Advanced Programming Practice

Assignment 6

Name – Adya Singh

Reg No – RA2211003010181

1. Write a Java program to create a class called "Person" with a name and age attribute. Create two instances of the "Person" class, set their attributes using the constructor, and print their name and age.

Code –

```
public class Person {
    private String name;
    private int age;

    // Constructor to initialize the name and age attributes
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    // Getter method to retrieve the name
    public String getName() {
        return name;
    }

    // Getter method to retrieve the age
    public int getAge() {
        return age;
```

```
    }
    public static void main(String[] args) {
        // Create two instances of the Person class
        Person person1 = new Person("John", 30);
        Person person2 = new Person("Alice", 25);

        // Print the name and age of the first person
        System.out.println("Person 1:");
        System.out.println("Name: " + person1.getName());
        System.out.println("Age: " + person1.getAge());

        // Print the name and age of the second person
        System.out.println("\nPerson 2:");
        System.out.println("Name: " + person2.getName());
        System.out.println("Age: " + person2.getAge());
    }
}
```

Output –

2.Write a Java program to create class called "TrafficLight" with attributes for color and duration, and methods to change the color and check for red or green.

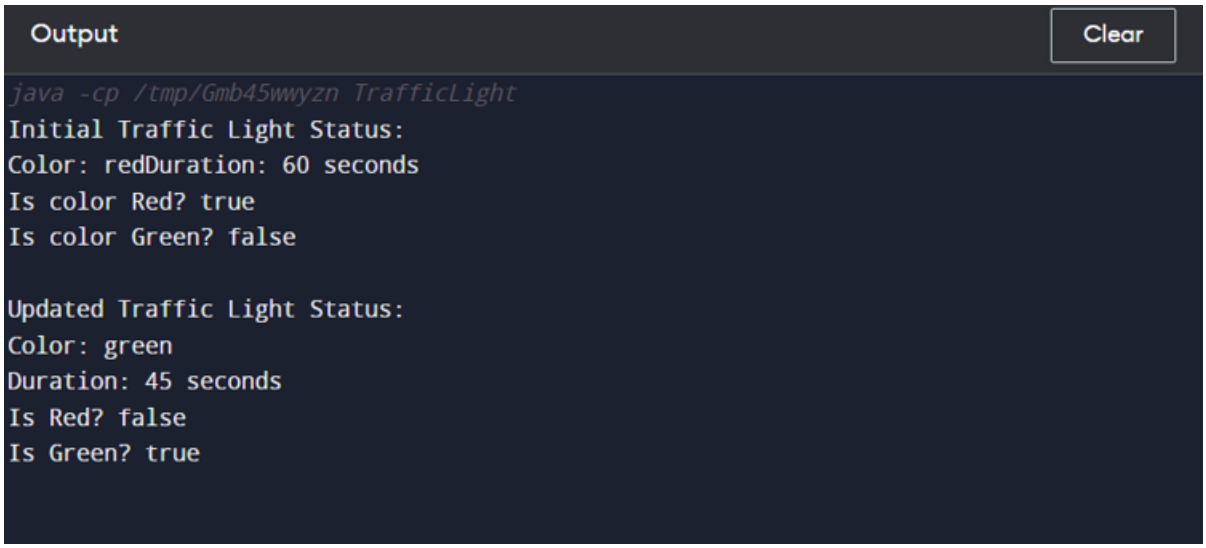
Code –

```
public class TrafficLight {
    private String color;
    private int duration;

    // Constructor to initialize the color and duration attributes
    public TrafficLight(String initialColor, int initialDuration) {
        color = initialColor;
        duration = initialDuration;
    }
    // Method to change the color of the traffic light
    public void changeColor(String newColor) {
        color = newColor;
    }
    // Method to check if the traffic light is currently red
    public boolean isRed() {
        return color.equalsIgnoreCase("red");
    }
    // Method to check if the traffic light is currently green
    public boolean isGreen() {
        return color.equalsIgnoreCase("green");
   }
    // Getter method to retrieve the current color
    public String getColor() {
```

```
    return color;
}
// Getter method to retrieve the current duration
public int getDuration() {
    return duration;
}
public static void main(String[] args) {
    // Create a TrafficLight instance with initial values
    TrafficLight trafficLight = new TrafficLight("red", 60);

    // Display the initial status
    System.out.println("Initial Traffic Light Status:");
    System.out.println("Color: " + trafficLight.getColor());
    System.out.println("Duration: " + trafficLight.getDuration() + " seconds");
    System.out.println("Is color Red? " + trafficLight.isRed());
    System.out.println("Is color Green? " + trafficLight.isGreen());

    // Change the color and duration
    trafficLight.changeColor("green");
    trafficLight.duration = 45;

    // Display the updated status
    System.out.println("\nUpdated Traffic Light Status:");
    System.out.println("Color: " + trafficLight.getColor());
    System.out.println("Duration: " + trafficLight.getDuration() + " seconds");
    System.out.println("Is Red? " + trafficLight.isRed());
```

```
        System.out.println("Is Green? " + trafficLight.isGreen());
    }
}
```

Output –



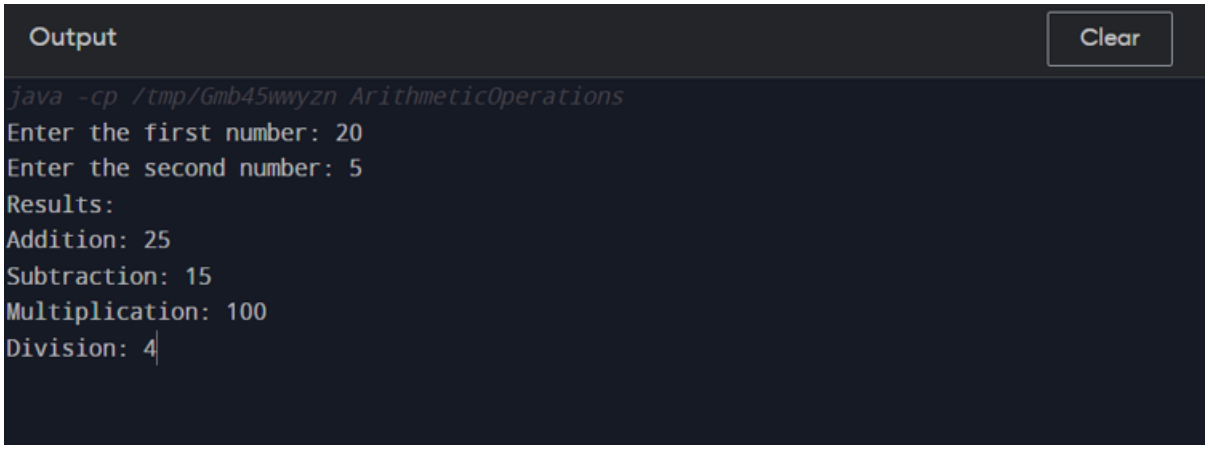
3. Write a Java program to perform arithmetic operations using method overloading.

Code –

```
import java.util.Scanner;
public class ArithmeticOperations {

   // Method to add two integers
   public int add(int a, int b) {
      return a + b;
   }
   // Method to subtract two integers
   public int subtract(int a, int b) {
      return a - b;
   }
   // Method to multiply two integers
   public int multiply(int a, int b) {
      return a * b;
   }
   // Method to divide two integers
   public int divide(int a, int b) {
      if (b == 0) {
         System.out.println("Cannot divide by zero.");
         return 0;
      }
      return a / b;
   }
```

```
    public static void main(String[] args) {
        ArithmeticOperations calculate = new ArithmeticOperations();
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the first number: ");
        int num1 = sc.nextInt();

        System.out.print("Enter the second number: ");
        int num2 = sc.nextInt();

        // Integer arithmetic operations
        int intResultAdd = calculate.add(num1, num2);
        int intResultSubtract = calculate.subtract(num1, num2);
        int intResultMultiply = calculate.multiply(num1, num2);
        int intResultDivide = calculate.divide(num1, num2);

        // Display results
        System.out.println("\nResults:");
        System.out.println("Addition: " + intResultAdd);
        System.out.println("Subtraction: " + intResultSubtract);
        System.out.println("Multiplication: " + intResultMultiply);
        System.out.println("Division: " + intResultDivide);
        sc.close();
    }
}
```

Output –


Output
Clear
java -cp /tmp/Gmb45wwyzn ArithmeticOperations
Enter the first number: 20
Enter the second number: 5
Results:
Addition: 25
Subtraction: 15
Multiplication: 100
Division: 4

4. Write a Java program to create a class called Employee with methods called work() and getSalary(). Create a subclass called HRManager that overrides the work() method and adds a new method called addEmployee().

Code –

```
// Base class Employee
class Employee {
    private String name;
    private double salary;

    public Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    public void work() {
        System.out.println(name + " is performing their job.");
    }

    public double getSalary() {
        return salary;
    }

    public String getName() {
        return name;
    }
}
```

```
// Subclass HRManager
class HRManager extends Employee {
    public HRManager(String name, double salary) {
        super(name, salary);
    }
    @Override
    public void work() {
        System.out.println(getName() + " is managing HR operations.");
    }
    public void addEmployee(Employee employee) {
        System.out.println(getName() + " is adding a new employee: " +
employee.getName());
    }
}
public class EmployeeDemo {
    public static void main(String[] args) {
        // Create an Employee instance
        Employee employee1 = new Employee("John", 50000.0);

        // Create an HRManager instance
        HRManager hrManager = new HRManager("Alice", 70000.0);

        // Call the work() and getSalary() methods for Employee
        employee1.work();
        System.out.println("Employee Salary: $" + employee1.getSalary());

        // Call the work(), getSalary(), and addEmployee() methods for HRManager
```

```
        hrManager.work();
        System.out.println("HR Manager Salary: $" + hrManager.getSalary());
        hrManager.addEmployee(new Employee("Bob", 45000.0));
    }
}
```

Output –

```
John is performing their job.
Employee Salary: $50000.0
Alice is managing HR operations.
HR Manager Salary: $70000.0
Alice is adding a new employee: Bob
```

5. Write a Java program to create a class called Shape with methods called getPerimeter() and getArea(). Create a subclass called Circle that overrides the getPerimeter() and getArea() methods to calculate the area and perimeter of a circle.

Code –

```
import java.lang.Math;

// Base class Shape
class Shape {
    public double getPerimeter() {
        return 0.0; // Default implementation for perimeter
    }

    public double getArea() {
        return 0.0; // Default implementation for area
    }
}

// Subclass Circle
class Circle extends Shape {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }
    @Override
    public double getPerimeter() {
        return 2 * Math.PI * radius; // Perimeter of a circle
    }
    @Override
```

```
    public double getArea() {
        return Math.PI * radius * radius; // Area of a circle
    }
}
public class ShapeDemo {
    public static void main(String[] args) {
        // Create a Circle instance with a radius of 5.0
        Circle circle = new Circle(5.0);

        // Calculate and display the perimeter and area of the circle
        System.out.println("Circle Perimeter: " + circle.getPerimeter());
        System.out.println("Circle Area: " + circle.getArea());
    }
}
```

Output –

```
Circle Perimeter: 31.41592653589793
Circle Area: 78.53981633974483
```

6. Write a Java program to create an interface Sortable with a method sort() that sorts an array of integers in ascending order. Create two classes BubbleSort and SelectionSort that implement the Sortable interface and provide their own implementations of the sort() method.

Code –

```
// Define the Sortable interface
interface Sortable {
    void sort(int[] arr);
}
// Implement BubbleSort class
class BubbleSort implements Sortable {
    @Override
    public void sort(int[] arr) {
        int n = arr.length;
        boolean swapped;

        for (int i = 0; i < n - 1; i++) {
            swapped = false;
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    // Swap arr[j] and arr[j + 1]
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                    swapped = true;      }
            }
    // If no two elements were swapped in inner loop, the array is already sorted
```

```
            if (!swapped) {
                break;      }
        }
    }
}
// Implement SelectionSort class
class SelectionSort implements Sortable {
    @Override
    public void sort(int[] arr) {
        int n = arr.length;

        for (int i = 0; i < n - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < n; j++) {
                if (arr[j] < arr[minIndex]) {
                    minIndex = j;
                }
            }
            // Swap the found minimum element with the first element
            int temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;      }
    }
}
    public class SortingDemo {
    public static void main(String[] args) {
```

```
    int[] arr1 = { 64, 25, 12, 22, 11 };
    int[] arr2 = { 64, 34, 25, 12, 22, 11, 90 };

    // Use BubbleSort to sort arr1
    BubbleSort bubbleSort = new BubbleSort();
    bubbleSort.sort(arr1);
    System.out.println("BubbleSort Sorted Array (arr1):");
    printArray(arr1);

    // Use SelectionSort to sort arr2
    SelectionSort selectionSort = new SelectionSort();
    selectionSort.sort(arr2);
    System.out.println("SelectionSort Sorted Array (arr2):");
    printArray(arr2);
  }
  // Utility method to print an array
  public static void printArray(int[] arr) {
    for (int num : arr) {
      System.out.print(num + " ");    }
    System.out.println();
  }     }
```

Output –

```
BubbleSort Sorted Array (arr1):
11 12 22 25 64
SelectionSort Sorted Array (arr2):
11 12 22 25 34 64 90
```

7. Write a Java program to create an interface Resizable with methods resizeWidth(int width) and resizeHeight(int height) that allow an object to be resized. Create a class Rectangle that implements the Resizable interface and implements the resize methods.

Code –

```
// Define the Resizable interface
interface Resizable {
    void resizeWidth(int width);
    void resizeHeight(int height);
}
// Implement the Rectangle class that implements the Resizable interface
class Rectangle implements Resizable {
    private int width;
    private int height;
    public Rectangle(int width, int height) {
        this.width = width;
        this.height = height;
    }
    @Override
    public void resizeWidth(int width) {
        this.width = width;
    }
    @Override
    public void resizeHeight(int height) {
        this.height = height;      }
```

```
    public int getWidth() {
        return width;
    }
    public int getHeight() {
        return height;    }
}
public class ResizableDemo {
    public static void main(String[] args) {
        // Create a Rectangle instance with initial width and height
        Rectangle rectangle = new Rectangle(10, 20);

        // Print the initial width and height of the rectangle
        System.out.println("Initial Width: " + rectangle.getWidth());
        System.out.println("Initial Height: " + rectangle.getHeight());

        // Resize the rectangle
        rectangle.resizeWidth(30);
        rectangle.resizeHeight(40);
        // Print the resized width and height of the rectangle
        System.out.println("Resized Width: " + rectangle.getWidth());
        System.out.println("Resized Height: " + rectangle.getHeight());
}     }
```

Output –

```
Initial Width: 10
Initial Height: 20
Resized Width: 30
Resized Height: 40
```

8. Write a Java program to create an interface Flyable with a method called fly_obj(). Create three classes Spacecraft, Airplane, and Helicopter that implement the Flyable interface. Implement the fly_obj() method for each of the three classes. Hint :- fly_obj definition – prints the particular object is flying.

Code –

```
// Define the Flyable interface
interface Flyable {
    void fly_obj();
}
// Implement the Spacecraft class that implements the Flyable interface
class Spacecraft implements Flyable {
    @Override
    public void fly_obj() {
        System.out.println("Spacecraft is flying.");
    }
}
// Implement the Airplane class that implements the Flyable interface
class Airplane implements Flyable {
    @Override
    public void fly_obj() {
        System.out.println("Airplane is flying.");
    }
}
// Implement the Helicopter class that implements the Flyable interface
class Helicopter implements Flyable {
```

```
    @Override
    public void fly_obj() {
        System.out.println("Helicopter is flying.");
    }
}
public class FlyableDemo {
    public static void main(String[] args) {
        // Create instances of each flying object
        Flyable spacecraft = new Spacecraft();
        Flyable airplane = new Airplane();
        Flyable helicopter = new Helicopter();

        // Call the fly_obj() method for each flying object
        spacecraft.fly_obj();
        airplane.fly_obj();
        helicopter.fly_obj();
    }
}
```

Output –

```
Spacecraft is flying.
Airplane is flying.
Helicopter is flying.
```

9. Write a Java program to have the arithmetic functions defined in different user-defined packages and incorporate all the packages and perform the function in a single class.

Code –

```
// Package 'addition'
package addition;
public class Addition {
    public static int add(int a, int b) {
        return a + b;
    }
}
// Package 'subtraction'
package subtraction;
public class Subtraction {
    public static int subtract(int a, int b) {
        return a - b;
    }
}
// Package 'multiplication'
package multiplication;
public class Multiplication {
    public static int multiply(int a, int b) {
        return a * b;
    }
}
// Package 'division'
package division;
```

```
public class Division {
    public static double divide(int a, int b) {
        if (b != 0) {
            return (double) a / b;
        } else {
            System.out.println("Division by zero is not allowed.");
            return Double.NaN; // Not-a-Number    }
    }
}
// Main class
import addition.Addition;
import subtraction.Subtraction;
import multiplication.Multiplication;
import division.Division;

public class ArithmeticOperations {
    public static void main(String[] args) {
        int num1 = 10;
        int num2 = 5;

        // Perform arithmetic operations using functions from different packages
        int sum = Addition.add(num1, num2);
        int difference = Subtraction.subtract(num1, num2);
        int product = Multiplication.multiply(num1, num2);
        double quotient = Division.divide(num1, num2);
        // Display the results
```

```
        System.out.println("Sum: " + sum);
        System.out.println("Difference: " + difference);
        System.out.println("Product: " + product);
        System.out.println("Quotient: " + quotient);
    }
}
```

Output –

```
Sum: 15
Difference: 5
Product: 50
Quotient: 2.0
```

10. Create two different packages to compute bubblesort and selection sort. Write a Java program to implement sorting functions in a single class.

Code –

```
public class SortingDemo {
    // Bubble Sort implementation
    public static void bubbleSort(int[] arr) {
        int n = arr.length;
        boolean swapped;

        for (int i = 0; i < n - 1; i++) {
            swapped = false;
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    // Swap arr[j] and arr[j + 1]
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                    swapped = true;
                }
            }
            // If no two elements were swapped in the inner loop, the array is already sorted
            if (!swapped) {
                break;
            }
        }
    }
```

```
// Selection Sort implementation
public static void selectionSort(int[] arr) {
    int n = arr.length;

    for (int i = 0; i < n - 1; i++) {
        int minIndex = i;

        for (int j = i + 1; j < n; j++) {
            if (arr[j] < arr[minIndex]) {
                minIndex = j;
            }
        }
        // Swap the found minimum element with the first element
        int temp = arr[minIndex];
        arr[minIndex] = arr[i];
        arr[i] = temp;
    }
}
public static void main(String[] args) {
    int[] arr1 = {64, 25, 12, 22, 11};
    int[] arr2 = {64, 34, 25, 12, 22, 11, 90};

    // Use Bubble Sort to sort arr1
    bubbleSort(arr1);
    System.out.println("Bubble Sort Sorted Array (arr1):");
```

```
        printArray(arr1);

        // Use Selection Sort to sort arr2
        selectionSort(arr2);

        System.out.println("Selection Sort Sorted Array (arr2):");
        printArray(arr2);
    }
    // Utility method to print an array
    public static void printArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```

Output –

```
Bubble Sort Sorted Array (arr1):
11 12 22 25 64
Selection Sort Sorted Array (arr2):
11 12 22 25 34 64 90
```