Assignment 1:

1. **Write a program to list all even numbers less than or equal to the number n. Take the value of n as input from user.**

```
import java.util.Scanner;
public class EvenNumbers {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter a number (n): ");
      int n = scanner.nextInt();
      System.out.println("Even numbers less than or equal to " + n + ":");
      for (int i = 1; i <= n; i++) {
         if (i % 2 == 0) {
            System.out.println(i);
         }
      }
      scanner.close();
   }
}
```

```
C:\Users\91826\Desktop\java>javac EvenNumbers.java

C:\Users\91826\Desktop\java>java EvenNumbers
Enter a number (n): 4
Even numbers less than or equal to 4:
2
4
```

2. Define a class Rectangle with its length and breadth.

Provide appropriate constructor(s), which gives facilty of constructing rectangle object with default values of length pf breadth as 0 or passing value of length and breadth externally to constructor.

Provide appropriate accessor & mutator methods to Rectangle class.

Provide methods to calculate area & to display all information of Rectangle.

Design different class TestRectangle class in separate source file, which will contain main function. From this main function, create 5 Rectangle objects by taking all necessary information from the user.

ANS-

```
class Rectangle
{
   int len;
   int bre;

   Rectangle()
   {
      len = bre = 0;
   }

   Rectangle(int len, int bre)
   {
      this.len = len;
      this.bre = bre;
   }
   public nt Area()
   {
      return len * bre;
   }

   public void Display()
   {
      System.out.println("Length=" +len);
      System.out.println("Breadth=" +bre);
```

```
    }

}

class TestRectangle
{
    public static void main(String[] args)
    {
        Rectangle obj1 = new Rectangle(2, 3);
        System.out.println("Rectangle 1");
        System.out.println("Area Is");
        System.out.println(obj1.Area());
        System.out.println("Rectangle Info");
        obj1.Display();

        Rectangle obj2 = new Rectangle(4, 5);
        System.out.println("Rectangle 2");
        System.out.println("Area Is");
        System.out.println(obj2.Area());
        System.out.println("Rectangle Info");
        obj2.Display();

        Rectangle obj3 = new Rectangle(5, 6);
        System.out.println("Rectangle 3");
        System.out.println("Area Is");
        System.out.println(obj3.Area());
        System.out.println("Rectangle Info");
        obj3.Display();

        Rectangle obj4 = new Rectangle(6, 7);
        System.out.println("Rectangle 4");
        System.out.println("Area Is");
        System.out.println(obj4.Area());
        System.out.println("Rectangle Info");
```

```
        obj4.Display();

        Rectangle obj5 = new Rectangle(9, 7);
        System.out.println("Rectangle 5");
        System.out.println("Area Is");
        System.out.println(obj5.Area());
        System.out.println("Rectangle Info");
        obj5.Display();
    }
}
```

```
C:\Users\91826\Desktop\java>javac TestRectangle.java

C:\Users\91826\Desktop\java>java TestRectangle
Rectangle 1
Area Is
6
Rectangle Info
Length=2
Breadth=3
Rectangle 2
Area Is
20
Rectangle Info
Length=4
Breadth=5
Rectangle 3
Area Is
30
Rectangle Info
Length=5
Breadth=6
Rectangle 4
Area Is
42
Rectangle Info
Length=6
Breadth=7
Rectangle 5
Area Is
63
Rectangle Info
Length=9
Breadth=7
```

3. Create a class Book which describes its Book_title and Book_price. Use getter and setter methods to get & set the Books description.

Implement createBooks and showBooks methods to create n objects of Book in an array. Display the books along with its description as follows:-

Book Title	**Price**
Java Programming	**Rs.350.50**
Let Us C	**Rs.200.00**

Note: createBooks & showBooks should not be member functions of Book class.

ANS-

```
import java.util.Scanner;

class Book {
    private String bookTitle;
    private double bookPrice;

    public String getBookTitle() {
        return bookTitle;
    }

    public void setBookTitle(String bookTitle) {
        this.bookTitle = bookTitle;
    }

    public double getBookPrice() {
        return bookPrice;
    }

    public void setBookPrice(double bookPrice) {
        this.bookPrice = bookPrice;
    }
```

```
}

public class BookManager {
    public static Book[] createBooks(int n) {
        Book[] books = new Book[n];
        Scanner scanner = new Scanner(System.in);
        for (int i = 0; i < n; i++) {
            Book book = new Book();
            System.out.print("Enter book title: ");
            book.setBookTitle(scanner.nextLine());
            System.out.print("Enter book price: ");
            book.setBookPrice(scanner.nextDouble());
            scanner.nextLine(); // Consume newline
            books[i] = book;
        }
        return books;
    }

    public static void showBooks(Book[] books) {
        System.out.println("Book Title\t\tPrice");
        for (Book book : books) {
            System.out.printf("%-20s\tRs.%.2f%n", book.getBookTitle(), book.getBookPrice());
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of books: ");
        int n = scanner.nextInt();
        scanner.nextLine(); // Consume newline
```

```
        Book[] books = createBooks(n);
        showBooks(books);
        scanner.close();
    }
}
```

```
C:\Users\91826\Desktop\java>javac BookManager.java

C:\Users\91826\Desktop\java>java BookManager
Enter the number of books: 2
Enter book title: Java Programming
Enter book price: 350.50
Enter book title: Let Us C
Enter book price: 200.00
Book Title              Price
Java Programming        Rs.350.50
Let Us C                Rs.200.00
```

4. **Modify the program which is created in assignment 2 as follows**

The class has attributes length and width, each of which defaults to 1. It should have member functions that calculate the perimeter and area of the rectangle. It should have set and get functions for both length and width. The set functions should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0

```
import java.util.Scanner;
public class Rectangle {
    private double length;
    private double width;
    public Rectangle() {
        length = 1.0;
        width = 1.0;
    }
    public double getLength() {
        return length;
```

```
    }

    public void setLength(double length) {
        if (length > 0.0 && length < 20.0) {
            this.length = length;
        } else {
            System.out.println("Invalid length value. Length must be a floating-point number larger than 0.0 and less than 20.0");
        }
    }
    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        if (width > 0.0 && width < 20.0) {
            this.width = width;
        } else {
            System.out.println("Invalid width value. Width must be a floating-point number larger than 0.0 and less than 20.0");
        }
    }
    public double calculatePerimeter() {
        return 2 * (length + width);
    }
    public double calculateArea() {
        return length * width;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Rectangle rectangle = new Rectangle();
```

```
        System.out.print("Enter the length of the rectangle: ");
        double length = scanner.nextDouble();
        rectangle.setLength(length);
        System.out.print("Enter the width of the rectangle: ");
        double width = scanner.nextDouble();
        rectangle.setWidth(width);
        System.out.println("Perimeter of the rectangle: " + rectangle.calculatePerimeter());
        System.out.println("Area of the rectangle: " + rectangle.calculateArea());
        scanner.close();
    }
}
```

```
C:\Users\91826\Desktop\java>javac Rectangle.java

C:\Users\91826\Desktop\java>java Rectangle
Enter the length of the rectangle: 5
Enter the width of the rectangle: 3
Perimeter of the rectangle: 16.0
Area of the rectangle: 15.0
```

---

5. Create a class Date for manipulating dates. Provide a constructor that enables an object of this class to be initialized when it is declared (You can select any default values for the day, month & year, e.g. your birth date). Provide the necessary functionality to perform error checking on the initializer values for data members day, month, and year. Also, provide a member function to add an integer in a date to obtain a new date.

Design separate class Employee which will have following information.

Employee Number	**Number**
Employee Name	**Text**
Joining Date	**Date**

Provide appropriate constructor(s) & methods to this class. Provide main function which will create 5 objects of Employee class.

ANS-

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

```
class Date {
    private int day;
    private int month;
    private int year;
    public Date() {
        day = 1;
        month = 1;
        year = 2000;
    }
    public Date(int day, int month, int year) {
        if (isValidDate(day, month, year)) {
            this.day = day;
            this.month = month;
            this.year = year;
        } else {
            System.out.println("Invalid date! Using default values.");
            this.day = 1;
            this.month = 1;
            this.year = 2000;
        }
    }
    private boolean isValidDate(int day, int month, int year) {
        if (year < 0 || month < 1 || month > 12 || day < 1 || day > daysInMonth(month, year)) {
            return false;
        }
        return true;
    }
    private int daysInMonth(int month, int year) {
        int days = 0;
        switch (month) {
```

```
        case 1: case 3: case 5: case 7: case 8: case 10: case 12:
            days = 31;
            break;
        case 4: case 6: case 9: case 11:
            days = 30;
            break;
        case 2:
            if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
                days = 29;
            } else {
                days = 28;
            }
            break;
    }
    return days;
}
public void addDays(int days) {
    for (int i = 0; i < days; i++) {
        if (day < daysInMonth(month, year)) {
            day++;
        } else {
            day = 1;
            if (month < 12) {
                month++;
            } else {
                month = 1;
                year++;
            }
        }
    }
```

```
    }
    public void displayDate() {
        System.out.println("Date: " + day + "/" + month + "/" + year);
    }
}
class Employee {
    private int employeeNumber;
    private String employeeName;
    private Date joiningDate;
    public Employee(int employeeNumber, String employeeName, Date joiningDate) {
        this.employeeNumber = employeeNumber;
        this.employeeName = employeeName;
        this.joiningDate = joiningDate;
    }
    public void displayEmployee() {
        System.out.println("Employee Number: " + employeeNumber);
        System.out.println("Employee Name: " + employeeName);
        System.out.print("Joining Date: ");
        joiningDate.displayDate();
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        for (int i = 1; i <= 5; i++) {
            System.out.println("Enter details for Employee " + i + ":");
            System.out.print("Employee Number: ");
            int employeeNumber = scanner.nextInt();
            scanner.nextLine(); // Consume newline
            System.out.print("Employee Name: ");
```

```
            String employeeName = scanner.nextLine();
            System.out.println("Joining Date (dd mm yyyy): ");
            int day = scanner.nextInt();
            int month = scanner.nextInt();
            int year = scanner.nextInt();
            Date joiningDate = new Date(day, month, year);
            Employee employee = new Employee(employeeNumber, employeeName,
joiningDate);
            employee.displayEmployee();
            System.out.println();
        }
        scanner.close();
    }
}
```

```
C:\Users\91826\Desktop\java>java Main
Enter details for Employee 1:
Employee Number: 101
Employee Name: Deep
Joining Date (dd mm yyyy):
10 02 2023
Employee Number: 101
Employee Name: Deep
Joining Date: Date: 10/2/2023

Enter details for Employee 2:
Employee Number: 102
Employee Name: Ajinkya
Joining Date (dd mm yyyy):
12 04 2023
Employee Number: 102
Employee Name: Ajinkya
Joining Date: Date: 12/4/2023

Enter details for Employee 3:
Employee Number: 103
Employee Name: Akshar
Joining Date (dd mm yyyy):
14 05 2023
Employee Number: 103
Employee Name: Akshar
Joining Date: Date: 14/5/2023

Enter details for Employee 4:
Employee Number: 104
Employee Name: Anamika
Joining Date (dd mm yyyy):
23 04 2023
Employee Number: 104
Employee Name: Anamika
Joining Date: Date: 23/4/2023
```

```
Enter details for Employee 5:
Employee Number: 105
Employee Name: Aditya
Joining Date (dd mm yyyy):
15 07 2023
Employee Number: 105
Employee Name: Aditya
Joining Date: Date: 15/7/2023


C:\Users\91826\Desktop\java>
```

6. Write a program that takes a String through Command Line argument and display the length of the string. Also display the string into uppercase and check whether it is a palindrome or not. (Refer Java API Documentation)

ANS-

```
public class StringManipulation {
    public static void main(String[] args) {
        if (args.length < 1) {
            System.out.println("Usage: java StringManipulation <string>");
            return;
        }
        String inputString = args[0];
        System.out.println("Length of the string: " + inputString.length());
        String uppercaseString = inputString.toUpperCase();
        System.out.println("Uppercase string: " + uppercaseString);
        boolean isPalindrome = checkPalindrome(inputString);
        if (isPalindrome) {
            System.out.println("The string is a palindrome.");
        } else {
            System.out.println("The string is not a palindrome.");
        }
    }
    public static boolean checkPalindrome(String str) {
        int left = 0;
        int right = str.length() - 1;
        while (left < right) {
            if (str.charAt(left) != str.charAt(right)) {
                return false;
            }
            left++;
            right--;
```

```
        }
        return true;
    }
}
```

```
C:\Users\91826\Desktop\java>javac StringManipulation.java

C:\Users\91826\Desktop\java>java StringManipulation ROTATOR
Length of the string: 7
Uppercase string: ROTATOR
The string is a palindrome.
```

7. **Write a program that accepts two numbers from the Command Line and prints them out. Then use a *for loop* to print the next 13 numbers in the sequence where each number is the sum of the previous two. For example:**

```
input> java prob2 1 3
     output> 1 3 4 7 11 18 29 47 76 123 322 521 843 1364
```

ANS:

```
public class FibonacciSequence {
    public static void main(String[] args) {
        if (args.length < 2) {
            System.out.println("Please provide two numbers as command line arguments.");
            return;
        }
        int num1 = Integer.parseInt(args[0]);
        int num2 = Integer.parseInt(args[1]);
        System.out.println("First Number: " + num1);
        System.out.println("Second Number: " + num2);
        for (int i = 0; i < 13; i++) {
            int nextNum = num1 + num2;
            System.out.println("Next Number: " + nextNum);
            num1 = num2;
            num2 = nextNum;
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac FibonacciSequence.java

C:\Users\91826\Desktop\java>java FibonacciSequence 10 12
First Number: 10
Second Number: 12
Next Number: 22
Next Number: 34
Next Number: 56
Next Number: 90
Next Number: 146
Next Number: 236
Next Number: 382
Next Number: 618
Next Number: 1000
Next Number: 1618
Next Number: 2618
Next Number: 4236
Next Number: 6854
```

8. **Write a program that accepts two numbers in the range from 1 to 40 from the Command Line. It then compares these numbers against a single dimension *array* of five integer elements ranging in value from 1 to 40. The program displays the message *BINGO* if the two inputted values are found in the array element. For example:**

```
  input>java prob3 3 29
  output>Your first number was 3
  Your second number was 29
Its Bingo! // this message if 3 and 29 is found in the array
Not Found! // this message if 3 and 29 is not found in
  the         //array
The array was 7 25 5 19 30
```

ANS:

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

      if (args.length < 2) {
         System.out.println("Please provide two numbers as command line arguments.");
         return;
      }
      int num1 = Integer.parseInt(args[0]);
      int num2 = Integer.parseInt(args[1]);

      int[] array = {5, 10, 15, 20, 25};
      boolean bingo = false;
      for (int i = 0; i < array.length; i++) {
         if (num1 == array[i] || num2 == array[i]) {
            bingo = true;
            break;
         }
```

```
        }

        if (bingo) {
            System.out.println("BINGO");
        } else {
            System.out.println("Numbers not found in the array.");
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac BingoChecker.java

C:\Users\91826\Desktop\java>java BingoChecker 10 12
BINGO
```

9. Write a program that allows you to create an integer *array* of 18 elements with the following values: *int A[]={3,2,4,5,6,4,5,7,3,2,3,4,7,1,2,0,0,0}*. The program computes the sum of element 0 to 14 and stores it at element 15, computes the average and stores it at element 16 and identifies the smallest value from the array and stores it at element 17

ANS-

```
public class ArrayOperations {
    public static void main(String[] args) {
        int[] A = {3, 2, 4, 5, 6, 4, 5, 7, 3, 2, 3, 4, 7, 1, 2, 0, 0, 0};

        int sum = findSum(A, 0, 14);
        A[15] = sum;

        double average = findAverage(A, 0, 14);
        A[16] = (int) average;

        int min = findMin(A, 0, 14);
        A[17] = min;

        System.out.println("Array after computations:");
        printArray(A);
```

```
    }

    public static int findSum(int[] arr, int start, int end) {
        int sum = 0;
        for (int i = start; i <= end; i++) {
            sum += arr[i];
        }
        return sum;
    }

    public static double findAverage(int[] arr, int start, int end) {
        int sum = findSum(arr, start, end);
        return (double) sum / (end - start + 1);
    }

    public static int findMin(int[] arr, int start, int end) {
        int min = arr[start];
        for (int i = start + 1; i <= end; i++) {
            if (arr[i] < min) {
                min = arr[i];
            }
        }
        return min;
    }

    public static void printArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```

```
C:\Users\91826\Desktop\java>javac ArrayOperations.java

C:\Users\91826\Desktop\java>java ArrayOperations
Array after computations:
3 2 4 5 6 4 5 7 3 2 3 4 7 1 2 58 3 1
```

10. **Create a class Term. This class represents a term of a polynomial such as $2x^4$ where 2 is coefficient and 4 is exponent of the term.**

<u>Data members</u>:-
int coefficient
int exponent

Create another class Polynomial. The internal representation of a polynomial is an array of Terms. The size of this array should be fixed.
Provide a constructor for this class that will set all terms of a polynomial object as zero (where coefficient is 0 and exponent is 0).

Provide following functions:

setTerm(int, int) – Setting a term of a polynomial object. Each successive call of this function should set next term of the polynomial object.

It should do the following validations:-

Whether the exponent of the term being set is already used.
Whether the array size limit is exceeded.
Whether the exponent is negative.

In all the cases it should not set the term and display an appropriate message.

sort() – to arrange the terms in ascending order of exponents.

Provide a function to print a polynomial object

ANS:

```
class Term {
    int coefficient;
    int exponent;
```

```
    public Term() {
        this.coefficient = 0;
        this.exponent = 0;
    }

    public Term(int coefficient, int exponent) {
        this.coefficient = coefficient;
        this.exponent = exponent;
    }
}

class Polynomial {
    private Term[] terms;
    private int size;
    private int currentIndex;

    public Polynomial(int size) {
        this.size = size;
        this.terms = new Term[size];
        this.currentIndex = 0;
        for (int i = 0; i < size; i++) {
            this.terms[i] = new Term();
        }
    }

    public void setTerm(int coefficient, int exponent) {
        if (exponent < 0) {
            System.out.println("Exponent cannot be negative. Term not set.");
```

```
            return;
        }

        if (currentIndex >= size) {
            System.out.println("Array size limit exceeded. Term not set.");
            return;
        }
        for (int i = 0; i < currentIndex; i++) {
            if (terms[i].exponent == exponent) {
                System.out.println("Exponent already used. Term not set.");
                return;
            }
        }
        terms[currentIndex].coefficient = coefficient;
        terms[currentIndex].exponent = exponent;
        currentIndex++;
    }

    public void sort() {
        for (int i = 0; i < currentIndex - 1; i++) {
            for (int j = 0; j < currentIndex - i - 1; j++) {
                if (terms[j].exponent > terms[j + 1].exponent) {
                    Term temp = terms[j];
                    terms[j] = terms[j + 1];
                    terms[j + 1] = temp;
                }
            }
        }
```

```
    }

    public void printPolynomial() {
        for (int i = 0; i < currentIndex; i++) {
            System.out.print(terms[i].coefficient + "x^" + terms[i].exponent);
            if (i != currentIndex - 1) {
                System.out.print(" + ");
            }
        }
        System.out.println();
    }
}

public class Main {
    public static void main(String[] args) {
        Polynomial polynomial = new Polynomial(5);

        polynomial.setTerm(2, 4);
        polynomial.setTerm(3, 3);
        polynomial.setTerm(1, 2);
        polynomial.setTerm(4, 1);
        polynomial.setTerm(5, 0); // Should not be set

        polynomial.sort();
        polynomial.printPolynomial();
    }
}
```

```
C:\Users\91826\Desktop\java>javac Main.java

C:\Users\91826\Desktop\java>java Main
5x^0 + 4x^1 + 1x^2 + 3x^3 + 2x^4
```

Passing objects to methods

Estimated time: 1 Hour

11. Create a class Matrix. Internal representation of this class will be a two dimensional array of size 10 by 10. In addition, the class should have following data members and member functions:

Data members:-
int rows
int columns
Constructors -
The default constructor
Matrix() - This should set each of the array element to zero.

Overloaded constructor
Matrix(int, int) - This constructor should call the default constructor first. It should then assign the value of first parameter to variable rows, and the value of the second parameter to variable columns. You can assume that the values of both the parameters will be less than or equal to 10.

Member functions -
void setElement(int r, int c, int value) - This function should set the array element at row r and column c to the value val. This assignment should be done only if val is positive r and c are valid else the element should be set to zero.

Matrix transpose () – This function should transpose the matrix. Transpose of a matrix is another matrix where the elements in rows of the first matrix become elements of the corresponding columns in the new matrix.

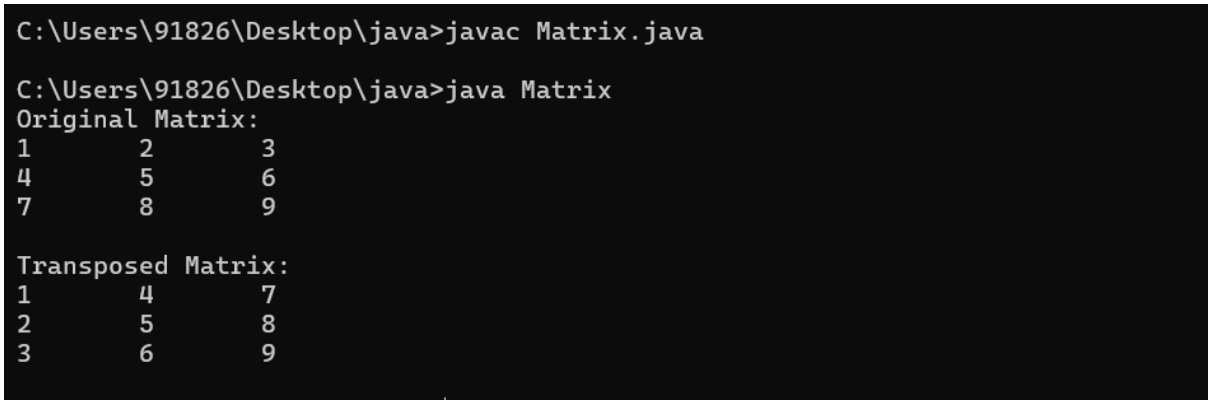
Provide a function to print a Matrix object.

ANS:

```
public class Matrix {
    private int[][] matrix;
    private int rows;
    private int columns;

    public Matrix() {
        this.rows = 10;
        this.columns = 10;
        this.matrix = new int[10][10];
        initializeMatrix();
    }
    public Matrix(int rows, int columns) {
        this();
        if (rows <= 10 && columns <= 10) {
            this.rows = rows;
            this.columns = columns;
        }
    }

    private void initializeMatrix() {
        for (int i = 0; i < this.rows; i++) {
            for (int j = 0; j < this.columns; j++) {
                this.matrix[i][j] = 0;
            }
        }
    }


    public void setElement(int r, int c, int value) {
        if (r >= 0 && r < rows && c >= 0 && c < columns && value > 0) {
```

```
            this.matrix[r][c] = value;
        } else {
            this.matrix[r][c] = 0;
            System.out.println("Invalid row or column index, or value is not positive. Element set to zero.");
        }
    }

    // Transpose the matrix
    public Matrix transpose() {
        Matrix transposedMatrix = new Matrix(this.columns, this.rows);
        for (int i = 0; i < this.rows; i++) {
            for (int j = 0; j < this.columns; j++) {
                transposedMatrix.matrix[j][i] = this.matrix[i][j];
            }
        }
        return transposedMatrix;
    }
    public void printMatrix() {
        for (int i = 0; i < this.rows; i++) {
            for (int j = 0; j < this.columns; j++) {
                System.out.print(this.matrix[i][j] + "\t");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {
        Matrix matrix = new Matrix(3, 3);
        matrix.setElement(0, 0, 1);
        matrix.setElement(0, 1, 2);
        matrix.setElement(0, 2, 3);
        matrix.setElement(1, 0, 4);
```

```
        matrix.setElement(1, 1, 5);
        matrix.setElement(1, 2, 6);
        matrix.setElement(2, 0, 7);
        matrix.setElement(2, 1, 8);
        matrix.setElement(2, 2, 9);

        System.out.println("Original Matrix:");
        matrix.printMatrix();

        System.out.println("\nTransposed Matrix:");
        Matrix transposedMatrix = matrix.transpose();
        transposedMatrix.printMatrix();
    }
}
```



12. Create a class called complex for performing arithmetic operations with complex numbers. Use floating point variables to represent the private data of the class. Provide a default constructor that initializes the object with some default values. Provide public member functions for each of the following

- **Addition of two complex numbers: It returns the result obtained by adding the respective real parts and the imaginary parts of the two complex numbers.**
- **Subtraction of two complex numbers: It returns the result obtained by subtracting the respective real parts and the imaginary parts of the two complex numbers.**
- **display() – It displays the complex number in a+bi format.**

The output should be displayed as follows:-

Sum of a1+b1 i & a2+b2 i is : a3+b3 i

ANS:

```
public class Complex {
   private double real;
   private double imaginary;
   public Complex() {
      this.real = 0.0;
      this.imaginary = 0.0;
   }
   public Complex(double real, double imaginary) {
      this.real = real;
      this.imaginary = imaginary;
   }
   public Complex add(Complex other) {
      double realPart = this.real + other.real;
      double imaginaryPart = this.imaginary + other.imaginary;
      return new Complex(realPart, imaginaryPart);
   }

   public Complex subtract(Complex other) {
      double realPart = this.real - other.real;
```

```
        double imaginaryPart = this.imaginary - other.imaginary;
        return new Complex(realPart, imaginaryPart);
    }
    public void display() {
        System.out.println("Sum of " + this.real + "+" + this.imaginary + "i & " + this.real + "+" +
this.imaginary + "i is: " + (this.real + this.real) + "+" + (this.imaginary + this.imaginary) + "i");
    }

    public static void main(String[] args) {
        Complex complex1 = new Complex(2.5, 3.5);
        Complex complex2 = new Complex(1.5, 2.5);

        Complex sum = complex1.add(complex2);
        System.out.println("Sum:");
        sum.display();

        Complex difference = complex1.subtract(complex2);
        System.out.println("Difference:");
        difference.display();
    }
}
```

```
C:\Users\91826\Desktop\java>javac Complex.java

C:\Users\91826\Desktop\java>java Complex
Sum:
Sum of 4.0+6.0i & 4.0+6.0i is: 8.0+12.0i
Difference:
Sum of 1.0+1.0i & 1.0+1.0i is: 2.0+2.0i
```

Assignment NO.II

1. **Create an abstract class Instrument which is having the abstract function play. Create three more sub classes from Instrument which is Piano, Flute, Guitar. Override the play method inside all three classes printing a message**

   "Piano is playing tan tan tan tan " for Piano class
   "Flute is playing toot toot toot toot" for Flute class
   "Guitar is playing tin tin tin " for Guitar class

   You must not allow the user to declare an object of Instrument class.

   Create an array of 10 Instruments.

   Assign different type of instrument to Instrument reference.

   Check for the polymorphic behavior of play method.

   Use the instanceof operator to print that which object stored at which index of instrument array.

ANS

```
abstract class Instrument
{
   abstract void play();
}

class Piano extends Instrument
{
   void play()
         {
      System.out.println("Piano is playing tan tan tan tan");
          }
}

class Flute extends Instrument
{
   void play()
         {
      System.out.println("Flute is playing toot toot toot toot");
          }
}

class Guitar extends Instrument
         {
   void play()
         {
      System.out.println("Guitar is playing tin tin tin");
         }
```

```
}

public class Main {
   public static void main(String[] args)
         {
      Instrument[] instruments = new Instrument[10];
      for (int i = 0; i < 10; i++)
         {
         if (i % 3 == 0)
               {
            instruments[i] = new Piano();
         }
            else if (i % 3 == 1)
               {
            instruments[i] = new Flute();
         }
            else
               {
            instruments[i] = new Guitar();
            }
      }

      for (int i = 0; i < 10; i++) {
         instruments[i].play();
         if (instruments[i] instanceof Piano)
               {
            System.out.println("Object at index " + i + " is a Piano.");
               System.out.println("----------------------------------------------------");
            }
            else if (instruments[i] instanceof Flute)
               {
            System.out.println("Object at index " + i + " is a Flute.");
               System.out.println("----------------------------------------------------");
            }
            else if (instruments[i] instanceof Guitar)
               {
            System.out.println("Object at index " + i + " is a Guitar.");
               System.out.println("----------------------------------------------------");
         }
      }
   }
}
```

```
C:\Users\91826\Downloads>javac Main.java

C:\Users\91826\Downloads>java Main
Piano is playing tan tan tan tan
Object at index 0 is a Piano.
-------------------------------------------------------
Flute is playing toot toot toot toot
Object at index 1 is a Flute.
-------------------------------------------------------
Guitar is playing tin tin tin
Object at index 2 is a Guitar.
-------------------------------------------------------
Piano is playing tan tan tan tan
Object at index 3 is a Piano.
-------------------------------------------------------
Flute is playing toot toot toot toot
Object at index 4 is a Flute.
-------------------------------------------------------
Guitar is playing tin tin tin
Object at index 5 is a Guitar.
-------------------------------------------------------
Piano is playing tan tan tan tan
Object at index 6 is a Piano.
-------------------------------------------------------
Flute is playing toot toot toot toot
Object at index 7 is a Flute.
-------------------------------------------------------
Guitar is playing tin tin tin
Object at index 8 is a Guitar.
-------------------------------------------------------
Piano is playing tan tan tan tan
Object at index 9 is a Piano.
-------------------------------------------------------
```

2. **Create an abstract class Compartment to represent a rail coach. Provide a abstract function notice in this class. Derive FirstClass, Ladies, General, Luggage classes from the compartment class. Override the notice function in each of them to print notice suitable to the type of the compartment.**

   Create a class TestCompartment . Write main function to do the following:

   Declare an array of Compartment pointers of size 10.

   Create a compartment of a type as decided by a randomly generated integer in the range 1 to 4.

   Check the polymorphic behavior of the notice method.

   ANS-

```
abstract class Compartment {
   abstract void notice();
}

class FirstClass extends Compartment {
   void notice() {
      System.out.println("Welcome to First Class!");
   }
}

class Ladies extends Compartment {
   void notice() {
      System.out.println("Welcome to Ladies Coach!");
   }
}

class General extends Compartment {
   void notice() {
      System.out.println("Welcome to General Coach!");
   }
}

class Luggage extends Compartment {
   void notice() {
      System.out.println("Welcome to Luggage Coach!");
   }
}

public class TestCompartment {
   public static void main(String[] args) {
      Compartment[] compartments = new Compartment[10];

      for (int i = 0; i < 10; i++) {
```

```
            int random = (int) (Math.random() * 4) + 1;
            switch (random) {
                case 1:
                    compartments[i] = new FirstClass();
                    break;
                case 2:
                    compartments[i] = new Ladies();
                    break;
                case 3:
                    compartments[i] = new General();
                    break;
                case 4:
                    compartments[i] = new Luggage();
                    break;
            }
        }

        for (int i = 0; i < 10; i++) {
            compartments[i].notice();
        }
    }
}
```

```
C:\Users\91826\Downloads>javac TestCompartment.java

C:\Users\91826\Downloads>java TestCompartment
Welcome to Ladies Coach!
Welcome to General Coach!
Welcome to Luggage Coach!
Welcome to Luggage Coach!
Welcome to General Coach!
Welcome to First Class!
Welcome to General Coach!
Welcome to Ladies Coach!
Welcome to General Coach!
Welcome to Luggage Coach!
```

3. **Create a class Medicine to represent a drug manufactured by a pharmaceutical company. Provide a function displayLabel() in this class to print Name and address of the company.**

   Derive Tablet, Syrup and Ointment classes from the Medicine class. Override the displayLabel() function in each of these classes to print additional information suitable to the type of medicine. For example, in case of tablets, it could be "store in a cool dry place", in case of ointments it could be "for external use only" etc.

   Create a class TestMedicine . Write main function to do the following:
   Declare an array of Medicine references of size 10
   Create a medicine object of the type as decided by a randomly generated integer in the range 1 to 3.

   Refer Java API Documentation to find out random generation feature.

   Check the polymorphic behavior of the displayLabel() method.

   ANS-

```
import java.util.Random;

class Medicine {
    private String companyName;
    private String companyAddress;

    Medicine(String companyName, String companyAddress) {
        this.companyName = companyName;
        this.companyAddress = companyAddress;
    }

    void displayLabel() {
        System.out.println("Company Name: " + companyName);
        System.out.println("Company Address: " + companyAddress);
    }
}

class Tablet extends Medicine {
    Tablet(String companyName, String companyAddress) {
        super(companyName, companyAddress);
    }

    void displayLabel() {
        super.displayLabel();
        System.out.println("Store in a cool dry place.");
    }
}

class Syrup extends Medicine {
    Syrup(String companyName, String companyAddress) {
```

```
        super(companyName, companyAddress);
    }

    void displayLabel() {
        super.displayLabel();
        System.out.println("Shake well before use.");
    }
}

class Ointment extends Medicine {
    Ointment(String companyName, String companyAddress) {
        super(companyName, companyAddress);
    }

    void displayLabel() {
        super.displayLabel();
        System.out.println("For external use only.");
    }
}

public class TestMedicine {
    public static void main(String[] args) {
        Medicine[] medicines = new Medicine[10];
        Random rand = new Random();

        for (int i = 0; i < 10; i++) {
            int random = rand.nextInt(3) + 1;
            switch (random) {
                case 1:
                    medicines[i] = new Tablet("ABC Pharmaceuticals", "123 Main St");
                    break;
                case 2:
                    medicines[i] = new Syrup("XYZ Pharmaceuticals", "456 Oak St");
                    break;
                case 3:
                    medicines[i] = new Ointment("DEF Pharmaceuticals", "789 Elm St");
                    break;
            }
        }

        for (int i = 0; i < 10; i++) {
            System.out.println("Medicine " + (i + 1) + " Label:");
            medicines[i].displayLabel();
            System.out.println();
        }
    }
}
```

```
C:\Users\91826\Downloads>javac TestMedicine.java

C:\Users\91826\Downloads>java TestMedicine
Medicine 1 Label:
Company Name: XYZ Pharmaceuticals
Company Address: 456 Oak St
Shake well before use.

Medicine 2 Label:
Company Name: XYZ Pharmaceuticals
Company Address: 456 Oak St
Shake well before use.

Medicine 3 Label:
Company Name: ABC Pharmaceuticals
Company Address: 123 Main St
Store in a cool dry place.

Medicine 4 Label:
Company Name: DEF Pharmaceuticals
Company Address: 789 Elm St
For external use only.

Medicine 5 Label:
Company Name: XYZ Pharmaceuticals
Company Address: 456 Oak St
Shake well before use.

Medicine 6 Label:
Company Name: DEF Pharmaceuticals
Company Address: 789 Elm St
For external use only.

Medicine 7 Label:
Company Name: DEF Pharmaceuticals
Company Address: 789 Elm St
For external use only.

Medicine 8 Label:
Company Name: XYZ Pharmaceuticals
Company Address: 456 Oak St
Shake well before use.

Medicine 9 Label:
Company Name: DEF Pharmaceuticals
Company Address: 789 Elm St
For external use only.

Medicine 10 Label:
Company Name: XYZ Pharmaceuticals
Company Address: 456 Oak St
Shake well before use.
```

4.Write a program that accepts two numbers and a operator like (+,-,*, /) as command line arguments and perform the appropriate operation indicated by operator.

If the user enters any other character the appropriate message will be displayed. The output of the program should be displayed to the user.

ANS_

```
public class Calculator {
    public static void main(String[] args) {
        if (args.length != 3) {
            System.out.println("Usage: java Calculator <number1> <operator> <number2>");
            return;
        }

        double num1, num2;
        try {
            num1 = Double.parseDouble(args[0]);
            num2 = Double.parseDouble(args[2]);
        } catch (NumberFormatException e) {
            System.out.println("Invalid number format. Please enter valid numbers.");
            return;
        }

        char operator = args[1].charAt(0);
        double result = 0;
        switch (operator) {
            case '+':
                result = num1 + num2;
                break;
            case '-':
                result = num1 - num2;
                break;
            case '*':
                result = num1 * num2;
                break;
            case '/':
                if (num2 != 0)
                    result = num1 / num2;
                else {
                    System.out.println("Error: Division by zero");
                    return;
                }
                break;
            default:
                System.out.println("Invalid operator. Please use +, -, *, or /");
                return;
        }

        System.out.println("Result: " + result);
```

```
    }
}
```

```
C:\Users\91826\Downloads>javac Calculator.java

C:\Users\91826\Downloads>java Calculator 20 + 30
Result: 50.0

C:\Users\91826\Downloads>
```

5. **Create a class Car which contains members speed, noOfGear. The class has a method drive() which is responsible to provide starting speed and noOfGears to a Car. Implement display() method which will display all attributes of Car class.**

The class SportCar is derived from the class Car which adds new features AirBallonType. When this method is invoked, initial speed and gear status must be displayed on console. Override the display method which display all attribute of the SportCar. Make use of super class display() method.

ANS-

```
class Car {
    protected int speed;
    protected int noOfGears;

    public void drive(int startingSpeed, int noOfGears) {
        this.speed = startingSpeed;
        this.noOfGears = noOfGears;
    }

    public void display() {
        System.out.println("Car Speed: " + speed);
        System.out.println("Number of Gears: " + noOfGears);
    }
}

class SportCar extends Car {
    private String airBalloonType;

    public void setAirBalloonType(String airBalloonType) {
        this.airBalloonType = airBalloonType;
    }

    public void display() {
```

```
        super.display();
        System.out.println("Air Balloon Type: " + airBalloonType);
    }
}

public class Main1{
    public static void main(String[] args) {
        Car car = new Car();
        car.drive(60, 6);
        System.out.println("Car Details:");
        car.display();

        System.out.println();

        SportCar sportCar = new SportCar();
        sportCar.drive(100, 8);
        sportCar.setAirBalloonType("Nylon");
        System.out.println("Sport Car Details:");
        sportCar.display();
    }
}
```

```
C:\Users\91826\Downloads>javac Main1.java

C:\Users\91826\Downloads>java Main1
Car Details:
Car Speed: 60
Number of Gears: 6

Sport Car Details:
Car Speed: 100
Number of Gears: 8
Air Balloon Type: Nylon
```

Assignment NO.III

Day #3: Packages and Exception Handling

1. Create a package techm.itp.hyd<your batch id>.cs/ncs<your emp id>.<your first name>.
For e.g., techm.itp.hyd10001.cs35123.Anu
Now create a Greeter class in this package having the following features:

Attributes:
name string //indicates name of the person to be greeted
Member functions:
Greeter(aName) //constructor to initialize the name of the //person to be greeted by this greeter.
sayHello() //returns a hello message with the name of the //person initialized earlier.
sayGoodBye() //bids goodbye to the person named earlier.

Create another class in the same package called Advisor that has the following features:
Attributes:
message string[5] //contains five advice messages
Member functions:
Advisor() //default constructor to initialize an array of //strings with atleast five advice messages
getAdvice() //randomly selects an advice from the available //list of messages and returns it to the caller of //this method
Outside the package, from your working directory, create a class GreeterTest that constructs Greeter objects for all command-line arguments and prints out the results of calling sayHello(). The program should then display an advice and finally bid goodbye to each of the persons/entities in reverse order of the names entered at the command line.

For e.g.,
java GreeterTest Mars Venus
then the program should print
Hello, Mars!
Hello, Venus!
Advice: Never say No
Goodbye Venus!
Goodbye Mars!

ANS-

// File: Greeter.java

```
package techm.itp.hyd10001.cs35123.Anu;

public class Greeter {

   String name;

   public Greeter(String name) {

      this.name = name;
```

```
    }

    public void sayHello() {
        System.out.println("Hello, " + name + "!");
    }
    public void sayGoodbye() {
        System.out.println("Goodbye " + name + "!");
    }
}
```

// File: Advisor.java

```
package techm.itp.hyd10001.cs35123.Anu;
import java.util.Random;
public class Advisor {
    String[] messages;
    public Advisor() {
        messages = new String[]{"Never say No", "Take risks", "Learn from failures", "Stay curious", "Be kind"};
    }
    public String getAdvice() {
        Random random = new Random();
        int index = random.nextInt(messages.length);
        return messages[index];
    }
}
```

// File: GreeterTest.java

```
import techm.itp.hyd10001.cs35123.Anu.Greeter;
import techm.itp.hyd10001.cs35123.Anu.Advisor;
public class GreeterTest {
    public static void main(String[] args) {
```

```
        Greeter[] greeters = new Greeter[args.length];


        for (int i = 0; i < args.length; i++) {
            greeters[i] = new Greeter(args[i]);
            greeters[i].sayHello();
        }
        Advisor advisor = new Advisor();
        System.out.println("Advice: " + advisor.getAdvice());
        for (int i = args.length - 1; i >= 0; i--) {
            greeters[i].sayGoodbye();
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac GreeterTest.java

C:\Users\91826\Desktop\java>java GreeterTest Mars Venus
Hello, Mars!
Hello, Venus!
Advice: Never say No
Goodbye Venus!
Goodbye Mars!
```

2. Create a package esg.itp.shape containing the following classes and interface

An interface Polygon containing the members as given below:

area float

perimeter float

void calcArea( ); abstract method to calculate area of a particular polygon given its dimensions

void calcPeri( ); abstract method to calculate perimeter of a particular polygon given its dimensions

void display( ); method to display the area and perimeter

of the given polygon

Create a class Square that implements Polygon and has the following member:

side float

Square(float s); constructor to initialize side of square

Create another class Rectangle that implements Polygon and has the following member:

length float

breadth float

Rectangle(int len, int bre); constructor to initialize length and breadth of a rectangle

Outside the package, create a class that imports the above package an instantiates an object of the Square class and an object of the Rectangle class.

Call the above methods on each of the classes to calculate the area and perimeter given the side and the length/breadth of the Square class and the Rectangle class respectively.

ANS-

// File: Square.java

```
package esg.itp.shape;
public class Square implements Polygon {
    private float side;
    private float area; // Instance variable
    private float perimeter; // Instance variable
    public Square(float s) {
        side = s;
    }
    public void calcArea() {
```

```
        area = side * side;
    }
    public void calcPeri() {
        perimeter = 4 * side;
    }
    public void display() {
        System.out.println("Square - Area: " + area + ", Perimeter: " + perimeter);
    }
}
```

// File: Square.java

```
package esg.itp.shape;
public class Square implements Polygon {
    private float side;
    private float area; // Instance variable
    private float perimeter; // Instance variable
    public Square(float s) {
        side = s;
    }
    public void calcArea() {
        area = side * side;
    }
    public void calcPeri() {
        perimeter = 4 * side;
    }
    public void display() {
        System.out.println("Square - Area: " + area + ", Perimeter: " + perimeter);
    }
}
```

```
// File: Rectangle.java
package esg.itp.shape;
public class Rectangle implements Polygon {
    private float length;
    private float breadth;
    private float area; // Instance variable
    private float perimeter; // Instance variable
    public Rectangle(float len, float bre) {
        length = len;
        breadth = bre;
    }
    public void calcArea() {
        area = length * breadth;
    }
    public void calcPeri() {
        perimeter = 2 * (length + breadth);
    }
    public void display() {
        System.out.println("Rectangle - Area: " + area + ", Perimeter: " + perimeter);
    }
}

// File: ShapeTester.java
package esg.itp.shape;
public class ShapeTester {
    public static void main(String[] args) {
        Square square = new Square(5);
        square.calcArea();
        square.calcPeri();
        square.display();
```

```
        Rectangle rectangle = new Rectangle(4, 6);
        rectangle.calcArea();
        rectangle.calcPeri();
        rectangle.display();
    }
}
```

```
C:\Users\91826\Desktop\java>javac ShapeTester.java

C:\Users\91826\Desktop\java>java esg.itp.shape.ShapeTester
Square - Area: 25.0, Perimeter: 20.0
Rectangle - Area: 24.0, Perimeter: 20.0

C:\Users\91826\Desktop\java>
```

3. **Create a class called CalcAverage that has the following method:**

   public double avgFirstN(int N)

This method receives an integer as a parameter and calculates the average of first N natural numbers. If N is not a natural number, throw an exception IllegalArgumentException with an appropriate message.

ANS:

```
public class CalcAverage {

    public double avgFirstN(int N) {
        if (N <= 0) {
            throw new IllegalArgumentException("N must be a positive integer");
```

```
        }

        int sum = 0;
        for (int i = 1; i <= N; i++) {
            sum += i;
        }

        return (double) sum / N;
    }

    public static void main(String[] args) {
        CalcAverage calculator = new CalcAverage();

        try {
            double avg = calculator.avgFirstN(5);
            System.out.println("Average of first 5 natural numbers: " + avg);
        } catch (IllegalArgumentException e) {
            System.out.println("Exception caught: " + e.getMessage());
        }
    }
}
```

```
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\New folder>javac CalcAverage.java

C:\New folder>java CalcAverage
Average of first 5 natural numbers: 3.0
```

4. **Create a class Number having the following features:**

Attributes

int	**first number**	
int	**second number**	
result	**double**	**stores the result of arithmetic operations performed on a and b**

Member functions

Number(x, y)	**constructor to initialize the values of a and b**
add()	**stores the sum of a and b in result**
sub()	**stores difference of a and b in result**
mul()	**stores product in result**
div()	**stores a divided by b in result**

Test to see if b is 0 and throw an appropriate exception since division by zero is undefined.

Display a menu to the user to perform the above four arithmetic operations.

Ans:

```
import java.util.Scanner;

public class Number {
   private int firstNumber;
   private int secondNumber;
   private double result;

   // Constructor to initialize the values of a and b
   public Number(int x, int y) {
      this.firstNumber = x;
      this.secondNumber = y;
   }
```

```
// Method to store the sum of a and b in result
public void add() {
    result = firstNumber + secondNumber;
}

// Method to store the difference of a and b in result
public void sub() {
    result = firstNumber - secondNumber;
}

// Method to store the product of a and b in result
public void mul() {
    result = firstNumber * secondNumber;
}

// Method to store a divided by b in result
public void div() {
    if (secondNumber == 0) {
        throw new ArithmeticException("Division by zero is undefined");
    }
    result = (double) firstNumber / secondNumber;
}

// Method to display a menu and perform arithmetic operations
public void displayMenu() {
    Scanner scanner = new Scanner(System.in);
    int choice;

    do {
        System.out.println("Menu:");
        System.out.println("1. Add");
```

```
System.out.println("2. Subtract");
System.out.println("3. Multiply");
System.out.println("4. Divide");
System.out.println("5. Exit");
System.out.print("Enter your choice: ");
choice = scanner.nextInt();

switch (choice) {
   case 1:
      add();
      System.out.println("Result: " + result);
      break;
   case 2:
      sub();
      System.out.println("Result: " + result);
      break;
   case 3:
      mul();
      System.out.println("Result: " + result);
      break;
   case 4:
      try {
         div();
         System.out.println("Result: " + result);
      } catch (ArithmeticException e) {
         System.out.println("Exception caught: " + e.getMessage());
      }
      break;
   case 5:
      System.out.println("Exiting...");
      break;
```

```
            default:
                System.out.println("Invalid choice! Please enter a number between 1 and 5.");
        }
    } while (choice != 5);
    scanner.close();
  }

  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the first number: ");
    int x = scanner.nextInt();
    System.out.print("Enter the second number: ");
    int y = scanner.nextInt();

    Number number = new Number(x, y);
    number.displayMenu();
  }
}
```

```
C:\New folder>javac Number.java

C:\New folder>java Number
Enter the first number: 12
Enter the second number: 13
Menu:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit
Enter your choice: 1
Result: 25.0
Menu:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit
Enter your choice: 4
Result: 0.9230769230769231
Menu:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit
Enter your choice: 5
Exiting...
```

5. **Create a class BankAccount having the members as given below:**

   accNo integer

   custName string

   accType string (indicates 'Savings' or 'Current')

   balance float

   Include the following methods in the BankAccount class:

   void deposit(float amt);

   void withdraw(float amt);

   float getBalance();

deposit(float amt) method allows you to credit an amount into the current balance. If amount is negative, throw an exception NegativeAmount to block the operation from being performed.

withdraw(float amt) method allows you to debit an amount from the current balance. Please ensure a minimum balance of Rs. 1000/- in the account for savings account and Rs. 5000/- for current account, else throw an exception InsufficientFunds and block the withdrawal operation. Also throw an exception NegativeAmount to block the operation from being performed if the amt parameter passed to this function is negative.

getBalance() method returns the current balance. If the current balance is below the minimum required balance, then throw an exception LowBalanceException accordingly.

Have constructor to which you will pass, accno, cust_name, acctype and initial balance.

And check whether the balance is less than 1000 or not in case of savings account and less than 5000 in case of a current account. If so, then raise a LowBalanceException.

In either case if the balance is negative then raise the NegativeAmount exception accordingly.

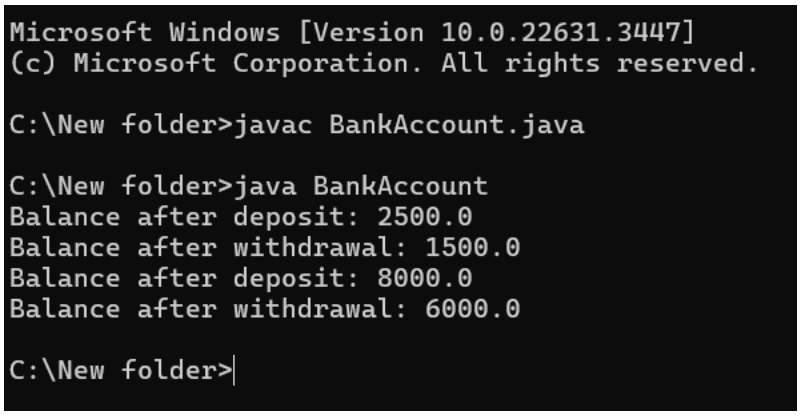
Ans:

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

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

class LowBalanceException extends Exception {
   public LowBalanceException(String message) {
```

```
        super(message);
    }
}

public class BankAccount {
    private int accNo;
    private String custName;
    private String accType;
    private float balance;

    public BankAccount(int accNo, String custName, String accType, float balance) throws LowBalanceException, NegativeAmountException {
        this.accNo = accNo;
        this.custName = custName;
        this.accType = accType;
        if (balance < 0) {
            throw new NegativeAmountException("Initial balance cannot be negative.");
        }
        if (accType.equalsIgnoreCase("Savings") && balance < 1000) {
            throw new LowBalanceException("Minimum balance for Savings account should be Rs. 1000/-");
        } else if (accType.equalsIgnoreCase("Current") && balance < 5000) {
            throw new LowBalanceException("Minimum balance for Current account should be Rs. 5000/-");
        }
        this.balance = balance;
    }

    public void deposit(float amt) throws NegativeAmountException {
        if (amt < 0) {
            throw new NegativeAmountException("Amount to deposit cannot be negative.");
```

```
        }
        balance += amt;
    }

    public void withdraw(float amt) throws NegativeAmountException, InsufficientFundsException, LowBalanceException {
        if (amt < 0) {
            throw new NegativeAmountException("Amount to withdraw cannot be negative.");
        }

        if (accType.equalsIgnoreCase("Savings") && balance - amt < 1000) {
            throw new InsufficientFundsException("Insufficient funds. Minimum balance for Savings account should be Rs. 1000/-");
        } else if (accType.equalsIgnoreCase("Current") && balance - amt < 5000) {
            throw new InsufficientFundsException("Insufficient funds. Minimum balance for Current account should be Rs. 5000/-");
        }

        balance -= amt;
    }

    public float getBalance() throws LowBalanceException {
        if (accType.equalsIgnoreCase("Savings") && balance < 1000) {
            throw new LowBalanceException("Low balance. Minimum balance for Savings account should be Rs. 1000/-");
        } else if (accType.equalsIgnoreCase("Current") && balance < 5000) {
            throw new LowBalanceException("Low balance. Minimum balance for Current account should be Rs. 5000/-");
        }
        return balance;
    }
```

```
        public static void main(String[] args) {
            try {
                BankAccount savingsAccount = new BankAccount(123456, "John Doe", "Savings", 2000);
                savingsAccount.deposit(500);
                System.out.println("Balance after deposit: " + savingsAccount.getBalance());
                savingsAccount.withdraw(1000);
                System.out.println("Balance after withdrawal: " + savingsAccount.getBalance());

                BankAccount currentAccount = new BankAccount(789012, "Jane Smith", "Current", 7000);
                currentAccount.deposit(1000);
                System.out.println("Balance after deposit: " + currentAccount.getBalance());
                currentAccount.withdraw(2000);
                System.out.println("Balance after withdrawal: " + currentAccount.getBalance());
            } catch (LowBalanceException | NegativeAmountException | InsufficientFundsException e) {
                System.out.println("Exception caught: " + e.getMessage());
            }
        }
    }
```



6. Create a class with following specifications.

Class Emp

empId	**int**
empName	**string**
designation	**string**
basic	**double**
hra	**double readOnly**

Methods

printDET()
calculateHRA()
printDET() methods will show details of the EMP.

calculateHRA() method will calculate HRA based on basic.

There will 3 designations supported by the application.

If designation is "Manager" - HRA will be 10% of BASIC
if designation is "Officer" - HRA will be 12% of BASIC
if category is "CLERK" - HRA will be 5% of BASIC

Have constructor to which you will pass, empId, designation, basic and price.

And checks whether the BASIC is less than 500 or not. If it is less than 500 raise a custom Exception as given below

Create LowSalException class with proper user message to handle BASIC less than 500.

Ans:

```
class LowSalException extends RuntimeException {
   public LowSalException(String message) {
      super(message);
   }
}
class Emp {
   private int empId;
   private String empName;
   private String designation;
   private double basic;
   private double hra;

   public Emp(int empId, String empName, String designation, double basic) {
      this.empId = empId;
      this.empName = empName;
      this.designation = designation;
      if (basic < 500) {
         throw new LowSalException("Basic salary cannot be less than 500");
```

```
        }
        this.basic = basic;
        calculateHRA();
    }

    private void calculateHRA() {
        switch (designation.toLowerCase()) {
            case "manager":
                hra = 0.1 * basic;
                break;
            case "officer":
                hra = 0.12 * basic;
                break;
            case "clerk":
                hra = 0.05 * basic;
                break;
            default:
                throw new IllegalArgumentException("Invalid designation");
        }
    }

    public void printDET() {
        System.out.println("Employee ID: " + empId);
        System.out.println("Employee Name: " + empName);
        System.out.println("Designation: " + designation);
        System.out.println("Basic Salary: " + basic);
        System.out.println("HRA: " + hra);
    }
}

public class Main {
    public static void main(String[] args) {
        try {
            Emp emp1 = new Emp(101, "John", "Manager", 600);
            emp1.printDET();
            System.out.println();

            Emp emp2 = new Emp(102, "Alice", "Officer", 700);
            emp2.printDET();
            System.out.println();

            Emp emp3 = new Emp(103, "Bob", "Clerk", 800);
            emp3.printDET();
            System.out.println();

            // This will throw LowSalException
            Emp emp4 = new Emp(104, "Doe", "Manager", 400);
        } catch (LowSalException e) {
            System.out.println(e.getMessage());
        }
```

```
    }
}
```

```
C:\Users\91826\Desktop\java>javac Main.java

C:\Users\91826\Desktop\java>java Main
Employee ID: 101
Employee Name: John
Designation: Manager
Basic Salary: 600.0
HRA: 60.0

Employee ID: 102
Employee Name: Alice
Designation: Officer
Basic Salary: 700.0
HRA: 84.0

Employee ID: 103
Employee Name: Bob
Designation: Clerk
Basic Salary: 800.0
HRA: 40.0

Basic salary cannot be less than 500
```

7. **Create a class USERTRAIL with following specifications.**

val1, val2 type int
Methods
boolean show () will check if val1 and val2 are greater or less than Zero

Have constructor which will val1, val2 and check whether if it is less than 0 then raise a custom Exception (name: Illegal value exception.)

Ans:

```
class IllegalValueException extends RuntimeException {
    public IllegalValueException(String message) {
        super(message);
    }
}
class USERTRAIL {
    private int val1;
    private int val2;
    public USERTRAIL(int val1, int val2) {
        if (val1 < 0 || val2 < 0) {
```

```
            throw new IllegalValueException("Values cannot be negative");
        }
        this.val1 = val1;
        this.val2 = val2;
    }
    public boolean show() {
        return val1 > 0 && val2 > 0;
    }
}
public class Main {
    public static void main(String[] args) {
        try {
            USERTRAIL trail1 = new USERTRAIL(5, 10);
            System.out.println("show() for trail1: " + trail1.show());

            USERTRAIL trail2 = new USERTRAIL(-3, 7); // This will throw
IllegalValueException
        } catch (IllegalValueException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac Main.java

C:\Users\91826\Desktop\java>java Main
show() for trail1: true
Values cannot be negative

C:\Users\91826\Desktop\java>
```

Assignment NO.IV

Day#4: I/O Streams and Introducing GUI Programming

1. Write a program which take source file and destination file as input as command line arguments.

It copies the source file contents to destination file. If source file does not exist, it should give

appriate message to use. If destination file does not exist, it should be created. If it exists,

program should ask that, “whether you want to overwrite?(Yes/No”).

On the basis of user choice, appropriate action should be taken.

Note: Files may be any type of files like bitmap files, exe files, text files etc.

Ans-

```
import java.io.*;
public class FileCopy {
    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("Usage: java FileCopy <source_file> <destination_file>");
            return;
        }

        String sourceFileName = args[0];
        String destinationFileName = args[1];

        File sourceFile = new File(sourceFileName);
        File destinationFile = new File(destinationFileName);

        if (!sourceFile.exists()) {
            System.out.println("Source file does not exist.");
            return;
        }

        try {
            if (destinationFile.exists()) {
                System.out.println("Destination file already exists.");
```

```
            BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
            System.out.print("Do you want to overwrite? (Yes/No): ");
            String choice = reader.readLine().toLowerCase();

            if (!choice.equals("yes")) {
                System.out.println("Operation cancelled.");
                return;
            }
        } else {
            destinationFile.createNewFile();
        }
        InputStream inputStream = new FileInputStream(sourceFile);
        OutputStream outputStream = new FileOutputStream(destinationFile);

        byte[] buffer = new byte[1024];
        int length;
        while ((length = inputStream.read(buffer)) > 0) {
            outputStream.write(buffer, 0, length);
        }
        inputStream.close();
        outputStream.close();

        System.out.println("File copied successfully.");

    } catch (IOException e) {
        System.out.println("An error occurred: " + e.getMessage());
    }
  }
}
```

```
C:\Users\91826\Desktop\java>javac FileCopy.java

C:\Users\91826\Desktop\java>java FileCopy
Usage: java FileCopy <source_file> <destination_file>
```

2. Write a stream based program which will accept Roll Number, Name, Age and Address

from user

Age and Roll-no should be numeric. Handle with built-in exception.

None of the field should be blank. Handle with custom exception,

Ask user ,whether to write the data in the file

If answer is yes then data is saved into a file as an object(User can write many records in

the file), otherwise terminate the current program

Write another program to display all the records saved into the file

Ans-

```
import java.io.*;
import java.util.*;

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

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

class Student implements Serializable {
    private int rollNumber;
    private String name;
    private int age;
    private String address;

    public Student(int rollNumber, String name, int age, String address) {
        this.rollNumber = rollNumber;
```

```
        this.name = name;
        this.age = age;
        this.address = address;
    }

    public String toString() {
        return "Roll Number: " + rollNumber + ", Name: " + name + ", Age: " + age + ",
Address: " + address;
    }
}

public class StudentRecordManager {
    public static void main(String[] args) {
        try {
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter Roll Number: ");
            int rollNumber = Integer.parseInt(scanner.nextLine());

            System.out.print("Enter Name: ");
            String name = scanner.nextLine().trim();
            if (name.isEmpty()) {
                throw new BlankFieldException("Name cannot be blank.");
            }

            System.out.print("Enter Age: ");
            String ageStr = scanner.nextLine().trim();
            if (ageStr.isEmpty()) {
                throw new BlankFieldException("Age cannot be blank.");
            }
            int age = Integer.parseInt(ageStr);

            System.out.print("Enter Address: ");
            String address = scanner.nextLine().trim();
            if (address.isEmpty()) {
```

```
                throw new BlankFieldException("Address cannot be blank.");
            }

            System.out.print("Do you want to write the data in the file? (Yes/No): ");
            String choice = scanner.nextLine().trim().toLowerCase();
            if (choice.equals("yes")) {
                writeToFile(new Student(rollNumber, name, age, address));
            } else {
                System.out.println("Program terminated.");
            }
        } catch (NumberFormatException e) {
            System.out.println("Invalid data format. Roll number and age should be numeric.");
        } catch (BlankFieldException e) {
            System.out.println("Error: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("An error occurred while writing to the file: " + e.getMessage());
        }
    }

    private static void writeToFile(Student student) throws IOException {
        ObjectOutputStream outputStream = new ObjectOutputStream(new
FileOutputStream("students.dat", true));
        outputStream.writeObject(student);
        outputStream.close();
        System.out.println("Record written to file successfully.");
    }

    public static void displayRecordsFromFile() {
        try {
            ObjectInputStream inputStream = new ObjectInputStream(new
FileInputStream("students.dat"));
            System.out.println("Records from the file:");
            try {
                while (true) {
```

```
                Student student = (Student) inputStream.readObject();
                System.out.println(student);
            }
        } catch (EOFException e) {
        }
        inputStream.close();
    } catch (IOException | ClassNotFoundException e) {
        System.out.println("An error occurred while reading from the file: " +
e.getMessage());
    }
  }
}
```

```
C:\Users\91826\Desktop\java>javac StudentRecordManager.java

C:\Users\91826\Desktop\java>java StudentRecordManager
Enter Roll Number: 101
Enter Name: Deepak
Enter Age: 21
Enter Address: At.Loni Kalbhor
Do you want to write the data in the file? (Yes/No): Yes
Record written to file successfully.

C:\Users\91826\Desktop\java>S
```

3. Write a program using java file system to copy the contents of one file into another.

(This is Self Study Assignment. Refer Java API documentation.)

Ans-

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
public class FileCopyExample {
    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("Usage: java FileCopyExample <source_file>
<destination_file>");
            return;
        }
        String sourceFilePath = args[0];
        String destinationFilePath = args[1];

        Path sourcePath = Paths.get(sourceFilePath);
        Path destinationPath = Paths.get(destinationFilePath);
        try {
            Files.copy(sourcePath, destinationPath);
            System.out.println("File copied successfully.");
        } catch (IOException e) {
            System.out.println("An error occurred while copying the file: " + e.getMessage());
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac FileCopyExample.java

C:\Users\91826\Desktop\java>java FileCopyExample
Usage: java FileCopyExample <source_file> <destination_file>

C:\Users\91826\Desktop\java>
```

4. Write a program which will accept an input String from user

(This is Self Study Assignment. Refer Java API documentation.)

Write the input in the file io.txt

Show size of the file

Read contets from the file and display them on console

Delete io.txt file using File class

Ans-

```
import java.io.*;
public class FileIOExample {
   public static void main(String[] args) {
      try {
         BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
         System.out.print("Enter a string: ");
         String inputString = reader.readLine();

         FileWriter fileWriter = new FileWriter("io.txt");
         fileWriter.write(inputString);
         fileWriter.close();
         System.out.println("Input written to io.txt file.");

         File file = new File("io.txt");
         long fileSize = file.length();
         System.out.println("Size of the file io.txt: " + fileSize + " bytes");

         FileReader fileReader = new FileReader(file);
         BufferedReader fileBufferedReader = new BufferedReader(fileReader);
         String line;
         System.out.println("Contents of the file io.txt:");
         while ((line = fileBufferedReader.readLine()) != null) {
            System.out.println(line);
         }
         fileReader.close();
```

```
            if (file.delete()) {
                System.out.println("File io.txt deleted successfully.");
            } else {
                System.out.println("Failed to delete file io.txt.");
            }

        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac FileIOExample.java

C:\Users\91826\Desktop\java>java FileIOExample
Enter a string: Deepak
Input written to io.txt file.
Size of the file io.txt: 6 bytes
Contents of the file io.txt:
Deepak
File io.txt deleted successfully.

C:\Users\91826\Desktop\java>
```

5. Accept a directory name in form of String from user using proper IO stream. Store it in a

variable. (This is Self Study Assignment. Refer Java API documentation.)

Search if it exists in your system.

If it exists then show all the files present in the directory otherwise print the message

that directory Does not Exists

Ans-

```
import java.io.*;
public class DirectorySearch {
    public static void main(String[] args) {
        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
        try {
            System.out.print("Enter directory name: ");
            String directoryName = reader.readLine();

            File directory = new File(directoryName);

            if (directory.exists() && directory.isDirectory()) {
                System.out.println("Directory exists. Files in the directory:");

                File[] files = directory.listFiles();
                for (File file : files) {
                    System.out.println(file.getName());
                }
            } else {
                System.out.println("Directory does not exist.");
            }
        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac DirectorySearch.java

C:\Users\91826\Desktop\java>java DirectorySearch
Enter directory name: .
Directory exists. Files in the directory:
Advisor.java
BlankFieldException.class
DirectorySearch.class
DirectorySearch.java
FileCopy.class
FileCopy.java
FileCopyExample.class
FileCopyExample.java
FileIOExample.class
FileIOExample.java
Greeter.java
GreeterTest.java
InvalidDataException.class
JDBCExample.class
JDBCExample.java
Student.class
StudentRecordManager.class
StudentRecordManager.java
students.dat
```

6. Create a class “CDR” with the following members:

A_Number

B_Number

duration

calculatedCharge

Write a program which will accept the A_Number, B_Number and duration of Call from

user(Duration is in minutes). Rate that call using 1 Rupee per min rate and store the

calculated charge in “calculatedCharge.txt” file. Write this object to “rated_cdr.txt”.

Note : CDR means Call Details Record

Ans-

```
import java.io.*;

class CDR implements Serializable {
    private String aNumber;
    private String bNumber;
    private int duration;
    private double calculatedCharge;

    public CDR(String aNumber, String bNumber, int duration) {
        this.aNumber = aNumber;
        this.bNumber = bNumber;
        this.duration = duration;
        this.calculatedCharge = duration; // Assuming rate is 1 Rupee per minute
    }

    public double getCalculatedCharge() {
        return calculatedCharge;
    }

    @Override
    public String toString() {
        return "A_Number: " + aNumber + ", B_Number: " + bNumber + ", Duration: " +
duration + " minutes, Calculated Charge: Rs. " + calculatedCharge;
```

```
    }
}

public class CallRatingSystem {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
            System.out.print("Enter A_Number: ");
            String aNumber = reader.readLine();

            System.out.print("Enter B_Number: ");
            String bNumber = reader.readLine();

            System.out.print("Enter duration of call (in minutes): ");
            int duration = Integer.parseInt(reader.readLine());
            double calculatedCharge = duration; // Assuming rate is 1 Rupee per minute

            FileWriter chargeWriter = new FileWriter("calculatedCharge.txt");
            chargeWriter.write(String.valueOf(calculatedCharge));
            chargeWriter.close();
            System.out.println("Calculated charge written to calculatedCharge.txt file.");
            CDR cdr = new CDR(aNumber, bNumber, duration);
            ObjectOutputStream objectOutputStream = new ObjectOutputStream(new
FileOutputStream("rated_cdr.txt"));
            objectOutputStream.writeObject(cdr);
            objectOutputStream.close();
            System.out.println("CDR object written to rated_cdr.txt file.");
        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac CallRatingSystem.java

C:\Users\91826\Desktop\java>java CallRatingSystem
Enter A_Number: 18
Enter B_Number: 22
Enter duration of call (in minutes): 1
Calculated charge written to calculatedCharge.txt file.
CDR object written to rated_cdr.txt file.

C:\Users\91826\Desktop\java>
```

7. Identify that, what functionality need to be added in above assignment , so that it will require object externalization. Take approval for this functionality from the faculty. Then implement this functionality using Object Externalization.

Ans-

```
import java.io.*;
class CDR implements Externalizable {
    private String aNumber;
    private String bNumber;
    private int duration;
    private double calculatedCharge;
    public CDR() {
    }
    public CDR(String aNumber, String bNumber, int duration) {
        this.aNumber = aNumber;
        this.bNumber = bNumber;
        this.duration = duration;
        this.calculatedCharge = duration;     }
    public void writeExternal(ObjectOutput out) throws IOException {
        out.writeObject(aNumber);
        out.writeObject(bNumber);
        out.writeInt(duration);
        out.writeDouble(calculatedCharge);
    }
    public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {
        aNumber = (String) in.readObject();
        bNumber = (String) in.readObject();
        duration = in.readInt();
        calculatedCharge = in.readDouble();
    }
    public String toString() {
        return "A_Number: " + aNumber + ", B_Number: " + bNumber + ", Duration: " +
duration + " minutes, Calculated Charge: Rs. " + calculatedCharge;
```

```
    }
}

public class CallRatingSystem1 {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

            System.out.print("Enter A_Number: ");
            String aNumber = reader.readLine();

            System.out.print("Enter B_Number: ");
            String bNumber = reader.readLine();

            System.out.print("Enter duration of call (in minutes): ");
            int duration = Integer.parseInt(reader.readLine());
            double calculatedCharge = duration; // Assuming rate is 1 Rupee per minute
            CDR cdr = new CDR(aNumber, bNumber, duration);
            ObjectOutputStream objectOutputStream = new ObjectOutputStream(new
FileOutputStream("rated_cdr.txt"));
            objectOutputStream.writeObject(cdr);
            objectOutputStream.close();
            System.out.println("CDR object written to rated_cdr.txt file.");
        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
    }
}
```

```
C:\Users\91826\Desktop\java>javac CallRatingSystem1.java

C:\Users\91826\Desktop\java>java CallRatingSystem1
Enter A_Number: 22
Enter B_Number: 23
Enter duration of call (in minutes): 12
CDR object written to rated_cdr.txt file.

C:\Users\91826\Desktop\java>
```