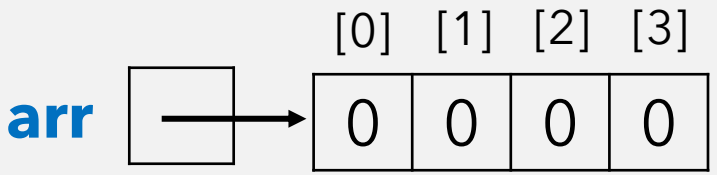
Passing Arrays as Parameters

Recap – Passing Parameters

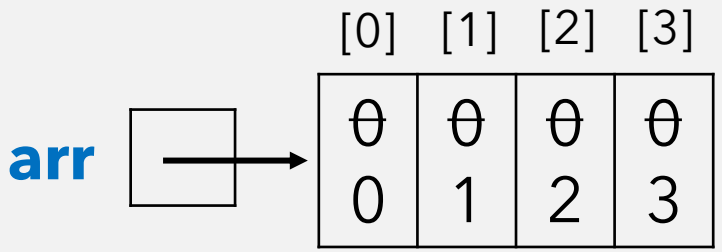
- When a **primitive type** is passed as a parameter, the **value** is copied

- When an **object** is passed as a parameter, the **reference** is copied
  - For example, Scanners, Strings, and Arrays.

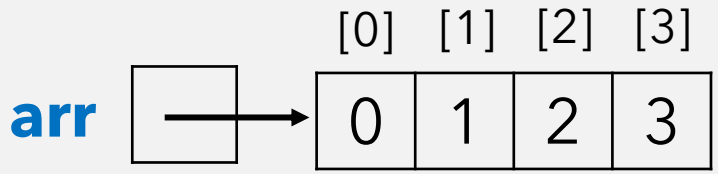
```java
import java.util.Arrays;

public class ArraysParameterDemo {
    public static void main(String []args) {
        int[] arr = new int[4];









}
```



```java
import java.util.Arrays;

public class ArraysParameterDemo {
    public static void main(String []args) {
        int[] arr = new int[4];

        for (int i = 0; i < arr.length; ++i) {
            arr[i] = i;
        }









}
```

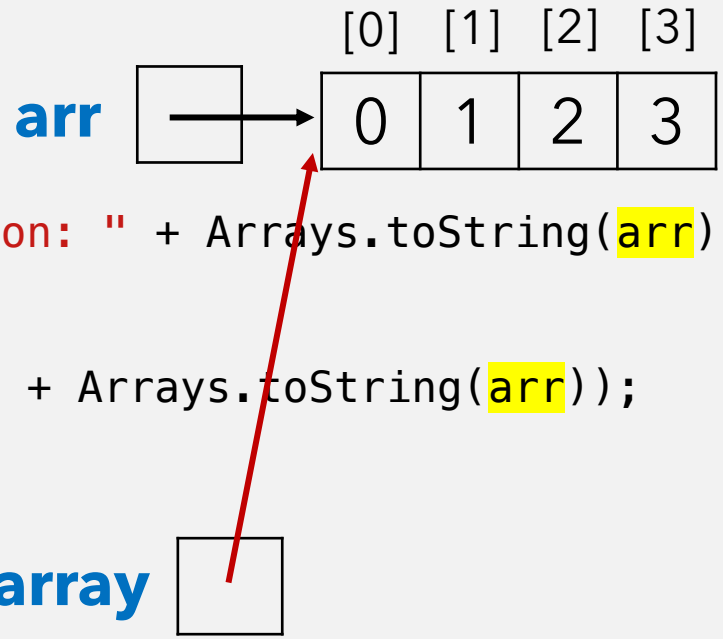


```java
import java.util.Arrays;

public class ArraysParameterDemo {
    public static void main(String []args) {
        int[] arr = new int[4];

        for (int i = 0; i < arr.length; ++i) {
            arr[i] = i;
        }
        System.out.println("Array after initialization: " + Arrays.toString(arr));

        incrementAll(arr);
        System.out.println("Array after increment: " + Arrays.toString(arr));
    }




}
```



```java
import java.util.Arrays;

public class ArraysParameterDemo {
    public static void main(String []args) {
        int[] arr = new int[4];

        for (int i = 0; i < arr.length; ++i) {
            arr[i] = i;
        }
        System.out.println("Array after initialization: " + Arrays.toString(arr));

        incrementAll(arr);
        System.out.println("Array after increment: " + Arrays.toString(arr));
    }

    public static void incrementAll(int[] array) {



    }
}
```

In method call, use the name arr only. NO [ ]

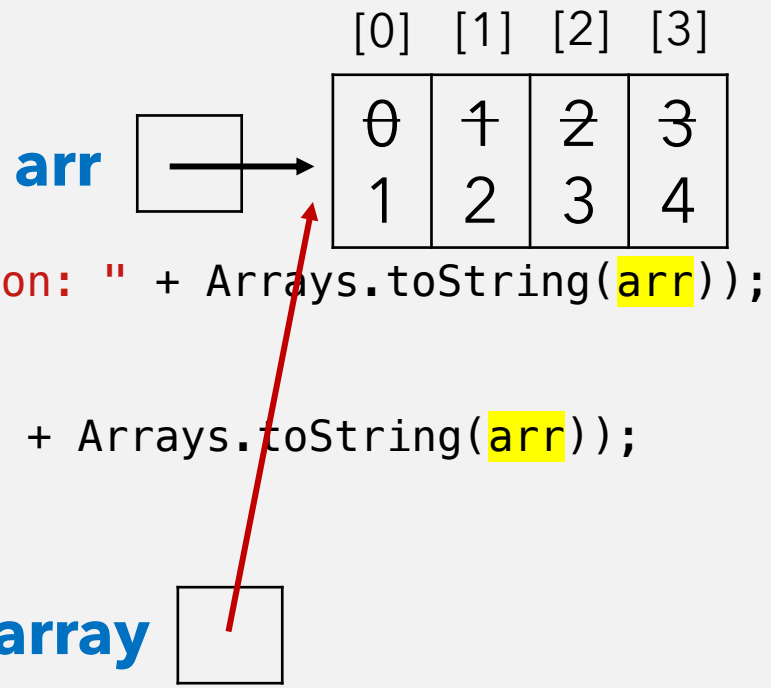


<type>[]

Name does not matter, reference matters.

```java
import java.util.Arrays;

public class ArraysParameterDemo {
    public static void main(String []args) {
        int[] arr = new int[4];

        for (int i = 0; i < arr.length; ++i) {
            arr[i] = i;
        }
        System.out.println("Array after initialization: " + Arrays.toString(arr));

        incrementAll(arr);
        System.out.println("Array after increment: " + Arrays.toString(arr));
    }

    public static void incrementAll(int[] array) {
        for (int i = 0; i < array.length; ++i) {
            array[i]++;
        }
    }
}
```

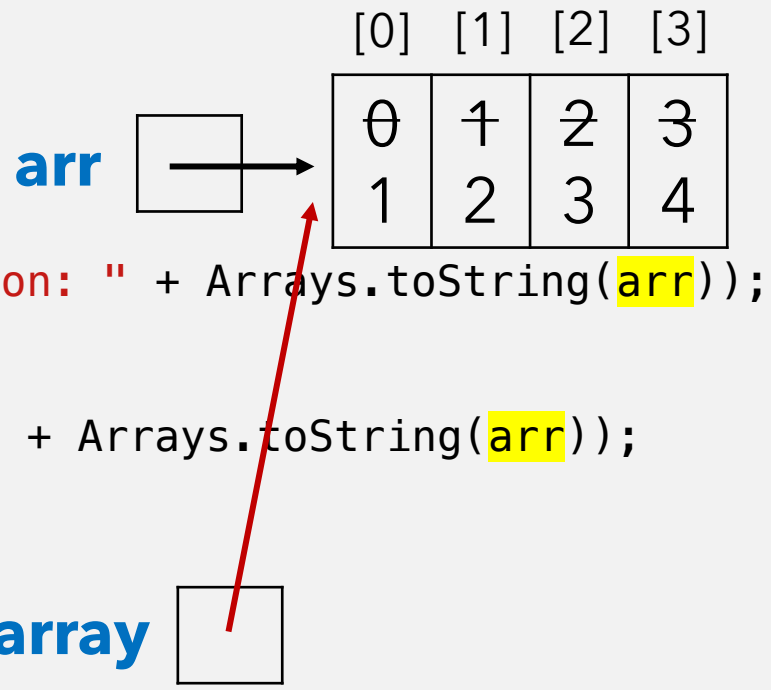
In method call, use the name arr only. NO [ ]



<type>[]

Name does not matter, reference matters.

```java
import java.util.Arrays;

public class ArraysParameterDemo {
    public static void main(String []args) {
        int[] arr = new int[4];

        for (int i = 0; i < arr.length; ++i) {
            arr[i] = i;
        }
        System.out.println("Array after initialization: " + Arrays.toString(arr));

        incrementAll(arr);
        System.out.println("Array after increment: " + Arrays.toString(arr));
    }

    public static void incrementAll(int[] array) {
        for (int i = 0; i < array.length; ++i) {
            array[i]++;
        }
    }
}
```




```
$ javac ArraysParameterDemo.java
$ java ArraysParameterDemo
Array after initialization: [0, 1, 2, 3]
Array after increment: [1, 2, 3, 4]
```

Array itself is modified