# Day 15 Assignment

Name: Mehul Anjikhane Email: mehulanjikhane13@gmail.com

## 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 array;

public class SortArray {

    public static int [] initializeArray(int size) {
        int [] arr = {46,211,89,20,42,15,3,362,52, 35};
            return arr;
    }

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

    public static void main(String[] args) {

        int [] array1 = initializeArray(10);

        System.out.println("Before sorting array: ");
            for(int num : array1) {
                System.out.print(num + " ");
            }
                System.out.println();

        array1 = bruteForceSort(array1);

        System.out.println("Sorted Array:" );
            for(int num : array1) {
                System.out.print(num + " ");
            }
            System.out.println();
```

```
	}

}
```

**Output:**

```
Before sorting array:
46 211 89 20 42 15 3 362 52 35
Sorted Array:
3 15 20 35 42 46 52 89 211 362
```

**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 array;

public class LinearSearch {

	public static void main(String[] args) {

		int [] arr = {15, 52, 65, 89, 20, 4};
		int target = 89;

		LinearSearch ls = new LinearSearch();
		int result = ls.performLinearSearch(arr, target);

		if(result == -1) {
			System.out.println("Element "+ target +" not found
in the array.");
		}
		else {
			System.out.println("Element "+ target +" found at
index "+ result +" the array.");
		}
	}

	public int performLinearSearch(int [] array, int target) {
		for(int i = 0; i <array.length; i++) {
			if(array[i] ==  target) {
				return i;
			}
		}
		return -1;
	}
}
```

**Output:**

```
Element 89 found at index 3 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 array;

import java.util.HashMap;

import java.util.Map;

public class TwoSum {

    public static void main(String[] args) {

        int [] arr = {15, 5, 20, 7, 32};

        int target = 37;


        TwoSum ts = new TwoSum();

        int [] result = ts.twoSum(arr, target);

        if(result != null) {

            System.out.println("Indices of two numbers that add
up to "+target+

                        " are "+result[0]+ " and "+result[1]);

            }

        else {

            System.out.println("No two numbers add up to
"+target);

        }


    }


    public int[] twoSum(int[] array, int target) {

        Map<Integer, Integer> map = new HashMap<>();

        for(int i = 0; i < array.length; i++) {

            int complement = target - array[i];

            if(map.containsKey(complement)) {
                return new int[] {map.get(complement), i};
            }
            map.put(array[i], i);
        }
        return null;
    }

}
```

**Output:**

```
Indices of two numbers that add up to 37 are 1 and 4.
```

**Task 3: Understanding Functions through Arrays**

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

```
package array;

public class RecursiveSum {

    public static int SumArray(int[] arr, int index) {
        //Basic case: if the index is out of bounds,  return 0
        if( index >= arr.length) {
            return 0;
            }
            //Recursive case: add the current element to the sum 
of the rest of the array
            return arr[index] + SumArray(arr, index + 1);
        }

    public static void main(String[] args) {
        int[] arr = {44, 30, 63, 42, 85};
        int sum = SumArray(arr, 0);
        System.out.println("Sum of Array elements: "+sum);
    }
}
```

**Output:**

```
Sum of Array elements: 264
```

## 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 array;

public class SliceArray {

    public static int[] SliceArray(int[] original, int start, int
end) {
        int size = end - start;
        int[] slicedArray = new int[size];
        System.arraycopy(original, start, slicedArray, 0, end -
start);
        return slicedArray;
    }


    public static void main(String[] args) {
        int[] originalArray = {1, 2, 3, 4, 5, 6, 7, 8, 9};
        int start = 2;
        int end = 8;
        System.out.print("Original Array: ");
        for(int element : originalArray) {
            System.out.print(element+ " ");
        }
        System.out.println();
        int[] slicedArray = SliceArray(originalArray, start,
end);
        System.out.print("Sliced Array: ");
        for(int element : slicedArray) {
            System.out.print(element+ " ");
        }
    }
}
```

**Output:**

```
Original Array: 1 2 3 4 5 6 7 8 9
Sliced Array: 3 4 5 6 7 8
```

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

```java
package array;

public class NthFibonacci {

    public static int[] fibonacciArray = new int[100];
    public static int fibonacci(int n) {
        if(n <= 1) {
            return n;
        }
        else {
            int result = fibonacci(n - 1) + fibonacci(n - 2);
            fibonacciArray[n] = result;
            return result;
        }
    }

    public static void main(String[] args) {

        int n = 10;
        int result = fibonacci(n);

        System.out.println("The "+n+"th element of the Fibonacci 
sequence is: "+result);
        System.out.print("The first "+n+" elements of Fibonacci 
sequence are: ");

        for(int i = 0; i <= n; i++) {
            System.out.print(fibonacciArray[i] + " ");
        }
    }
}
```

**Output:**

```
The 10th element of the Fibonacci sequence is: 55
The first 10 elements of Fibonacci sequence are: 0 1 1 2 3 5 8 13 21 
34 55
```