out.println("Insufficient balance!");
        }
    }
}  class CurAcct extends Account {
double minimumBalance;     double
serviceCharge;
    public CurAcct(String customerName, int accountNumber, double minimumBalance, double serviceCharge)
{
        super(customerName,              accountNumber,              "Current");
this.minimumBalance = minimumBalance;              this.serviceCharge =
serviceCharge;
    }
```

```
   public void withdraw(double amount) {        if
(amount <= balance) {            balance -= amount;
          System.out.println("Amount withdrawn: " + amount);            if (balance <
minimumBalance) {                imposePenalty();
          }
          System.out.println("Updated balance: " + balance);
       } else {
          System.out.println("Insufficient balance!");
       }
   }
   private void imposePenalty() {        balance -=
serviceCharge;
       System.out.println("Balance fell below minimum. Service charge imposed: "
+ serviceCharge);
   }
}  public class Bank {
   public static void main(String[] args) {
       Scanner scanner = new Scanner(System.in);
       System.out.println("Choose account type:\n1. Savings Account\n2. Current Account");        int choice =
scanner.nextInt();        scanner.nextLine();

       System.out.println("Enter customer name: ");
       String name = scanner.nextLine();
       System.out.println("Enter account number: ");        int accNum =
scanner.nextInt();

       if (choice == 1) {
          System.out.println("Enter interest rate for savings account: ");            double interestRate =
scanner.nextDouble();
          SavAcct savAccount = new SavAcct(name, accNum, interestRate);
          System.out.println("Enter amount to deposit: ");            double deposit =
scanner.nextDouble();            savAccount.deposit(deposit);

          savAccount.computeAndDepositInterest();
          System.out.println("Enter amount to withdraw: ");
```

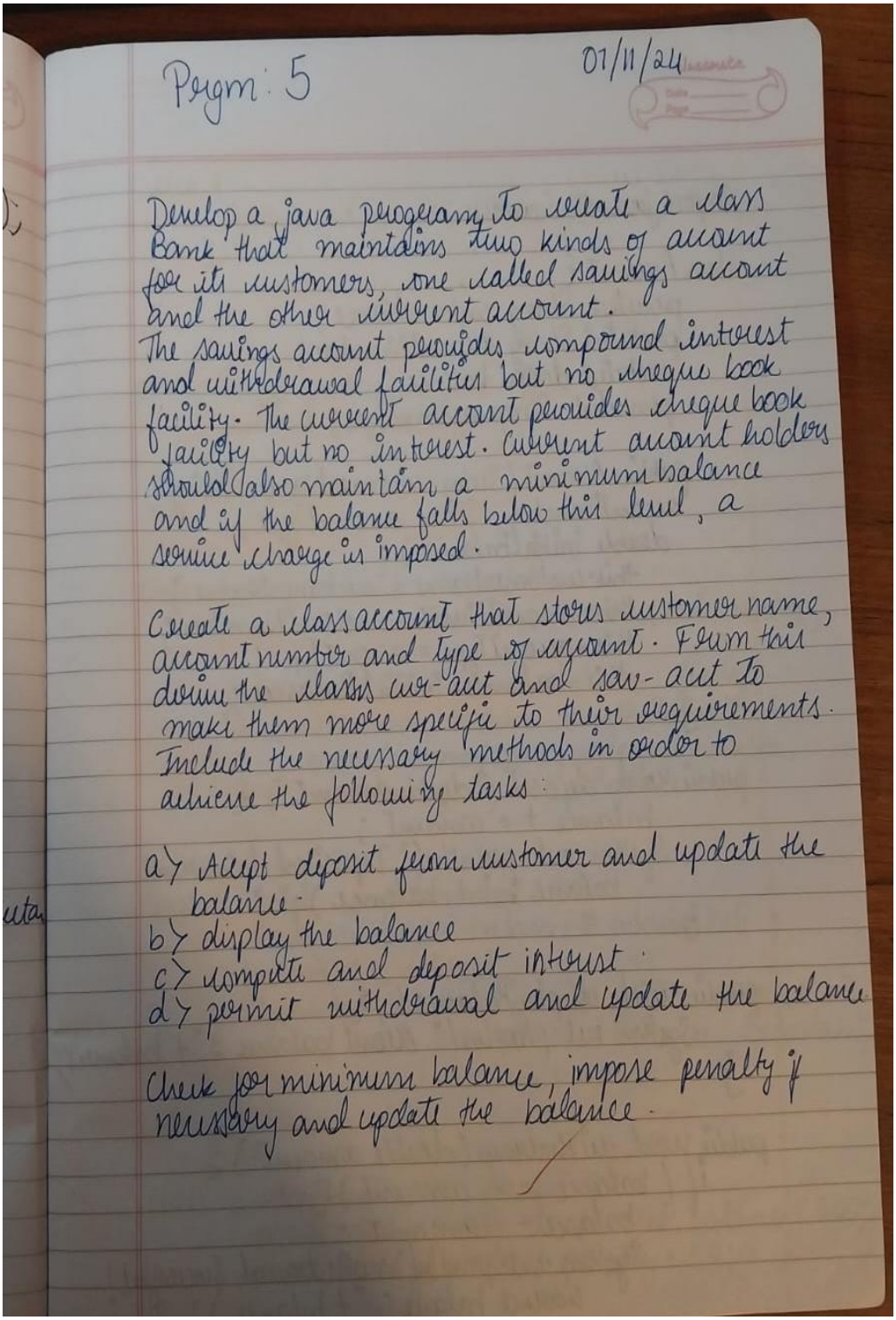
```
            double withdrawAmount = scanner.nextDouble();
            savAccount.withdraw(withdrawAmount);
        } else if (choice == 2) {
            System.out.println("Enter minimum balance for current account: ");            double minBalance =
scanner.nextDouble();
            System.out.println("Enter service charge for falling below minimum balance: ");
            double serviceCharge = scanner.nextDouble();
            CurAcct curAccount = new CurAcct(name, accNum, minBalance, serviceCharge);

            System.out.println("Enter amount to deposit: ");            double deposit =
scanner.nextDouble();            curAccount.deposit(deposit);

            System.out.println("Enter amount to withdraw: ");            double withdrawAmount =
scanner.nextDouble();            curAccount.withdraw(withdrawAmount);

        } else {
            System.out.println("Invalid account type selected.");
        }
        scanner.close();
    }
}
```

Output :

```
Choose account type:
1. Savings Account
2. Current Account
1
Enter customer name:
sagar
Enter account number:
1234
Enter interest rate for savings account:
3
Enter amount to deposit:
5000
Amount deposited: 5000.0
Updated balance: 5000.0
Interest added: 150.0
Updated balance: 5150.0
Enter amount to withdraw:
4800
Amount withdrawn: 4800.0
Updated balance: 350.0
```

```
Choose account type:
1. Savings Account
2. Current Account
2
Enter customer name:
chetan
Enter account number:
9876
Enter minimum balance for current account:
1000
Enter service charge for falling below minimum balance:
150
Enter amount to deposit:
6000
Amount deposited: 6000.0
Updated balance: 6000.0
Enter amount to withdraw:
5200
Amount withdrawn: 5200.0
Balance fell below minimum. Service charge imposed: 150.0
Updated balance: 650.0
```

OBSERVATION:



Pgm: 5 01/11/24

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

```
import java.util.Scanner;

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

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

public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited! Current
            balance: " + balance);
    }

public void displayBalance() {
    System.out.println("Account balance: " + balance
}

public void withdraw(double amount) {
    if (balance >= amount) {
        balance -= amount;
        System.out.println("withdrawal Successful!
            current balance: " + balance); }
```

```
        else {
            System.out.println("Insufficient balance!");
        }
    }

    public String getAccountType() {
        return accountType;
    }
}

class SavAcct extends Account {
    private static final double interestRate = 0.04;

    public SavAcct(String customerName, String
    accountNumber, double initialBalance) {
        super(customerName, accountNumber,
            "Savings", initialBalance);
    }

    public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest of " + interest + "
            has been added. New Balance: " + balance);
    }
}

class curAcct extends Account {
    private static final double min_balance = 500;
    private static final double penalty = 50;
```

```
publi curAct (String customerNam, String accountNumb
    double initialBalance) {
        super(customerName, accountNumber,
            "current", initialBalance);
    }
publi void checkminimumBalance(){
    if(balance < min_balance) {
            balance -= penalty;
            System.out.println("Balance is below
                minimum. penalty added/deduct
            new balance: " + balance);
        }
    }
}

publi class Bank1 {
    publi static void main(String[] args) {
        Scanner scanner=new scanner(System.in);

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account type (1 for
            Savings, 2 for Current): ");
        int accountChoice = scanner.nextInt();

        scanner.nextLine();
        System.out.print("Enter account number: ");
        String accountNumber = scanner.nextLine();

        Account account = null;
```

```
if (accountChoice == 1) {
    System.out.print("Enter initial deposit for
        Savings account : ");
    double initialDeposit = scanner.nextDouble();
    account = new SavAcct(customerName,
    accountNumber, initialDeposit);
} else if (accountChoice == 2) {
    System.out.print("Enter initial deposit for
    current account : ");
    double initialDeposit = scanner.nextDou
    account = new CurAcct(customerName, accountNumb
        initialDeposit);
} else {
    System.out.println("Invalid");
    return;
}

boolean running = true;
while (running) {
    System.out.println("\n Bank Operations : ");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    System.out.println("4. Compute Interest");
    System.out.println("5. Check and apply
        minimum balance penalty");
    System.out.println("6. Exit");
    System.out.print("Enter your choice : ");
    int choice = scanner.nextInt();

    switch (choice) {
```

```
case 1:
    System.out.print("Enter deposit amount:");
    double depositAmount = scanner.nextDouble();
    account.deposit(depositAmount);
    break;

case 2:
    System.out.print("Enter withdrawal amount:");
    double withdrawAmount = scanner.nextDouble();
    account.withdraw(withdrawAmount);
    break;

case 3:
    account.displayBalance();
    break;

case 4:
    if (account instanceof SavAct) {
        ((SavAct)account).computeInterest();
    } else {
        System.out.println("Interest can be
calculated only for savings account.");
    }
    break;

case 5:
    if (account instanceof CurAct) {
        ((CurAct)account).checkMinimumBalance();
    } else {
        System.out.println("minimum balance
check can only be applied to current
account.");
    }
    break;
```

```
            case 6 :
                System.out.println("Exiting program ");
                running = false;
                break;
            default :
                System.out.println("Invalid!");

        }
    }
    scanner.close();
}
}
```

```
Enter customer name : Sameksha
Enter accountType (1 for savings, 2 for current):
1
Enter account number : 2345SE59
Enter initial deposit for savings account : 5000
Bank Operations :
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest
5. Check and apply balance penalty
6. Exit
Enter your choice :
1
Enter deposit amount : 3000
Deposited ! current balance : 8000.0
```

Me

Me

2
Enter withdrawal amount : 4000
withdrawal successful! balance : 4000.0

Me

3
Account balance : 4000.0

Me
4
Interest of 160.0 has been added!
New balance : 4160.0

Me
5
minimum balance check can only be applied to current account.

Me
2
Enter withdrawal amount : 5000
Insufficient balance! withdrawal failed!

Me
6.
Exited!

=> Enter initial deposit for current account : 50000
Me
1
Enter deposit amount : 100
Deposited ! current balance : 50100.0
Me
2
Enter withdrawal amount : 50000
withdrawal successfull ! current balance : 100

Me
5

Balance is below minimum. A penalty has been charged / deducted. New balance : 50

Me *
6
Exited !

21/11/24

WEEK 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.

Source Code :

```
import CIE.Internals; import
SEE.External; import
java.util.Scanner;
 public class Studentmarks {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);



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



      Internals[] cieStudents = new Internals[n];
      External[] seeStudents = new External[n];
            for (int i = 0; i < n; i++) {

         System.out.println("Enter details for CIE Student " + (i + 1) + ":
");
         System.out.print("USN: ");
         String usn = scanner.nextLine();
         System.out.print("Name: ");
         String name = scanner.nextLine();
System.out.print("Semester: ");            int sem =
scanner.nextInt();           int[] internalMarks = new int[5];
         System.out.println("Enter internal marks for 5 courses: ");           for (int j = 0; j < 5; j++)
{               internalMarks[j] = scanner.nextInt();
```

```
        }            cieStudents[i] = new Internals(usn, name, sem, internalMarks);
scanner.nextLine();



        System.out.println("Enter details for SEE Student " + (i + 1) + ":
");
        System.out.print("USN: ");            usn =
scanner.nextLine();            System.out.print("Name: ");
name = scanner.nextLine();
System.out.print("Semester: ");            sem =
scanner.nextInt();
        int[] externalMarks = new int[5];
        System.out.println("Enter external marks for 5 courses: ");            for (int j = 0; j < 5; j++) {
externalMarks[j] = scanner.nextInt();
        }            seeStudents[i] = new External(usn, name, sem, externalMarks);
scanner.nextLine();
    }



    System.out.println("\nFinal Marks for all students:");
     for (int i = 0; i < n; i++) {
                    cieStudents[i].displayStudentDetails();
cieStudents[i].displayInternalMarks();

                    seeStudents[i].displayStudentDetails();
seeStudents[i].displayExternalMarks();

                    int[] internalMarks = cieStudents[i].getInternalMarks();            int[]
externalMarks = seeStudents[i].getExternalMarks();            int[] finalMarks = new int[5];
        for (int j = 0; j < 5; j++) {
          finalMarks[j] = internalMarks[j] + externalMarks[j];            }



        System.out.print("Final Marks: ");
```

```
            for (int mark : finalMarks) {
                System.out.print(mark + " ");
            }
            System.out.println("\n");
        }
        scanner.close();
    }
}
```

```
package CIE;
 public class Internals extends Student {
        private int[] internalMarks = new int[5];
         public Internals(String usn, String name, int sem, int[] internalMarks) {         super(usn, name,
sem); // Call parent constructor         this.internalMarks = internalMarks;
    }

    public void displayInternalMarks() {
System.out.print("Internal Marks: ");         for (int mark :
internalMarks) {
            System.out.print(mark + " ");
        }
        System.out.println();
    }

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

```
package CIE;  public class Student
{      protected String usn;
protected String name;
```

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

    public void displayStudentDetails() {
        System.out.println("USN: " + usn + ", Name: " + name + ", Semester: " + sem);     }
}
```

```
package SEE;
 import CIE.Student;
 public class External extends Student {     private int[]
externalMarks = new int[5];
     public External(String usn, String name, int sem, int[] externalMarks) {         super(usn, name, sem);
this.externalMarks = externalMarks;
    }
    public void displayExternalMarks() {
System.out.print("External Marks: ");         for (int mark :
externalMarks) {
            System.out.print(mark + " ");
        }
        System.out.println();
    }      public int[] getExternalMarks() {
return externalMarks;
    }
}
```

Output :

```
Enter number of students: 2
Enter details for CIE Student 1:
USN: 1
Name: sagar
Semester: 2
Enter internal marks for 5 courses:
38 40 41 45 46
Enter details for SEE Student 1:
USN: 1
Name: sagar
Semester: 2
Enter external marks for 5 courses:
39 42 45 50 48
Enter details for CIE Student 2:
USN: 2
Name: chetan
Semester: 3
Enter internal marks for 5 courses:
40 44 46 47 50
Enter details for SEE Student 2:
USN: 2
Name: chetan
Semester: 3
Enter external marks for 5 courses:
40 44 46 47 50

Final Marks for all students:
USN: 1, Name: sagar, Semester: 2
Internal Marks: 38 40 41 45 46
USN: 1, Name: sagar, Semester: 2
External Marks: 39 42 45 50 48
Final Marks: 77 82 86 95 94

USN: 2, Name: chetan, Semester: 3
Internal Marks: 40 44 46 47 50
USN: 2, Name: chetan, Semester: 3
External Marks: 40 44 46 47 50
Final Marks: 80 88 92 94 100
```

OBSERVATION:

14/11/24

Pgm: 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 package in a file that declares the final marks of n students in all five courses.

```
package CIE ;
import java.util.*;
public class Student
{
 public String USN;
 public String Name;
 public int sem;

 Scanner SC = new Scanner( System.in );

public void getd(){
  System.out.println();
  SC.nextLine();
 System.out.println("Enter USN: ");
 USN = SC.nextLine();
```

```
        System.out.println("Enter name : ");
        Name = sc.nextLine();
        System.out.println("Enter sem : ");
        Sem = sc.nextInt();
    }
    public void putd(){
        System.out.println(" USN : " + USN);
        System.out.println(" Name : " + Name);
        System.out.println(" sem : " + Sem);
    }
}
```

```
package CIE;
import java.util.*;
public class Internals;
Scanner sc = new Scanner(System.in);
public int marks[] = new int[5];
public void getd(){
  System.out.println("Enter CIE marks:");
  for(int i=0; i<5; i++){
      (marks[i] = sc.nextInt();
  }
}
public void putd(){
    for(int i=0; i<5; i++){
        System.out.println("(marks[i] + " ");
    }
}
}
```

```
package SEE
import CIE.student;
import java.util.*;
publi class External extends student
{
    Scanner sc = new scanner(System.in);
    publi int smarks[] = new int[5];
    publi void getd(){
        System.out.println("Enter SEE marks ");
        for(int i=0; i<5; i++){
            Smarks[i] = sc.nextInt();
        }
    }
public void putd(){
    for(int i=0; i<5; i++){
        System.out.println(Smarks[i] + " ");
    }
}
publi int finalmarks(int x, int Smark){
    return (marks + (Smark[x]/2));
}
}

import java.util.*;
import CIE.*;
import SEE.*;
class Main{
 publi static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter number of student");
    int n = sc.nextInt();
    Student s = new Student();
```

```
Internals I[] = new Internals[n];
External E[] = new External[n];
for(int i=0; i<n; i++) {
    s.getId();
    s.putId();
    I[i] = new Internals();
    I[i].getId();
    I[i].putId();
    E[i] = new External();
    E[i].getId();
    E[i].putId();

    System.out.println("FinalMarks");
    for(int j=0; j<5; j++) {
        int finalmarks = E[i].getfinalmarks(
            j, I[i].(marks[j]));

        System.out.println(finalmarks);
    }
}
}
```

Enter number of Student
2

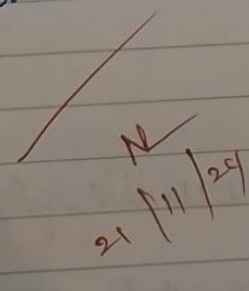
Enter USN:
1BM23CS292
Enter name:
Samrudha

Enter Sem:
3

USN: 1BM23CS292
Name: Samarth
Sem: 3
Enter CIE marks:
20
21
22
23
24
Enter SEE marks
30
31
32
33
34
Final marks
35
36
38
39
41

Enter USN:
1BM23AI018
Enter Name:
Sahana
Enter Sem:
3

USN : 1BM23AI058
Name : Sahana
Sem : 3
Enter CIE marks :
30
31
32
33
34
Enter SEE mark
30
31
32
33
34
~~33~~
Final marks
60
62
64
66
~~70~~ 68



WEEK 7 :

Write a program that demonstrates handling of excep ons 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 excep on WrongAge( ) when the input age<0. In Son class, implement a constructor that uses both father and son’s age and throws an excep on if son’s age is >=father’s age.

Source Code :

```
import java.util.Scanner;

class WrongAgeException extends Exception {     public
WrongAgeException(String message) {         super(message);
    }
}  class SonAgeException extends Exception {     public
SonAgeException(String message) {         super(message);
    }
}  class Father {      int age;     public Father(int age) throws
WrongAgeException {         if (age <= 0) {             throw new
WrongAgeException("Wrong age");
        }
        this.age = age;
    }     public int getAge() {
return age;
    }
}  class Son extends Father {      int
sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAgeException, SonAgeException {
super(fatherAge);         if (sonAge >= fatherAge) {
```

```
            throw new SonAgeException("Son's age cannot be greater than or equal to father's age");
        }        if(sonAge <= 0){        throw new
WrongAgeException("Wrong age");
        }        this.sonAge = sonAge;
    }    public int getSonAge() {
return sonAge;
    }
}

public class FatherSon{
    public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);            System.out.print("Enter
Father's Age: ");            int fatherAge = sc.nextInt();
            System.out.print("Enter Son's Age: ");            int sonAge =
sc.nextInt();            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Accepted Succesfully");
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }

        }
    }
```

Output :

```
Enter Son's Age: 26
Accepted Succesfully
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac FatherSon.
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java FatherSon
Enter Father's Age: 30
Enter Son's Age: 32
Son's age cannot be greater than or equal to father's age
```

```
Enter Father's Age: 30
Enter Son's Age: 0
Wrong age
```

OBSERVATION:

21/11/24

Pgm 7

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 uses both father and son's age and throws an exception if son's age is >= father's age.

```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}
class Father {
    protected int age;
    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age
                cannot be negative");
        }
        this.age = age;
    }
}
class Son extends Father {
    private int sonAge;

    public Son(int fatherAge, int sonAge) throws WrongAgeExc
```

```
        super(fatherAge);
        if(sonAge<0){
            throw new WrongAgeException("Son's age
                                cannot be negative");
        }
        if(sonAge >= fatherAge){
            throw new WrongAgeException("Son's age
                cannot be greater than or equal to
                father's age");
        }
        this.sonAge = sonAge;
    }

    public String toString(){
        return "Father's Age: "+age+"Son's Age"+sonAge;
    }
}

public class ExceptionInheritanceDemo{
    public static void main(String[] args){
        try{
            Father father = new Father(45);
            System.out.println("Father created with
                                age: "+fatherAge);
            Son son = new Son(45, 20);
            System.out.println(son);
        }catch(WrongAgeException e){
            System.out.println("Exception occured"+
                                e.getMessage());
        }
```

```
        try {
            Son invalidSon = new Son(40,40);
        } catch(WrongAgeException e) {
            System.out.println("Exception occured "+e.getMe
        }
    try {
        Father invalidFather = new Father(-5);
    } catch(WrongAgeException e){
        System.out.println("Exception occured "+e.getMessage
    }
}
}
```

Father created with age : 45
Father's Age : 45, Son's Age : 20
Exception occured : Son's age cannot be greater than or equal to father's age.
Exception occured : Fathers age cannot be negative.

28/11/24

WEEK 8 :

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.

Source Code :

```
class ThreadDemo extends Thread{     public
void run(){     while(true){
    System.out.println("BMS College Of Engineering");     try{
    Thread.sleep(10000);
    }
    catch(InterruptedException e){
    e.printStackTrace();
    }
     }
  }
}  class CSEThread extends Thread{
public void run(){     while(true){
    System.out.println("CSE");     try{
    Thread.sleep(2000);
    }     catch(InterruptedException e){
    e.printStackTrace();
    }
     }
  }
}  public class Demo{
    public static void main(String[] args){
    ThreadDemo t1 = new ThreadDemo();
CSEThread t2 = new CSEThread();     t1.start();
t2.start();
     }
 }
```

Output :

```
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
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
```

OBSERVATION:

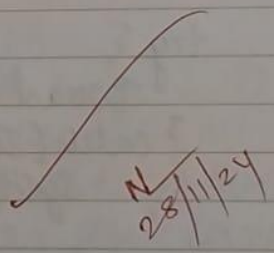
28/11/24

Pgm 8

Write a program which creates Two threads, one thread displaying "BMS College of Engineering" one every ten seconds and another displaying "CSE" one every two seconds.

```
public class Threadbms {
    public static void main(String[] args) {
        Thread thread1 = new Thread( () -> {
            while (true) {
                System.out.println("BMS College of
                                    Engineering");
                try {
                    Thread.sleep(10000);
                } catch (InterruptedException e) {
                    System.out.println("Thread Interrupted: " +
                                        e.getMessage());
                }
            }
        });
        Thread thread2 = new Thread( () -> {
            while (true) {
                System.out.println("CSE");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    System.out.println("Thread Interrupte
                                        + e.getMessage());
                }
            }
        });
        thread1.start();
        thread2.start();
    }
}
```

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
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE



28/11/24