**Q. 1) How are Strings stored in memory?**

- Strings are objects of the `String` class.
- String objects are stored in **heap memory**.
- Java maintains a special memory area called the **String Constant Pool (SCP)**.
- String literals (created using double quotes) are stored in the SCP.
- If the same literal already exists in the SCP, no new object is created.
- Multiple references can point to the same pooled String object.
- Strings created using the `new` keyword are stored in normal heap memory.
- Using `new String()` creates a separate object even if the same value exists in the SCP.
- Strings are **immutable**, meaning their value cannot be changed after creation.

## Q. 2) What is an array in Java?

- An array is a collection of similar data types.
- All elements in an array are stored in contiguous memory locations.
- An array is an object in Java.
- Arrays are stored in heap memory.
- The size of an array is fixed at the time of creation.
- Each element is accessed using an index.
- Array indexing starts from **0** and ends at `length - 1`.
- The `length` property is used to get the size of the array.

## Q .3) How do you declare and initialize an array?

- Int [] arr;
- int[] arr = new int[5];
- int[] arr = {10, 20, 30, 40, 50};
- int[] arr;
- arr = new int[] {1, 2, 3};

## Q. 4) What are jagged arrays?

- A jagged array is an array of arrays.
- In a jagged array, each row can have a different number of columns.

- Jagged arrays are used to represent non-rectangular data.
- They are created using multi-dimensional array syntax.
- Memory is allocated row by row, not as a single block.
- Jagged arrays help save memory when rows have unequal sizes.
- Each row is an independent array.

## Q. 5) What is an ArrayIndexOutOfBoundsException?

- `ArrayIndexOutOfBoundsException` is a runtime exception in Java.
- It occurs when an array is accessed with an invalid index.
- Valid array indices range from `0` to `length - 1`.
- Accessing a negative index causes this exception.
- Accessing an index greater than or equal to array length causes this exception.
- It is part of the `java.lang` package.
- The program compiles successfully but fails at runtime.
- Java performs automatic bounds checking to prevent memory corruption.

## Q. 6) Write a program to count even and odd numbers in an array.

```
import java.util.Scanner;

public class Count {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();

        int[] arr = new int[n];

        int evenCount = 0;
        int oddCount = 0;

        System.out.println("Enter array elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();

            if (arr[i] % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }

        System.out.println("Even numbers count: " + evenCount);
        System.out.println("Odd numbers count: " + oddCount);
    }
}
```

**Output:-**

Enter number of elements: 5

Enter array elements:

1

2

3

4

5

Even numbers count: 2

Odd numbers count: 3

**Q. 7) Write a program to copy all elements of one array to another.**

```
import java.util.Scanner;

class CopyArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();

        int[] arr1 = new int[n];
        int[] arr2 = new int[n];

        System.out.println("Enter elements of first array:");
        for (int i = 0; i < n; i++) {
            arr1[i] = sc.nextInt();
        }

        for (int i = 0; i < n; i++) {
            arr2[i] = arr1[i];
        }

        System.out.println("Elements of second array:");
        for (int i = 0; i < n; i++) {
            System.out.print(arr2[i] + " ");
        }
```

```
    }
}
```

**Output:-**

Enter number of elements: 5

Enter elements of first array:

1

2

3

4

5

Elements of second array:

1 2 3 4 5

**Q. 8) Write a program to search an element in an array.**

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

public class Search{

    public static void main(String[] args){

        int key,n;

        Boolean flag=true;
        Scanner sc=new Scanner(System.in);

        System.out.println("enter n value");
        n=sc.nextInt();


        int[] arr=new int[n];

        System.out.println("Enter Data for Arrays");
        for(int i=0;i<n;i++){
            arr[i]=sc.nextInt();
        }

        System.out.println("enter element for search");
```

```
        key=sc.nextInt();

        for(int x : arr){
            if(x==key){
                flag=false;
                break;
            }
        }
        if(flag)
            System.out.println("not found");
        else
            System.out.println("found");
    }
}
```

**Output:**

enter n value

5

Enter Data for Arrays

11

2

3

4

5

enter element for search

4

found

**Q .9) Write a program to merge two arrays.**

```
import java.util.Scanner;

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

```
        System.out.print("Enter size of first array: ");
        int n1 = sc.nextInt();

        int[] arr1 = new int[n1];
        System.out.println("Enter elements of first array:");
        for (int i = 0; i < n1; i++) {
            arr1[i] = sc.nextInt();
        }

        System.out.print("Enter size of second array: ");
        int n2 = sc.nextInt();

        int[] arr2 = new int[n2];
        System.out.println("Enter elements of second array:");
        for (int i = 0; i < n2; i++) {
            arr2[i] = sc.nextInt();
        }

        int[] merged = new int[n1 + n2];

        // Copy first array
        for (int i = 0; i < n1; i++) {
            merged[i] = arr1[i];
        }

        // Copy second array
        for (int i = 0; i < n2; i++) {
            merged[n1 + i] = arr2[i];
        }

        System.out.println("Merged array:");
        for (int i = 0; i < merged.length; i++) {
            System.out.print(merged[i] + " ");
        }
    }
}
```

**Output:-**

Enter size of first array: 5

Enter elements of first array:

1

2

3

4

5

Enter size of second array: 4

Enter elements of second array:

1

2

3

4

Merged array:

1 2 3 4 5 1 2 3 4