WEEK 1:

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.

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

  }
}
```

Output:

```
PS C:\Users\satis\OneDrive\Desktop\project> javac Main.java
PS C:\Users\satis\OneDrive\Desktop\project> java Main
enter the values of a,b, and c
2 3 6
-39.0
the solutions are
There are no real roots
PS C:\Users\satis\OneDrive\Desktop\project> javac Main.java
PS C:\Users\satis\OneDrive\Desktop\project>  java Main
enter the values of a,b, and c
1 2 1
0.0
the solutions are
roots are equal
-1.0
PS C:\Users\satis\OneDrive\Desktop\project> javac Main.java
PS C:\Users\satis\OneDrive\Desktop\project> java Main
enter the values of a,b, and c
1 5 3
13.0
the solutions are
roots are unique
-0.6972243622680054 -4.302775637731995
```

Written Code & Output:

WEEK - 1

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

```
import java.util.Scanner;
class Quadratic{
    float d;
    Scanner sc = new Scanner(System.in);

Void solver()
 {
    System.out.println("enter the values for 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("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 Main{
    public static void main (String[] args){
        Quadratic q1 = new Quadratic();
        q1.solver();
    }
}
```

Output:

1) enter the values of a, b, and c
2 3 6
-39.0
the solutions are
There are no real roots.

2) Enter the values of a, b, c
1
2
1
0.0
the solutions are
roots are equal
-1.0

3) Enter the values of a, b and c
1 5 3
13.0
the solutions are
roots are unique
-0.6972243622680054
-4.302775637731995

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:

```
C:\1BM23CS306>javac Student.java

C:\1BM23CS306>java Student
Enter USN: 1bm23cs306
Enter Name: sagar
Enter the number of subjects: 6
Enter credits for subject 1: 4
Enter marks for subject 1: 95
Enter credits for subject 2: 3
Enter marks for subject 2: 85
Enter credits for subject 3: 3
Enter marks for subject 3: 75
Enter credits for subject 4: 4
Enter marks for subject 4: 88
Enter credits for subject 5: 2
Enter marks for subject 5: 60
Enter credits for subject 6: 1
Enter marks for subject 6: 97

Student Details:
USN: 1bm23cs306
Name: sagar
Subjects and Marks:
Subject 1: Marks = 95, Credits = 4
Subject 2: Marks = 85, Credits = 3
Subject 3: Marks = 75, Credits = 3
Subject 4: Marks = 88, Credits = 4
Subject 5: Marks = 60, Credits = 2
Subject 6: Marks = 97, Credits = 1
SGPA:8.882352941176471
```

Written Code & Output :

WEEK - 2

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

```
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.println("Enter name:");
        name = sc.nextLine();
    System.out.println("Enter usn:");
        usn = sc.nextLine();
    System.out.println("Enter no. of
        subjects:");
        numsubjects = sc.nextInt();

    credits= new credits[numsubjects];
    marks = new marks[numsubjects];

    for(int i=0; i<numsubjects; i++){
    System.out.print("Enter credets for subject"
        + (i+1)+":");
        credits[i] = sc.nextInt();
    System.out.print("Enter marks for subject"
        + (i+1)+":");
2       marks[i] = sc.nextInt();}
```

```
public void displayDetails(){
    System.out.println("Student 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];
        totarCredits += 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.println("SGPA of the student is:"
        + sgpa);
    }
}
public stat

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

WEEK - 2

Output

Enter USN: 4bm23cs306
Enter Name: sagar
Enter the number of subjects 6
Enter credits for subject 1: 4
Enter marks for subject 1: 95
Enter credits for subject 2: 3
Enter marks for subject 2: 85
Enter credits for subject 3: 3
Enter marks for subject 3: 75
Enter credits for subject 4: 4
Enter marks for subject 4: 88
~~Enter marks for 8~~
Enter credits for subject 5: 2
Enter marks for subject 5: 60
Enter credits for subject 6: 1
Enter marks for subject 6: 97

Student Details:
USN: 4bm23cs306
Name: sagar
Subjects and Marks:
Subject 1: Marks = 95, Credits = 4
Subject 2: Marks = 85, Credits = 3
Subject 3: Marks = 75, Credits = 3
Subject 4: Marks = 88, Credits = 4
Subject 5: Marks = 60, Credits = 2
Subject 6: Marks = 97, Credits = 1

SGPA: 8.882352941176471

28/11/24

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";
    }
}
```

```java
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:

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac Pro.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java Pro
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
```

Written Code & Output :

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 books.

→

```
Class Book {
    String name;
    String author;
    int price;
    int num_pages;

    Book (String name, String author, int price,
          int num_pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }
Scanner sc = new Scanner (System.in);

    void set()
    {
        name = sc.next();
        author = sc.next();
        price = sc.nextInt();
        num_pages = sc.nextInt();
```

```
void display()
{
  sout("The book details are");
  System.out.println(name);
  System.out.println(author);
  System.out.println(price);
  System.out.println(num_pages);
}

public string tostring()
  {
    System.out.println("The book details are
      displayed by tostring method");

      System.out.println(name);
      System.out.println(author);
      System.out.print(price);
      System.out.print(num_pages);
  }
}
Class main
  {
    public static void main(string[] args)
  {
    Scanner S1 = new Scanner(System.in);

    System.out.println("enter the number of
      objects");
    int n = S1.nextInt();

    Book[] b1 = new Book[n];
    for(int i=0; i<n; i++){
      b1[i] = new Book(200, "sachin",
        "The Pride", 120);
      b1[i].display();
```

```
        b1[i].display();
        b1[i].set();
        b1[i].display();

        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.
```

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:

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

Written Code & Output :

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.

```
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 triagle(int base, int height)
    {
        this.dim1 = base; this.dim2 = height;
    }
```

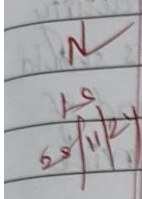
```
void printArea(){
    double area = 0.5 * dim1 * dim2;
    System.out.println("Area of Triangle:" + area);
  }
}

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)
  {
    Rectangle rectangle1 = new Rectangle(8,9);

    Triangle triangle1 = new Triangle(8,6);

    Circle circle1 = new Circle(14);

    rectangle1.printArea();
    triangle1.printArea();
    circle1.printArea();
}
```

Output:

Area of rectangle: 72

Area of triangle: 24

Area of circle: 615.7521601035994.



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 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.

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();
```

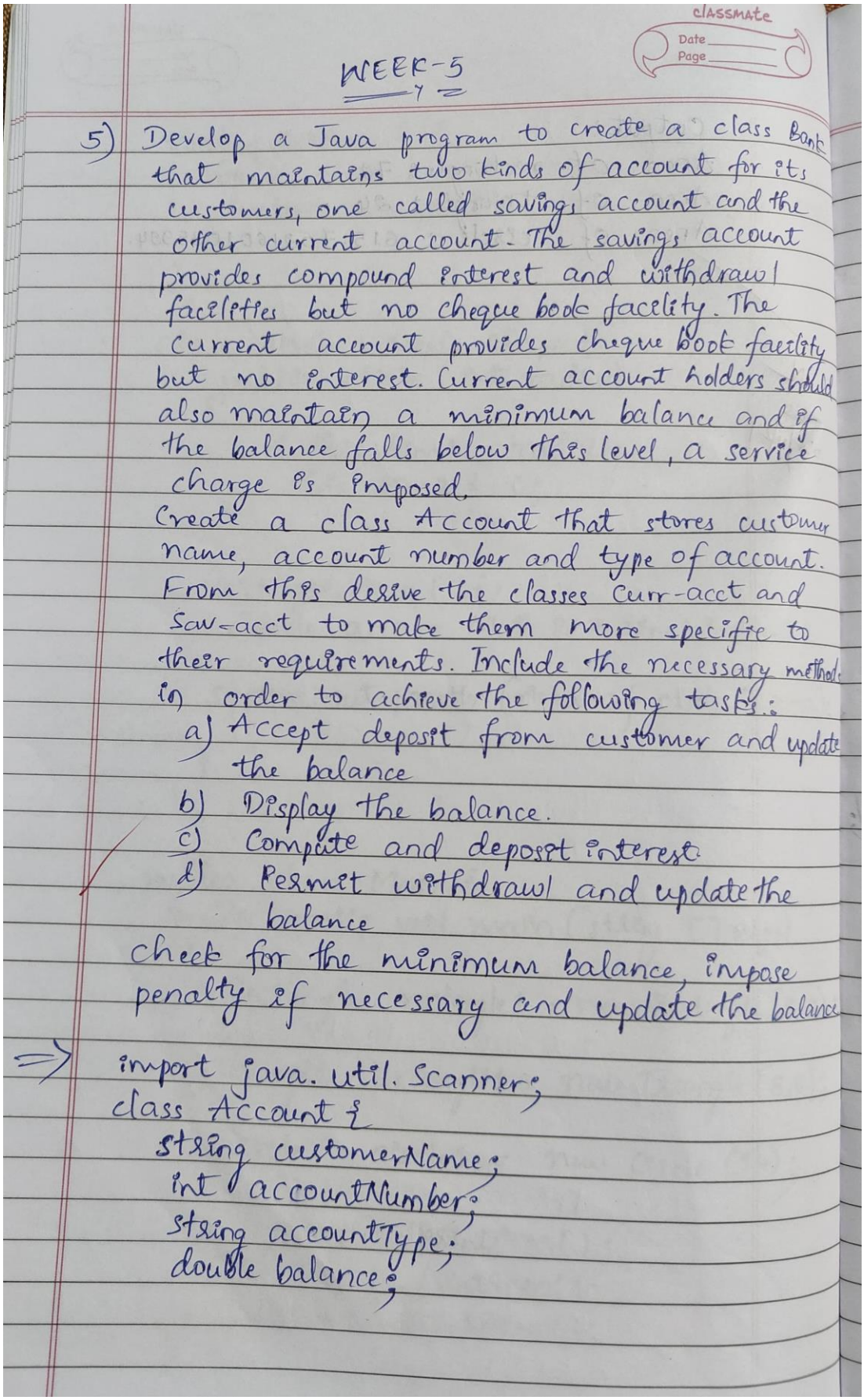
```
            System.out.println("Enter service charge for falling below minimum
balance: ");
            double serviceCharge = scanner.nextDouble();
            CurAcct curAccount = new CurAcct(name, accNum, minBalance,
serviceCharge);

            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 :

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac Bank.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java Bank
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
```

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac Bank.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java Bank
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
```

Written Code & Output :



WEEK-5

5) Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.
Create a class Account that stores customer name, account number and type of account. From this derive the classes Curr-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

=>

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

```
public Account (String customerName, int accountNum
- ber, 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;
        S.O.P ("Amount deposited:" + amount);
        S.O.P ("Updated balance:" + balance);
    } else {
        S.O.P ("Invalid deposit amount!");
    }
}
public void displayBalance() {
    S.O.P ("Baldnce:" + balance);
}
}
class SavingsAcct extends Account {
    private double interestRate;
public SavingsAcct(String customerName, int
    accountNumber, double interestRate) {
    super (customerName, accountNumber);
    this. interestRate = interestRate;
}
public void computeAndDepositInterest() {
    double interest = balance * (interestRate/100);
        balance += interest;
    S.O.P ("Interest added:" + interest);
    S.O.P ("Updated balance:" + balance);
}
```

```
public void withdraw (double amount){
    if (amount <= balance){
        balance -= amount;
        S.O.P ("Amount withdrawn:" + amount);
        S.O.P ("Updated balance:" + balance);
    }else{
        S.O.P ("Insufficient balance");
    }
  }
}

class CurrentAcct extends Account{
    double minimumBalance;
    double serviceCharge;

    public CurrentAcct (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;
    S.O.P(" Amount withdrawn:" + amount);
    if (balance < minimumBalance){
        imposePenalty();
    }
    S.O.P ("Updated balance:" + balance);
```

```
} else {
    s.o.p ("Insufficient balance!");
  }
}

private void imposePenalty(){
    balance -= serviceCharge;
    s.o.p ("Balance fell below minimum. Service
    charge imposed:" + serviceCharge);
  }
}

public class Bank {
  public static void main (String[] args){
  Scanner sc = new Scanner (System.in);
  s.o.p ("Choose account type:\n1. Savings
  Account\n 2. Current Account");
  int choice = sc.nextInt();
  sc.nextLine();

  s.o.p ("Enter customer name:");
  String name = sc.nextLine();
  s.o.p ("Enter account number:");
  int accNum = sc.nextInt();
  if (choice == 1){
    s.o.p ("Enter interest rate for savings account:");
  double interestRate = sc.nextDouble();
  SavingsAcct savAccount = new SavingsAcct(name,
  accNum, interestRate);

  s.o.p ("Enter amount to deposit:");
  double deposit = sc.nextDouble();
```

```
        savAccount.computeAndDepositInterest();
        s.o.p("Enter amount to withdraw:");
        double withdrawAmount = sc.nextDouble();

        savAccount.withdraw(withdrawAmount);

    } else if (choice == 2) {
        s.o.p("Enter minimum balance for current
        account:");

        double minBalance = sc.nextDouble();
        s.o.p("Enter service charge for falling below
    minimum balance:");
        double serviceCharge = sc.nextDouble();

    CurrentAcct curAccount = new CurrentAcct(name,
    accNum, minBalance, serviceCharge);

        s.o.p("Enter amount to deposit:");
    double deposit = sc.nextDouble();
        curAccount.deposit(deposit);

        s.o.p("Enter amount to withdraw:");
    double withdrawAmount = sc.nextDouble();

        curAccount.withdraw(withdrawAmount);

    } else {
        s.o.p("Invalid account type selected.");
    }

        sc.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: 5000.0
Updated balance : 5000.0
Interest added : 150.0
Updated balance : 5150.0
Enter amount to withdraw: 4800.0
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: 6000.0
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

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 :

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac Studentmarks.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java Studentmarks
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
```

Written Code & Output :

WEEK - 6

6) Create a package CIE which has two classes Student and Internals. The class student has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import two packages in a file that declares the final marks of n students in all five courses.

⇒

```
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(){
    S.O.P("USN:"+usn+", Name:"+name+", Semester:"+
    sem);
```

P.T.O

```
package CIE;
public class Internals extends student {
    psivate int[] internalMarks = new int[5];

    public Internals (string usn, string name, int
        sem, int[] internalMarks) {
        super (usn, name, sem);
        this. internalMarks = internalMarks;
    }

    public void displayInternalMarks() {
        s.o.p ("Internal Marks:");
        for (int mark : internalMarks) {
            s.o.p (mark + "  ");
        }
        s.o.p();
    }

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

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(){
        s.o.p ("External Marks:");
        for(int mark: externalMarks){
            s.o.p (mark+" ");
        }
        s.o.p();
    }
    public int[] getExternalMarks(){
        return externalMarks;
    }
}

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

public class Studentmarks{
    public static void main (string[] args){

    Scanner sc = new Scanner(System.in);

    s.o.p ("Enter number of students:");
    int n = sc.nextInt();
    sc.nextLine();

    Internals[] cieStudents = new Internals[n];
    Externals[] seeStudents = new External[n];

    for(int i=0; i<n; i++){
        s.o.p ("Enter detail of the student for CIE"+
        (i+1)+":");
        s.o.p ("USN:");
```

```
String usn = sc.nextLine();
s.o.p ("Name:");
String name = sc.nextLine();
s.o.p ("Semester:");
int sem = sc.nextInt();
int[] internalMarks = new int[5];
s.o.p ("Enter internal marks for 5 courses:");
for (int j=0; j<5; j++){
    internalMarks[j] = sc.nextInt();
}
cieStudents[i] = new Internals (usn, name, sem,
internalMarks);
sc.nextLine();

s.o.p ("Enter details of the student for SEE"
(i+1)+":");

s.o.p ("USN:");
usn = sc.nextLine();
s.o.p ("Name:");
name = sc.nextLine();
s.o.p ("Semester:");
sem = sc.nextInt();
int[] externalMarks = new int[5];
s.o.p ("Enter external marks for five courses:");

for (int j=0; j<5; j++){
    externalMarks[j] = sc.nextInt();
}
seeStudents[i] = new External (usn, name, sem,
externalMarks);
sc.nextLine();
}
```

```
S.O.p ("\n Final Marks for all students:");
for (int i=0; i<n; i++){
  cieStudents[i].displayStudentDetails();
  seeStudents[i].display
  cieStudents[i].displayInternalMarks();

  seeStudents[i].displayStudentDetails();
  seeStudents[i].displayExternalMarks();
int[] internalMarks = cieStudents[i].getInternal
                                    Marks();
int[] externalMarks = seeStudents[i].getExternalMarks();
int[] finalMarks = new int[5];

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

  finalMarks[j] = internalMarks[j] + externalMarks[j];
}
  S.O.p ("Final Marks:");
  for (int mark : finalMarks){
    S.O.p (mark + " ");
  }
  S.O.p("\n");
}
 sc.close();
}
}
```

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 42 43 44 45
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
Final Marks: 77, 82 86 95 94

USN: 2 Name: chetan Semester: 3
Final Marks: 80 88 92 94 100

WEEK 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.

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());
            }

       }
  }
```

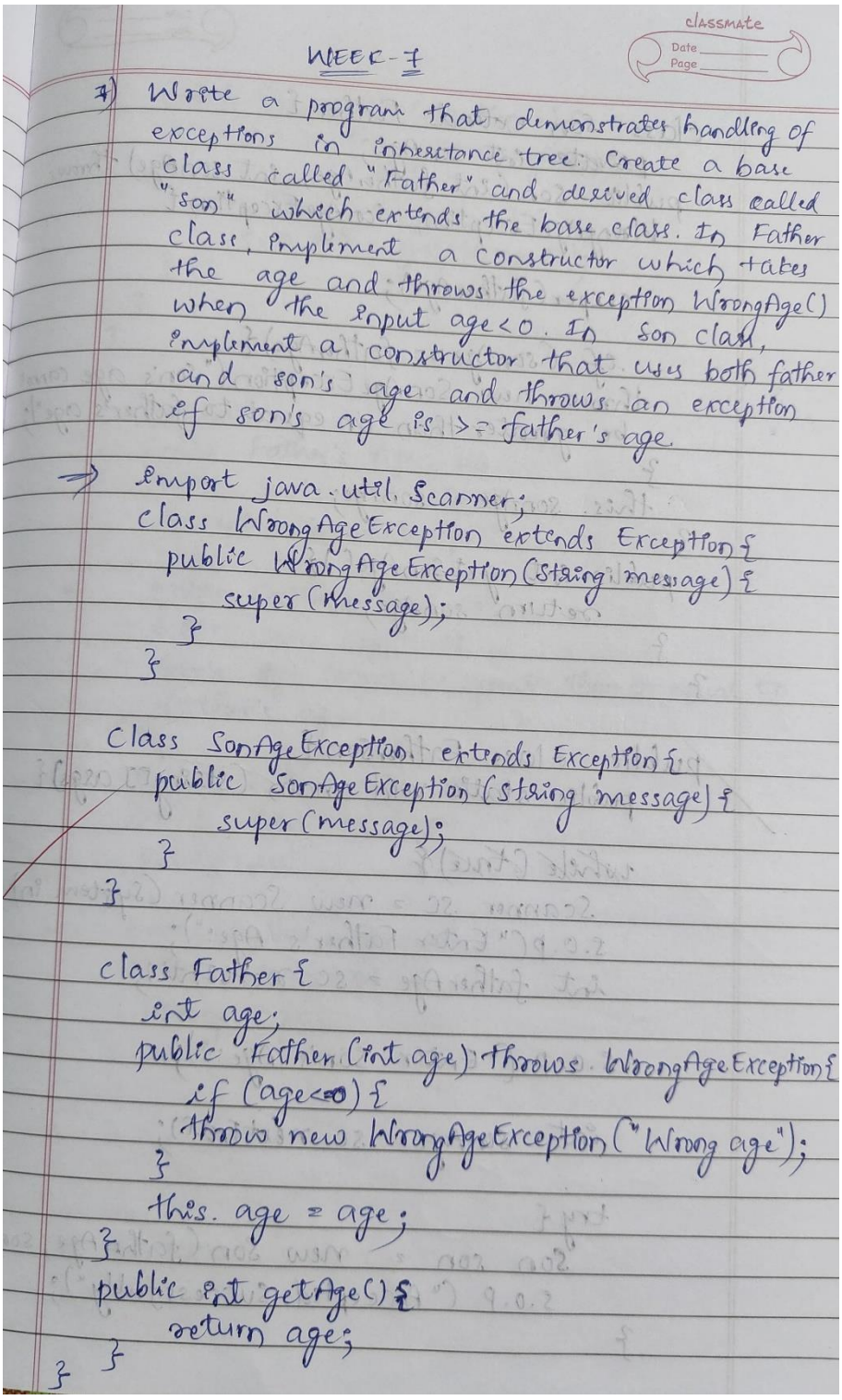
Output :

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac FatherSon.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java FatherSon
Enter Father's Age: 44
Enter Son's Age: 26
Accepted Succesfully
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac FatherSon.java
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
```

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac FatherSon.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java FatherSon
Enter Father's Age: 30
Enter Son's Age: 0
Wrong age
```

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac FatherSon.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab>  java FatherSon
Enter Father's Age: 0
Enter Son's Age: 15
Wrong age
```

Written Code & Output :



WEEK-7

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.

```
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 {
    private 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");
        }
        this.sonAge = sonAge;
    }
    public int getSonAge() {
        return sonAge;
    }
}

public class FatherSon {
    public static void main (String[] args) {
        while (true) {
            Scanner sc = new Scanner (System.in);
            S.O.P("Enter Father's Age:");
            int fatherAge = sc.nextInt();

            S.O.P ("Enter Son's Age:");

            int sonAge = sc.nextInt();

            try {
                Son son = new Son (fatherAge, sonAge);
                S.O.P ("Accepted Successfully");
            }
```

```
catch (WrongAgeException e){
        S.O.P (e.getmessage());
}
catch (SonAgeException e) {
        S.O.P (e.getMessage());
}
}
}
```

Output:

Enter Father's Age: 46
Enter Son's Age: 26
Accepted successfully

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: 0
Enter Son's Age: 15
Wrong age

28/11/24

Enter Father's Age: 30
Enter son's Age: 0
Wrong age

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 :

```
PS C:\Users\satis\OneDrive\Documents\ooj_lab> javac Demo.java
PS C:\Users\satis\OneDrive\Documents\ooj_lab> java Demo
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
```

Written Code & Output :

WEEK-8

Write a program which creates two threads one thread displaying "BMS College of Engineering" once every 10 seconds and another displaying "CSE" once every 2 seconds.

soln:

```
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 Main {
    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