WEEK 7

Question 1: Write a Java function to implement binary search.
Code:

```
import java.util.Arrays;
import java.util.Scanner;
public class BinarySearch {
   public static void binarySearch(int[] arr, int key){
      int idx = -1, low = 0, high = arr.length - 1;
      while (low <= high) {
         int mid = low + (high - low) / 2;
         if (arr[mid] == key) {
            idx = mid; break;
         } else if (arr[mid] < key) {
            low = mid + 1;
         } else
            high = mid - 1;
      }
      if (idx != -1) {
         System.out.printf("Value %d found at index %d or position %d.", key, idx, (idx +1));
      } else
         System.out.printf("Value %d is not present in array!", key);
   }
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the size of array: ");
      int size = scanner.nextInt();
      if(size < 1) {
         System.out.println("Array size must be greater than 0");
         scanner.close(); return;
      }
```

Code:

```
int[] arr = new int[size];
        System.out.printf("Enter %d elements (extra values will be ignored): 
", size);
        for (int i = 0; i < size; i++)
            arr[i] = scanner.nextInt();
        Arrays.sort(arr);
        System.out.println("Sorted Array: " + Arrays.toString(arr));
        System.out.print("Enter value to search: ");
        scanner.nextLine();
        int key = scanner.nextInt();
        binarySearch(arr, key);
        scanner.close();
    }
}
```

Output:

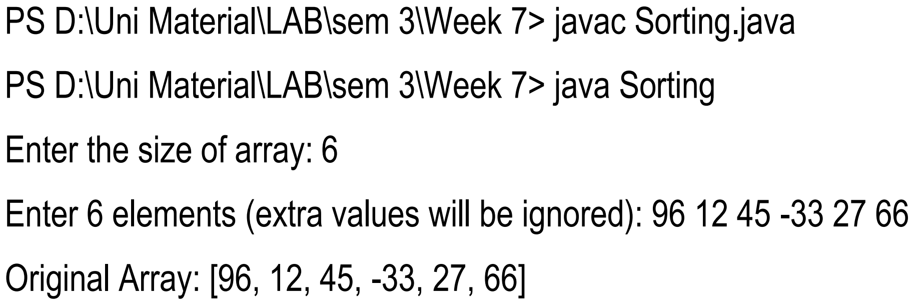
```
PS D:\Uni Material\LAB\sem 3\Week 7> javac BinarySearch.java
PS D:\Uni Material\LAB\sem 3\Week 7> java BinarySearch
Enter the size of array: 5
Enter 5 elements (extra values will be ignored): 12 45 25 34 65
Sorted Array: [12, 25, 34, 45, 65]
Enter value to search: 45
Value 45 found at index 3 or position 4.
PS D:\Uni Material\LAB\sem 3\Week 7>
```

Question 2: Write a Java function to arrange the elements of an array in ascending order (Sorting).

Code:

```
import java.util.Arrays;
import java.util.Scanner;
public class Sorting {
    public static void insertionSort(int[] arr) {
        if(arr.length < 2) return;
        for(int i = 1; i < arr.length; i++){
            int key = arr[i], j = i - 1;
            while(j >= 0 && arr[j] > key){
                arr[j+1] = arr[j];
                j--;
            }
            arr[j+1] = key;
        }
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the size of array: ");
        int size = scanner.nextInt();
        if(size < 1) {
            System.out.println("Array size must be greater than 1");
            scanner.close(); return;
        }
        int[] arr = new int[size];
        System.out.printf("Enter %d elements (extra values will be ignored): ", size);
        for (int i = 0; i < size; i++)
            arr[i] = scanner.nextInt();
        System.out.println("Original Array: " + Arrays.toString(arr));
        insertionSort(arr);
        System.out.println("Sorted Array: " + Arrays.toString(arr));
    }
}
```

Output:



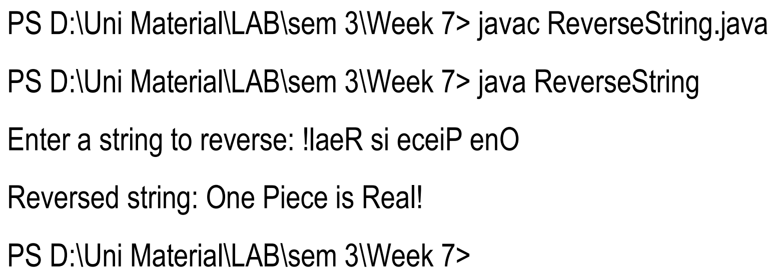
```
PS D:\Uni Material\LAB\sem 3\Week 7> javac Sorting.java
PS D:\Uni Material\LAB\sem 3\Week 7> java Sorting
Enter the size of array: 6
Enter 6 elements (extra values will be ignored): 96 12 45 -33 27 66
Original Array: [96, 12, 45, -33, 27, 66]
```

Question 3: Write a program to reverse a given string.

Code:

```
import java.util.Scanner;
public class ReverseString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string to reverse: ");
        String original = scanner.nextLine(); scanner.close();
        if (original.equals("")) {
            System.out.println("Nothing to reverse."); return;
        }
        if (original.length() == 1) {
            System.out.println("Reversed string: " + original); return;
        }
        String reversed = "";
        for (int i = original.length() - 1; i >= 0; i--)
            reversed = reversed + original.charAt(i);
        System.out.println("Reversed string: " + reversed);
    }
}
```

Output:



```
PS D:\Uni Material\LAB\sem 3\Week 7> javac ReverseString.java
PS D:\Uni Material\LAB\sem 3\Week 7> java ReverseString
Enter a string to reverse: !laeR si eceiP enO
Reversed string: One Piece is Real!
PS D:\Uni Material\LAB\sem 3\Week 7>
```

Question 4: Write a program to check whether a given string is palindrome or not.

Code:

```
import java.util.Scanner;
public class PalindromeString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine(); scanner.close();
        if (str.equals("")) {
            System.out.println("String is empty."); return;
        }
        int len = str.length();
        String original = str.toLowerCase();
        boolean isPalindrome = true;
        for(int i = 0; i < len/2; i++){
            if(original.charAt(i) != original.charAt(len-1-i))
                isPalindrome = false; break;
        }
        if (isPalindrome) {
            System.out.println("The string is a palindrome!");
        } else
            System.out.println("The string is NOT a palindrome.");
    }
}
```

Output:

```
PS D:\Uni Material\LAB\sem 3\Week 7> javac PalindromeString.java
PS D:\Uni Material\LAB\sem 3\Week 7> java PalindromeString
Enter a string: RACECAR
The string is a palindrome!
PS D:\Uni Material\LAB\sem 3\Week 7>
```

Question 5: Write a program to implement factorial of a number through recursion

Code:

```java
import java.util.Scanner;
public class RecursiveFactorial {
    public static long factorial(int n) {
        if (n == 0 || n == 1)
            return 1;
        return n * factorial(n - 1);
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a non-negative integer: ");
        int n = scanner.nextInt();
        if (n < 0) {
            System.out.println("Factorial is not defined for negative numbers.");
        } else {
            long result = factorial(n);
            System.out.printf("%d! = %d%n", n, result);
        }
        scanner.close();
    }
}
```

Output:

```
PS D:\Uni Material\LAB\sem 3\Week 7> javac RecursiveFactorial.java
PS D:\Uni Material\LAB\sem 3\Week 7> java RecursiveFactorial
Enter a non-negative integer: 17
17! = 355687428096000
PS D:\Uni Material\LAB\sem 3\Week 7>
```

Question 6: Write a program to implement Fibonacci series of a number with and without recursion.

Code:

```
import java.util.Scanner;
public class Fibonacci {
    public static long fibRecursive(int n) {
        if (n == 0) return 0;
        if (n == 1) return 1;
        return fibRecursive(n - 1) + fibRecursive(n - 2);
    }
    public static void fibIterative(int n) {
        long a = 0, b = 1;
        for (int i = 0; i < n; i++) {
            System.out.print(a + " ");
            long next = a + b;
            a = b;
            b = next;
        }
        System.out.println();
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of terms: ");
        int n = scanner.nextInt();
        if (n < 0) {
            System.out.println("Number must be non-negative.");
            scanner.close(); return;
        }
        System.out.println("Choose method:");
        System.out.println("1. Recursive");
        System.out.println("2. Iterative");
```

Code:

```
        System.out.print("Enter your choice (1 or 2): ");
        int choice = scanner.nextInt(); scanner.close();
        switch(choice) {
            case 1:
                System.out.print("Fibonacci series (recursive): ");
                for (int i = 0; i < n; i++)
                    System.out.print(fibRecursive(i) + " ");
                break;
            case 2:
                System.out.print("Fibonacci series (iterative): ");
                fibIterative(n);
                break;
            default:
                System.out.println("Invalid choice!");
        }
    }
}
```

Output:

```
PS D:\Uni Material\LAB\sem 3\Week 7> javac Fibonacci.java
PS D:\Uni Material\LAB\sem 3\Week 7> java Fibonacci
Enter number of terms: 9
Choose method:
1. Recursive
2. Iterative
Enter your choice (1 or 2): 2
Fibonacci series (iterative): 0 1 1 2 3 5 8 13 21
PS D:\Uni Material\LAB\sem 3\Week 7>
```

OPTIONAL

Question 7: Write a Java function to find the greatest common divisor (GCD) of two numbers with and without using recursion.

Code:

```
import java.util.Scanner;
public class GCD {
    public static int gcdRecursive(int a, int b) {
        if (b == 0) return a;
        return gcdRecursive(b, a % b);
    }
    public static int gcdIterative(int a, int b) {
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter first number: ");
        int num1 = scanner.nextInt();
        System.out.print("Enter second number: ");
        int num2 = scanner.nextInt();
        if (num1 <= 0 || num2 <= 0) {
            System.out.println("Numbers must be positive integers.");
            scanner.close(); return;
        }
        System.out.println("Choose method:");
        System.out.println("1. Recursive");
        System.out.println("2. Iterative");
```

Code:

```
        System.out.print("Enter your choice (1 or 2): ");
        int choice = scanner.nextInt();
        int gcd;
        switch(choice) {
            case 1:
                gcd = gcdRecursive(num1, num2);
                System.out.printf("GCD (recursive) of %d and %d is %d%n",
num1, num2, gcd);
                break;
            case 2:
                gcd = gcdIterative(num1, num2);
                System.out.printf("GCD (iterative) of %d and %d is %d%n",
num1, num2, gcd);
                break;
            default:
                System.out.println("Invalid choice!");
        }
        scanner.close();
    }
}
```

Output:

```
PS D:\Uni Material\LAB\sem 3\Week 7> javac GCD.java
PS D:\Uni Material\LAB\sem 3\Week 7> java GCD
Enter first number: 95
Enter second number: 67
Choose method:
1. Recursive
2. Iterative
Enter your choice (1 or 2): 1
GCD (recursive) of 95 and 67 is 1
PS D:\Uni Material\LAB\sem 3\Week 7>
```

Question 8: Write a program to check whether two strings are anagrams of each other (“listen” and “silent” are

Code:

```
import java.util.Arrays;
import java.util.Scanner;
public class Anagrams {
    public static boolean areAnagrams(String str1, String str2) {
        str1 = str1.toLowerCase();
        str2 = str2.toLowerCase();
        str1 = str1.replaceAll("\\s+", "");
        str2 = str2.replaceAll("\\s+", "");
        if (str1.length() != str2.length()) return false;
        char[] arr1 = str1.toCharArray();
        char[] arr2 = str2.toCharArray();
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        return Arrays.equals(arr1, arr2);
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter first string: ");
        String s1 = scanner.nextLine();
        System.out.print("Enter second string: ");
        String s2 = scanner.nextLine();
        if (areAnagrams(s1, s2))
            System.out.println("The strings are anagrams!");
        else
            System.out.println("The strings are NOT anagrams.");
        scanner.close();
    }
}
```

Output:

```
PS D:\Uni Material\LAB\sem 3\Week 7> javac Anagrams.java
PS D:\Uni Material\LAB\sem 3\Week 7> java Anagrams
Enter first string: A Gentleman
Enter second string: Elegant Man
The strings are anagrams!
PS D:\Uni Material\LAB\sem 3\Week 7>
```

Question 9: Implement quick sort using recursion

Code:

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

public class QuickSort {
    public static void quickSort(int[] arr, int low, int high) {
        if (low < high) {
            int pivotIndex = partition(arr, low, high);
            quickSort(arr, low, pivotIndex - 1);
            quickSort(arr, pivotIndex + 1, high);
        }
    }
    private static int partition(int[] arr, int low, int high) {
        int pivot = arr[high];
        int i = low - 1;
        for (int j = low; j < high; j++) {
            if (arr[j] <= pivot) {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
```

Code:

```
        int temp = arr[i + 1];
        arr[i + 1] = arr[high];
        arr[high] = temp;
        return i + 1;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter size of array: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        System.out.print("Enter " + n + " elements:");
        for (int i = 0; i < n; i++) arr[i] = scanner.nextInt();
        scanner.close();
        System.out.println("Original Array: " + Arrays.toString(arr));
        quickSort(arr, 0, n - 1);
        System.out.println("Sorted Array:   " + Arrays.toString(arr));
    }
}
```

Output:

```
PS D:\Uni Material\LAB\sem 3\Week 7> javac QuickSort.java
PS D:\Uni Material\LAB\sem 3\Week 7> java QuickSort
Enter size of array: 6
Enter 6 elements:7 56 89 24 46 30
Original Array: [7, 56, 89, 24, 46, 30]
Sorted Array:   [7, 24, 30, 46, 56, 89]
PS D:\Uni Material\LAB\sem 3\Week 7>
```