DAY - 9

ASSIGNMENT

Task 1 - Task 5

/*Task 1: Array Sorting and Searching (a)

Implement a function called BruteForceSort that sorts an array using the brute force approach. Use this function to sort an array created with InitializeArray. */

```
package TASK;

public class BruteForceSort {
    public static void bruteForceSort(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = i + 1; j < n; j++) {
                if (arr[i] > arr[j]) {
                    int temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
    }

    public static void main(String[] args) {
        int[] arr = initializeArray(10);
        System.out.println("Original array:");
        printArray(arr);

        bruteForceSort(arr);

        System.out.println("\nSorted array:");
        printArray(arr);
    }

    public static int[] initializeArray(int size) {
        int[] arr = new int[size];
        for (int i = 0; i < size; i++) {
            arr[i] = (int) (Math.random() * 100);
        }
        return arr;
    }

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

/*Task 1: Array Sorting and Searching (b)

Write a function named PerformLinearSearch that searches for a specific element in an array and returns the index of the element if found or -1 if not found. */

```
package TASK;
public class linearSearch {
    public static int search(int[] arr, int target) {
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == target) {
                return i;
            }
        } return -1;
    }
    public static void main(String[] args) {
        int[] arr = {5, 3, 8, 2, 9, 1, 6};
        int target = 8;
        int index = search(arr, target);
        if (index != -1) {
            System.out.println("Element " + target + " found at index: " + index);
        } else {
            System.out.println("Element " + target + " not found in the array.");
        }
    }
}
```

/* Task 2: Two-Sum Problem (a)

Given an array of integers, write a program that finds if there are two numbers that add up to a specific target. You may assume that each input would have exactly one solution, and you may not use the same element twice. Optimize the solution for time complexity.*/

```
package TASK;
import java.util.HashMap;
import java.util.Map;
public class TwoSum {
    public static int[] findTwoSum(int[] nums, int target) {
        Map<Integer, Integer> map = new HashMap<>();
        for (int i = 0; i < nums.length; i++) {
            int complement = target - nums[i];
            if (map.containsKey(complement)) {
                return new int[]{map.get(complement), i};
            } map.put(nums[i], i);
        }
        throw new IllegalArgumentException("No two sum solution");
    }
    public static void main(String[] args) {
        int[] nums = {2, 7, 11, 15};
        int target = 9;
        int[] result = findTwoSum(nums, target);
        System.out.println("The two numbers that add up to the target:");
        System.out.println("Index 1: " + result[0]);
        System.out.println("Index 2: " + result[1]);
    }
}
```

/* Task 3: Understanding Functions through Arrays

Write a recursive function named SumArray that calculates and returns the sum of elements in an array, demonstrate with example*/

```
package TASK;
public class RecSumArr {
    public static int sumArray(int[] arr) {
        return sumArrayHelper(arr, 0);
    }
    private static int sumArrayHelper(int[] arr, int index) {
        if (index >= arr.length) {
            return 0;
        }return arr[index] + sumArrayHelper(arr, index + 1);
    }
    public static void main(String[] args) {
        int[] exampleArray = {1, 2, 3, 4, 5};
        int sum = sumArray(exampleArray);
        System.out.println("Sum of the elements in the array: " + sum);
    }
}
```

/* Task 4: Advanced Array Operations (a)

Implement a method SliceArray that takes an array, a starting index, and an end index, then returns a new array containing the elements from the start to the end index. */

```
package TASK;
import java.util.Arrays;
public class SliceArray {
    public static int[] sliceArray(int[] arr, int start, int end) {
        int length = end - start;
        int[] slice = new int[length];
        System.arraycopy(arr, start, slice, 0, length);
        return slice;
    }
    public static void main(String[] args) {
        int[] originalArray = {1, 2, 3, 4, 5, 6, 7, 8, 9};
        int start = 2;
        int end = 6;
        int[] slicedArray = sliceArray(originalArray, start, end);
        System.out.println("Original array: " + Arrays.toString(originalArray));
        System.out.println("Slice from index " + start + " to index " + end + ": "
        + Arrays.toString(slicedArray));
    }
}
```

/* Task 4: Advanced Array Operations (b)

Create a recursive function to find the nth element of a Fibonacci sequence and store the first n elements in an array.*/

```
package TASK;
import java.util.Arrays;
public class Fibb {
    public static long fibonacci(int n) {
        if (n <= 1) {
            return n;
        } return fibonacci(n - 1) + fibonacci(n - 2);
    }
    public static long[] fibonacciSequence(int n) {
        long[] sequence = new long[n];
        for (int i = 0; i < n; i++) {
            sequence[i] = fibonacci(i);
        } return sequence;
    }
    public static void main(String[] args) {
        int n = 10;
        long[] fibonacciArray = fibonacciSequence(n);
        System.out.println("First " + n + " elements of the Fibonacci sequence:");
        System.out.println(Arrays.toString(fibonacciArray));
        int nthElement = 8;
        System.out.println("The " + nthElement + "th element of the Fibonacci sequence is: "
        + fibonacci(nthElement));
    }
}
```

/* Task 5: Iterators and Comparators

Write a custom Comparator to sort a list of Employee objects by their salary and then by name if the salary is the same.*/

```
package TASK;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;
class Employee {
    private String name;
    private double salary;
    public Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }
    public String getName() {
        return name;
    }
    public double getSalary() {
        return salary;
    }
    @Override
    public String toString() {
        return "Name: " + name + "\nSalary = " + salary;
    }
}
class EmployeeComparator implements Comparator<Employee> {
    @Override
    public int compare(Employee emp1, Employee emp2) {
        int salaryComparison = Double.compare(emp1.getSalary(), emp2.getSalary());
        if (salaryComparison == 0) {
            return emp1.getName().compareTo(emp2.getName());
        } return salaryComparison;
    }
}
public class comparators {
    public static void main(String[] args) {
        List<Employee> employees = new ArrayList<>();
        employees.add(new Employee("Kiran", 90000));
        employees.add(new Employee("Somu", 60000));
        employees.add(new Employee("Ram", 58560));
        employees.add(new Employee("Charan", 55900));
        Collections.sort(employees, new EmployeeComparator());
        for (Employee employee : employees) {
            System.out.println(employee);
        }
    }
}
```