WEEK 7

1. Write a Java function to implement binary search.

week7_1.java × week7_2.java week7_3.java week7_4.java week7_5.java week7_6.java

```java
import java.util.Scanner;

public class week7_1 {  Stephen047
    
    static int search(int[] arr, int x){  1 usage  Stephen047
        int low = 0;
        int high = arr.length - 1;
        while(low <= high){
            int mid = (low + high)/2;
            if(arr[mid] == x) return mid;
            else if(x < arr[mid]) high = mid-1;
            else low = mid + 1;
        }
        return -1;
    }

    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the length :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter elements : ");
        for (int i = 0; i < l; i++) {
            arr[i] = sc.nextInt();
        }
        System.out.print("Enter the target : ");
        int x = sc.nextInt();

        int index = search(arr, x);
        if(index != -1)
            System.out.println("Found at index "+index);
        else
            System.out.println("Not Found");
    }
}

```

```
Enter the length :
5
Enter elements :
1 2 3 4 5
Enter the target : 4
Found at index 3
```

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

WEEK 7

week7_1.java | week7_2.java | week7_3.java | week7_4.java | week7_5.java

```java
import java.util.Scanner;

public class week7_2 {  Stephen047

    static void sort(int[] arr){  1 usage  Stephen047
        for (int i = 0; i < arr.length - 1; i++) {
            boolean swapped = false;
            for (int j = 0; j < arr.length - 1 - i; j++)
                if (arr[j] > arr[j+1]) {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                    swapped = true;
                }
            if(!swapped) return;
        }
    }

    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the length :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter elements : ");
        for (int i = 0; i < l; i++) {
            arr[i] = sc.nextInt();
        }
        sort(arr);
        System.out.println("Sorted Array :");
        for (int j : arr) {
            System.out.print(j + " ");
        }
    }
}
```

```
Enter the length :
5
Enter elements :
1 3 4 2 5
Sorted Array :
1 2 3 4 5
```

3. Write a program to reverse a given string.

WEEK 7

week7_1.java week7_2.java week7_3.java × week7_4.java

```java
import java.util.Scanner;

public class week7_3 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a String : ");
        String str = sc.next();
        String rev = "";
        for (int i = 0; i < str.length(); i++) {
            rev = str.charAt(i) + rev;
        }
        System.out.println("Reversed string : "+rev);
    }
}

```

```
Enter a String : SHaheer
Reversed string : reehaHS
```

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

week7_1.java week7_2.java week7_3.java week7_4.java ×

```java
import java.util.Scanner;

public class week7_4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a String : ");
        String str = sc.next().toLowerCase();
        String rev = "";
        for (int i = 0; i < str.length(); i++) {
            rev = str.charAt(i) + rev;
        }
        if (str.equals(rev)) System.out.println("Palindrome");
        else System.out.println("Not Palindrome");
    }
}

```

```
Enter a String : Dad
Palindrome
```

5. Write a program to implement factorial of a number through recursion.

WEEK 7

week7_1.java week7_2.java week7_3.java

```java
import java.util.Scanner;

public class week7_5 {

    static int fact(int n){  2 usages
        if (n <= 1) return 1;
        return n * fact( n: n-1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int n = sc.nextInt();
        System.out.println(fact(n));
    }
}
```

```
Enter a number : 6
720
```

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

week7_1.java week7_2.java week7_3.java week7_4.java wee

```java
public class week7_6 {  Stephen047

    static void fib(int n){  1 usage  Stephen047
        if (n >= 1) System.out.print("0 ");
        if (n >= 2) System.out.print("1 ");
        if (n > 2) {
            int a = 0;
            int b = 1;
            for (int i = 3; i <= n; i++) {
                int c = a + b;
                a = b;
                b = c;
                System.out.print(c + " ");
            }
        }
    }
    static void fib_rec(int n, int a, int b){  2 usages  Stephen047
        if(n > 0){
            System.out.print(a + " ");
            fib_rec( n: n-1, b,  b: a+b);
        }
    }

    public static void main(String[] args) {  Stephen047
        fib( n: 10);
        System.out.println("\nWith recursion : ");
        fib_rec( n: 10,  a: 0,  b: 1);
    }
}
```

```
0 1 1 2 3 5 8 13 21 34
With recursion :
0 1 1 2 3 5 8 13 21 34
```

WEEK 7

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

week7_4.java | week7_5.java | week7_6.java | week7_7.java

```java
public class week7_7 {  Stephen047
    static int gcd(int a, int b){  1 usage  Stephen047
        while(a%b != 0){
            int r = a%b;
            a = b;
            b = r;
        }
        return b;
    }
    static int gcd_rec(int a, int b){  2 usages  Stephen047
        if (a%b == 0) return b;
        return gcd_rec(b,  b: a%b);
    }

    public static void main(String[] args) {  Stephen047
        System.out.println(gcd( a: 15,  b: 100));
        System.out.println("With Recursion : ");
        System.out.println(gcd_rec( a: 15,  b: 100));
    }
}
```

```
5
With Recursion :
5
```

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

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

public class week7_8 {  Stephen047

    static String sort(String str){  2 usages  Stephen047
        char[] temp = str.toLowerCase().toCharArray();
        Arrays.sort(temp);
        return new String(temp);
    }

    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.print("String 1 : ");
        String s1 = sc.nextLine();
        System.out.print("String 2 : ");
        String s2 = sc.nextLine();
        if ((sort(s1).trim()).equals(sort(s2).trim()))
            System.out.println("Anagrams");
        else System.out.println("Not Anagrams");
    }
}
// Examples - Listen, Silent; Heart, Earth; The Eyes, They see; Debit Card, Bad Credit;
```

WEEK 7

```
String 1 : Debit Card
String 2 : Bad Credit
Anagrams
```

9. Implement quick sort using recursion.

week7_4.java week7_5.java week7_6.java week7_7.java week7_8.java

```java
public class week7_9 {  Stephen047
    static int partition(int[] arr, int low, int high) {  1 usage  Stephen047

        int pivot = arr[high];
        int i = low - 1;
        for (int j = low; j < high; j++) {
            if (arr[j] < pivot) {
                i++;
                swap(arr, i, j);
            }
        }
        //putting pivot (high) in front of last exchange
        swap(arr,  i: i + 1, high);
        return i + 1;
    }
    static void swap(int[] arr, int i, int j) {  2 usages  Stephen047
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }

    static void quickSort(int[] arr, int low, int high) {  3 usages  Stephen047
        if (low < high) {
            //pi is pivot index
            int pi = partition(arr, low, high);
            quickSort(arr, low,  high: pi - 1);
            quickSort(arr,  low: pi + 1, high);
        }
    }
```

```
C:\Program Files\Ja
1 5 7 8 9 10
Process finished wit
```