

به نام خدا



برنامه‌سازی ییشوفته

دانشگاه شهید بهشتی · دانشکده مهندسی کامپیوتر

دکتر مجتبی وحیدی اصل

آرایه‌های یک بعدی

نازین زهرا فرهنگ

فهرست مطالب

1. معرفی آرایه ها
2. مراحل تعریف آرایه ها
3. متغیرهای اندیس
4. مقداردهی اولیه آرایه ها
5. پردازش آرایه ها (حل چند مثال)
6. ارسال آرایه ها به متدها

انگیزه

این فصل را با یک سوال شروع می کنیم.

- 10 عدد را از کاربر بگیرید، میانگین آنها را حساب کرده و بگویید چه تعدادی از آنها از میانگین بزرگتر هستند؟

```
In [ ]: !pip install jbang  
import jbang
```

```
jbang.exec("trust add https://github.com/jupyter-java")
jbang.exec("install-kernel@jupyter-java")
```

Failed to start the Kernel 'Java (JBang Python)'.

View Jupyter log for further details. spawn EINVAL

```
In [ ]: public class AnalyzeNumbers {
    public static void main(String[] args) {
        final int NUMBER_OF_ELEMENTS = 10;
        double[] numbers = new double[NUMBER_OF_ELEMENTS];
        double sum = 0;

        java.util.Scanner input = new java.util.Scanner(System.in);
        for (int i = 0; i < NUMBER_OF_ELEMENTS; i++) {
            System.out.print("Enter a new number: ");
            numbers[i] = input.nextDouble();
            sum += numbers[i];
        }

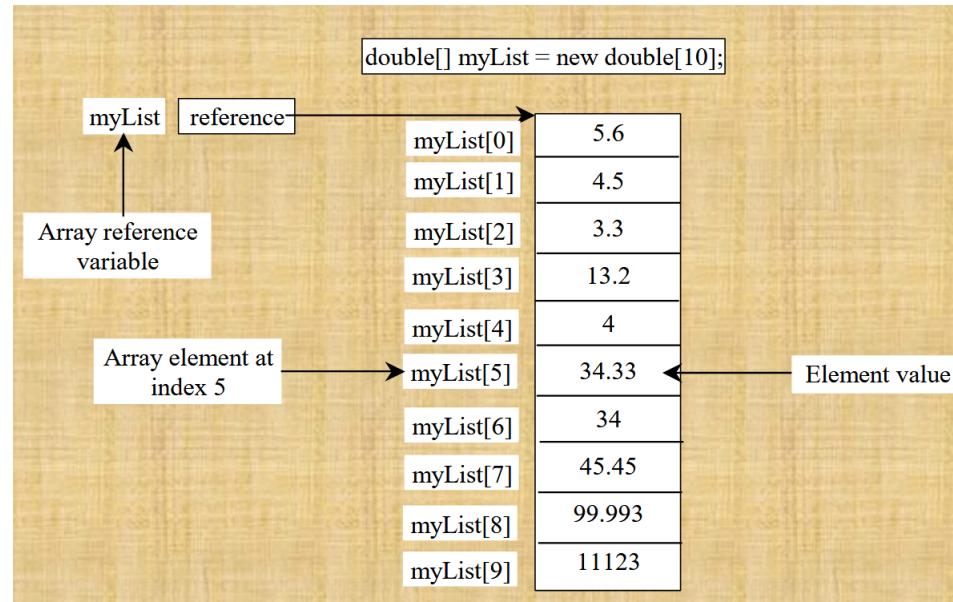
        double average = sum / NUMBER_OF_ELEMENTS;

        int count = 0; // The number of elements above average
        for (int i = 0; i < NUMBER_OF_ELEMENTS; i++)
            if (numbers[i] > average)
                count++;

        System.out.println("Average is " + average);
        System.out.println("Number of elements above the average " + count);
    }
}
```

معرفی آرایه ها

یک آرایه مثل مجموعه ای از متغیر ها است. که هر متغیر در آن یک عنصر آرایه ساختمان داده ای است حاوی مجموعه ای از داده های هم نوع نامیده می شود.



مراحل تعریف آرایه ها

- اعلان آرایه

```
;double[] nums
```

ایجاد آرایه

```
;nums = new double[4]
```

مقداردهی آرایه

- مقداردهی آرایه همزمان با تعریف

```
;double[] nums = {1.9, 2.9, 3.4, 3.5}
```

مقداردهی آرایه پس از تعریف

```
;nums[0] = 1.9
```

```
;nums[1] = 2.9
```

```
;nums[2] = 3.4
```

```
;nums[3] = 3.5
```

- 1** Declare an int array variable. An array variable is a remote control to an array object.

```
int[] nums;
```

- 2** Create a new int array with a length of 7, and assign it to the previously-declared int [] variable nums

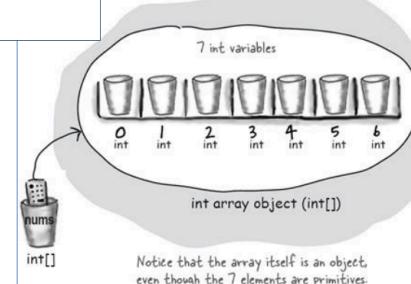
```
nums = new int[7];
```

- 3** Give each element in the array an int value.

Remember, elements in an int array are just int variables.

7 int variables

```
nums[0] = 6;  
nums[1] = 19;  
nums[2] = 44;  
nums[3] = 42;  
nums[4] = 10;  
nums[5] = 20;  
nums[6] = 1;
```



ایجاد آرایه ها

```
;arrayRefVar = new datatype[arraySize]
```

myList = new double[10] myList[0] اشاره به آخرین عنصر آرایه myList[9] اشاره به اولین عنصر آرایه

اعلان و ایجاد آرایه ها در یک گام

```
datatype[] arrayRefVar = new datatype[arraySize];
```

```
;double[] myList = new double[10]
```

```
;datatype arrayRefVar[] = new datatype[arraySize]
```

هردو تعریف فوق صحیح است، ولی تعریف اول رایج‌تر و بهتر است [10].

اندازه آرایه

یک بار که آرایه ایجاد شود، اندازه آن ثابت می‌شود. این اندازه دیگر قابل تغییر نمی‌باشد. اندازه آرایه با دستور زیر محاسبه می‌شود:

```
arrayRefVar.length
```

که در اسلاید قبل با 10 عنصر تعریف شده است، مقدار 10 را بر می‌گرداند myList برای مثال اعمال دستور بالا بر روی آرایه

```
myList.length //returns 10
```

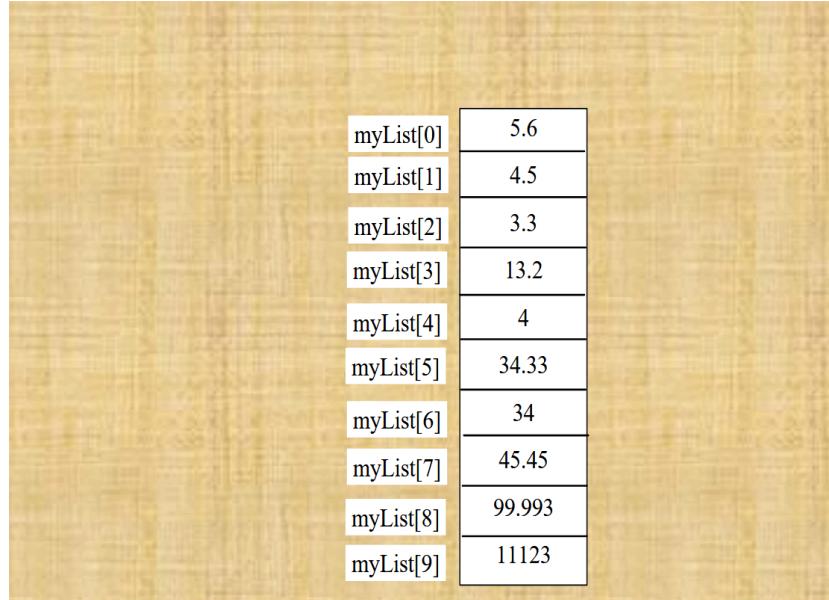
مقادیر پیش فرض

هرگاه آرایه ای ایجاد شود، به عناصر آن مقداراولیه پیش فرض داده می شود:

- صفر برای انواع داده اصلی عددی
- '\u0000' برای نوع کاراکتری
- boolean برای نوع false

متغیرهای اندیس

- عناصر آرایه از طریق اندیس قابل دسترسی هستند.
- هستند (0-based) اندیس های آرایه مبتنی بر 0.
- می باشد **1 - arrayRefVar.length** اولین خانه آنها دارای اندیس 0 و خانه آخر دارای اندیس.
- و اندیس ها از 0 تا 9 هستند **double** را نشان می دهد که 10 مقدار نوع **myList** مثال نشان داده شده در شکل، آرایه.



| | |
|-----------|--------|
| myList[0] | 5.6 |
| myList[1] | 4.5 |
| myList[2] | 3.3 |
| myList[3] | 13.2 |
| myList[4] | 4 |
| myList[5] | 34.33 |
| myList[6] | 34 |
| myList[7] | 45.45 |
| myList[8] | 99.993 |
| myList[9] | 11123 |

- دسترسی به هر عنصر آرایه که به آن متغیر اندیس دار گفته می شود، با دستور زیر انجام می شود

```
;arrayRefVar[index]
```

استفاده از عناصر آرایه (متغیرهای اندیس دار)

- پس از اینکه یک آرایه ایجاد شد، با یک متغیر اندیس دار (هریک از خانه‌های آرایه) مانند یک متغیر عادی رفتار می شود

برای مثال، کد زیر مقدار دو عنصر آرایه (با اندیس 0) و (با اندیس 1) را جمع کرده و در عنصر دیگر (با اندیس 2) قرار می‌دهد

```
;myList[2] = myList[0] + myList[1]
```

مقداردهی اولیه آرایه

- اعلان، ایجاد و مقداردهی اولیه آرایه می تواند در یک گام انجام شود.

```
;double[] myList = {1.9, 2.9, 3.4, 3.5}
```

تمامی این اعمال در یک دستور انجام می شود.

دستور بالا معادل دستورات زیر است:

```
;myList[0] = 1.9  
;myList[1] = 2.9  
;myList[2] = 3.4  
;myList[3] = 3.5
```

استفاده از یک دستور برای مراحل اعلان، ایجاد و مقداردهی آرایه در بیشتر مواقع توصیه می شود.

توزيع این مراحل در چند دستور باید با دقت انجام شود. دستور زیر نادرست است

```
;double[] myList  
;myList = {1.9, 2.9, 3.4, 3.5}
```

برنامه حاوی آرایه

در ادامه برنامه ای حاوی آرایه را به همراه ردگیری آن بررسی می کنیم.

```
In [ ]: public class TestArray {  
    public static void main(String[] args) {  
        int[] values = new int[5];  
        for(int i = 1; i < 5; i++) {  
            values[i] = i + values[i - 1];  
        }  
        values[0] = values[1] + values[4];  
    }  
}
```

ردگیری برنامه حاوی آرایه

Declare array variable **values**, create an array, and assign its reference to **values**.

After the array is created

| | |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |

i (=1) is less than 5

$$\text{values}[1] = 1 + \text{values}[0] = 1 + 0 = 1$$

After the first iteration

| | |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |

After `i++`, i becomes 2.

i (=2) is less than 5

$$\text{values}[2] = 2 + \text{values}[1] = 2 + 1 = 3$$

After the second iteration

| | |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 3 |
| 3 | 0 |
| 4 | 0 |

After $i++$, i becomes 3.

i ($=3$) is less than 5

$$\text{values}[3] = 3 + \text{values}[2] = 3 + 3 = \mathbf{6}$$

After the third iteration

| | |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 3 |
| 3 | 6 |
| 4 | 0 |

After $i++$, i becomes 4.

i ($=4$) is less than 5

$$\text{values}[4] = 4 + \text{values}[3] = 4 + 6 = \mathbf{10}$$

| After the fourth iteration | |
|----------------------------|----|
| 0 | 0 |
| 1 | 1 |
| 2 | 3 |
| 3 | 6 |
| 4 | 10 |

After $i++$, i becomes 5.

$i (=5) < 5$ is false \rightarrow Exit the loop.

$$\text{values}[0] = \text{values}[1] + \text{values}[4] = 1 + 10 = \mathbf{11}$$

| After the fourth iteration | |
|----------------------------|----|
| 0 | 11 |
| 1 | 1 |
| 2 | 3 |
| 3 | 6 |
| 4 | 10 |

پردازش آرایه ها

در مثال های نشان داده شده در اسلاید های بعدی، روش های مختلف پردازش آرایه ها بیان شده است.

مقداردهی اولیه آرایه ها با مقادیر ورودی و چاپ آرایه ها

```
In [ ]: public class ArrayInitializing {
    public static void main(String[] args) {
        double[] myList = new double[5];
        java.util.Scanner input = new java.util.Scanner(System.in);
        System.out.print("Enter " + myList.length + " values: ");
        for (int i = 0; i < myList.length; i++)
            myList[i] = input.nextDouble();

        for (int i = 0; i < myList.length; i++)
            System.out.print(myList[i] + " ");
    }
}
```

مقداردهی اولیه آرایه ها با مقادیر تصادفی و چاپ آرایه ها

```
In [ ]: import java.util.Arrays;

public class RandomArray {
    public static void main(String[] args) {
        double[] myList = new double[5];
        for (int i = 0; i < myList.length; i++)
            myList[i] = Math.random() * 100;

        System.out.println(Arrays.toString(myList));
    }
}
```

جمع کردن همه عناصر آرایه

```
In [ ]: public class TotalArray {
    public static void main(String[] args) {
        double[] myList = new double[5];
        java.util.Scanner input = new java.util.Scanner(System.in);
        System.out.print("Enter " + myList.length + " values: ");
```

```
        for (int i = 0; i < myList.length; i++)
            myList[i] = input.nextDouble();

        double total = 0;
        for(int i = 0;i < myList.length; i++)
            total += myList[i];
        System.out.println(total);
    }
}
```

یافتن بزرگترین مقدار

```
In [ ]: public class MaxArray {
    public static void main(String[] args) {
        double[] myList = new double[5];
        java.util.Scanner input = new java.util.Scanner(System.in);
        System.out.print("Enter " + myList.length + " values: ");
        for (int i = 0; i < myList.length; i++)
            myList[i] = input.nextDouble();

        double max = myList[0];
        for(int i = 1;i < myList.length; i++)
            if(myList[i] > max) max = myList[i];
        System.out.println(max);
    }
}
```

شافلینگ تصادفی

```
In [ ]: public class ShuffleRandomArray {
    public static void main(String[] args) {
        double[] myList = new double[5];
        java.util.Scanner input = new java.util.Scanner(System.in);
        System.out.print("Enter " + myList.length + " values: ");
        for (int i = 0; i < myList.length; i++)
            myList[i] = input.nextDouble();

        for (int i = 0; i < myList.length; i++) {
            // Generate an index j randomly
            int index = (int)(Math.random()* myList.length);
            // Swap myList[i] with myList[j]
            double temp = myList[i];
            myList[i] = myList[index];
            myList[index] = temp;
        }
        for (int i = 0; i < myList.length; i++)
            System.out.print(myList[i] + " ");
    }
}
```

شیفت چرخشی عناصر-شیفت چپ

فرض کنید خانه صفر آرایه، سمپ چپ لیست قرار گرفته است.

```
In [ ]: public class ShiftArray {
    public static void main(String[] args) {
        double[] myList = new double[5];
        java.util.Scanner input = new java.util.Scanner(System.in);
        System.out.print("Enter " + myList.length + " values: ");
        for (int i = 0; i < myList.length; i++)
            myList[i] = input.nextDouble();

        double temp = myList[0]; // Retain the first element
        // Shift elements left
        for (int i = 1; i < myList.length; i++) {
            myList[i - 1] = myList[i];
        }
        // Move the first element to fill in the last position
        myList[myList.length - 1] = temp;

        for(int i = 0; i < myList.length; i++)
            System.out.print(myList[i] + " ");
    }
}
```

حلقه for توسعه یافته

- معرفی کرده که به شما امکان می‌دهد، یک آرایه کامل را بدون استفاده از اندیس آرایه، خانه به خانه پیمایش کنید for یک ساختار JDK.
- را نمایش می‌دهد myList برای مثال، کد زیر تمامی عناصر در آرایه

```
for (double value : myList)
    ;System.out.println(value)
```

- در حالت کلی قاعده نحوی به فرم زیر است

```
} for (elementType value : arrayRefVar)
Process the value //
{
```

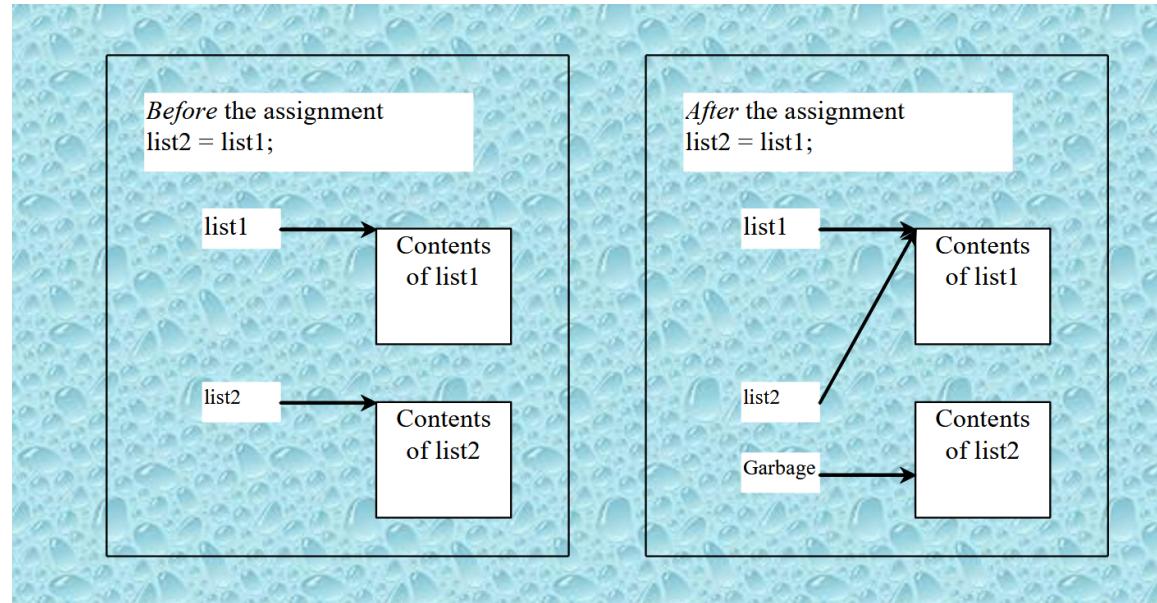
- اگر بخواهید عناصر را به ترتیب دیگری پیمایش کنید یا محتوای عناصر را به شکل دیگری تغییر دهید، باید از اندیس استفاده کنید

کپی آرایه ها

- اغلب، در یک برنامه می خواهیم محتوای یک آرایه را در یک آرایه دیگر کپی کنیم

در این موقع ممکن است `az = list1` به صورت زیر استفاده کنید، که نتیجه ای خلاف انتظار شما خواهد داشت

```
;list2 = list1
```



کپی آرایه با استفاده از حلقه

```
In [ ]: public class CopyArray {
    public static void main(String[] args) {
        int[] sourceArray = {2, 3, 1, 5, 10};
        int[] targetArray = new int[sourceArray.length];
        for (int i = 0; i < sourceArray.length; i++) {
            targetArray[i] = sourceArray[i];
            System.out.print(targetArray[i] + " ");
        }
    }
}
```

کپی آرایه با استفاده از عملیات arraycopy

```
arraycopy(sourceArray, src_pos, targetArray, tar_pos, length);
```

```
In [ ]: public class ArrayCopy {
    public static void main(String[] args) {
        int[] sourceArray = {2, 3, 1, 5, 10};
        int[] targetArray = new int[sourceArray.length];
        System.arraycopy(sourceArray, 0, targetArray, 0, sourceArray.length);
        for(int i = 0; i < targetArray.length; i++)
            System.out.print(targetArray[i] + " ");
    }
}
```

ارسال آرایه ها به متدها

```
In [ ]: public class ParameterPassing {
    public static void main(String[] args) {
        int[] list = {3, 1, 2, 6, 4, 2};
        // Invoke the method
    }
}
```

```
    printArray(list);
    printArray(new int[]{3, 1, 2, 6, 4, 2}); //Anonymous array
}
public static void printArray(int[] array) {
    for (int i = 0; i < array.length; i++) {
        System.out.print(array[i] + " ");
    }
    System.out.println();
}
}
```

آرایه بی نام (Anonymous

- دستور

```
;printArray(new int[]{3, 1, 2, 6, 4, 2})
```

با قاعده نحوی زیر یک آرایه ایجاد می کند:

```
;new dataType[]{literal0, literal1, ..., literalk}
```

در اینجا یک متغیر ارجاعی دارای نام برای آرایه وجود ندارد. به چنین آرایه ای، آرایه بی نام گفته می شود.

ارسال با مقدار

- جاوا از سازوکار ارسال با مقدار برای ارسال پارامترها به یک متده استفاده می کند. تفاوت های مهمی بین ارسال مقدار متغیرهای انواع اصلی و ارسال آرایه ها وجود دارد.

برای پارامتری از نوع اولیه، مقداری که ارسال می شود تنها در داخل متده ممکن است تغییر داده شود، اما تاثیری بر متغیری که مقدار آن به متده ارسال شده است، ندارد.

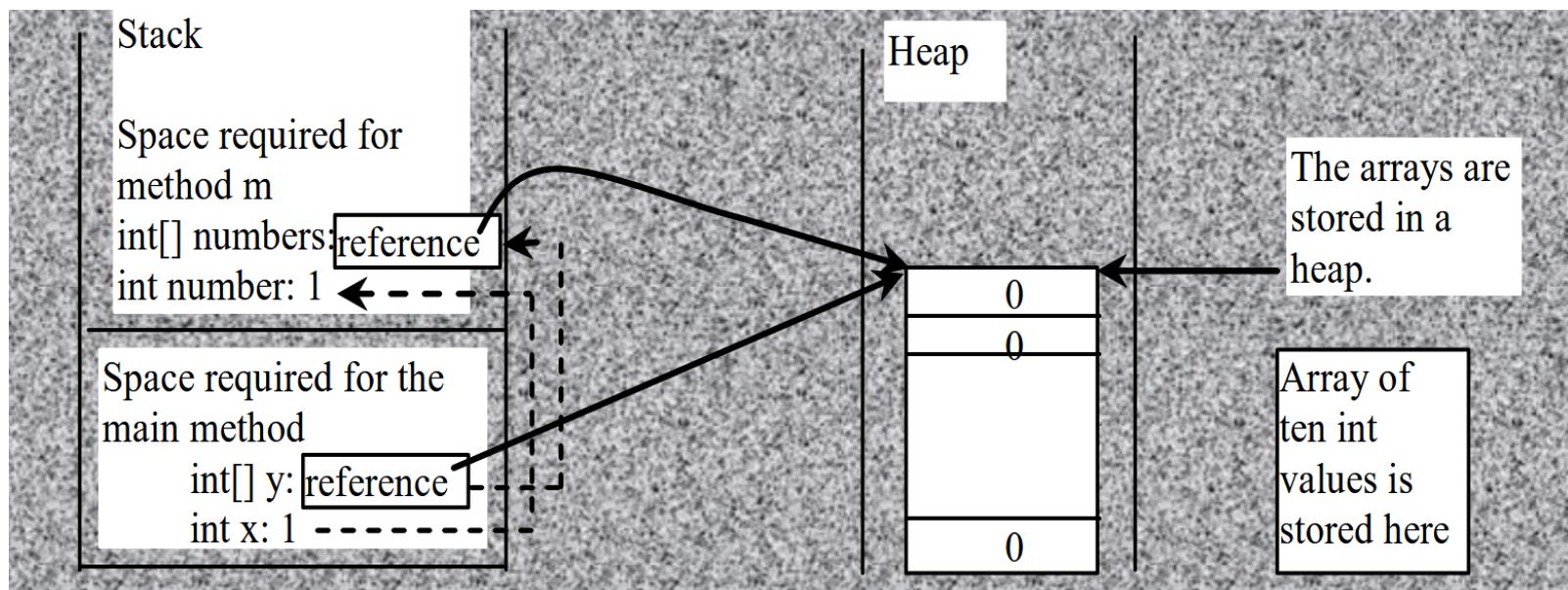
برای پارامتری از نوع آرایه، مقدار پارامتر ارسال شده، ارجاعی به یک آرایه است (یک کپی از ریموت کنترلی که به آرایه اشاره می کند، به متده ارسال می شود). بنابراین هر تغییری برروی آرایه در داخل متده، بر آرایه اصلی که به متده ارسال شده، تاثیر خواهد گذاشت.

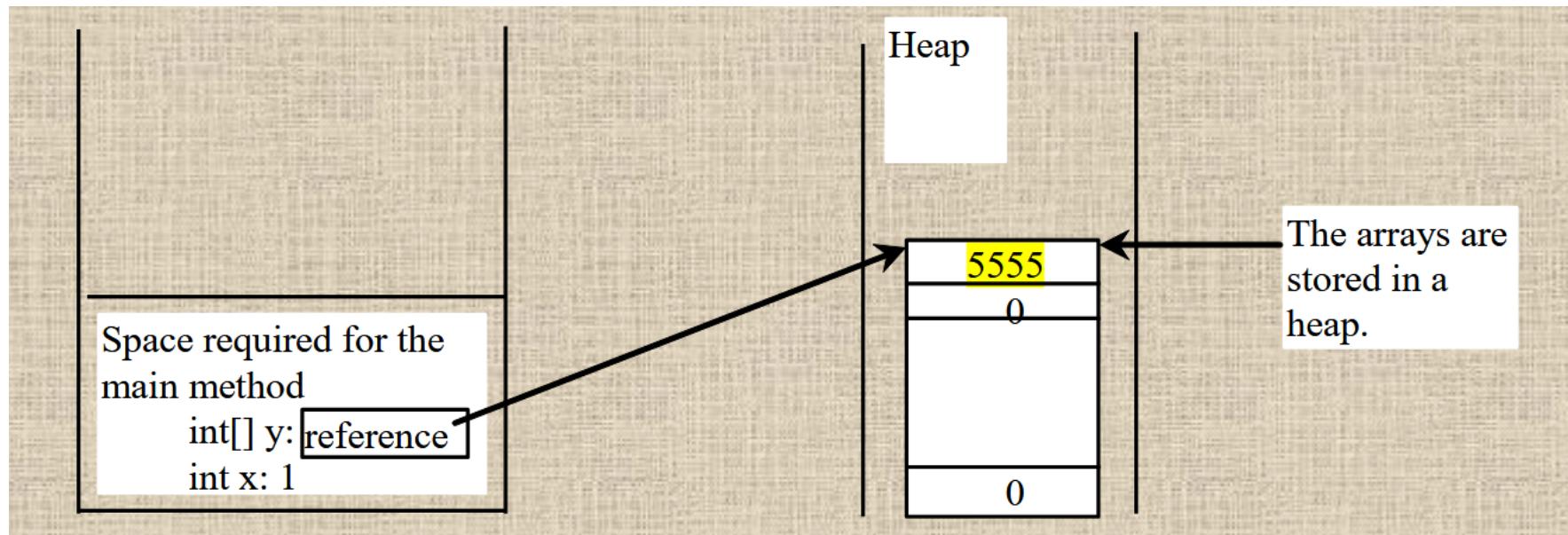
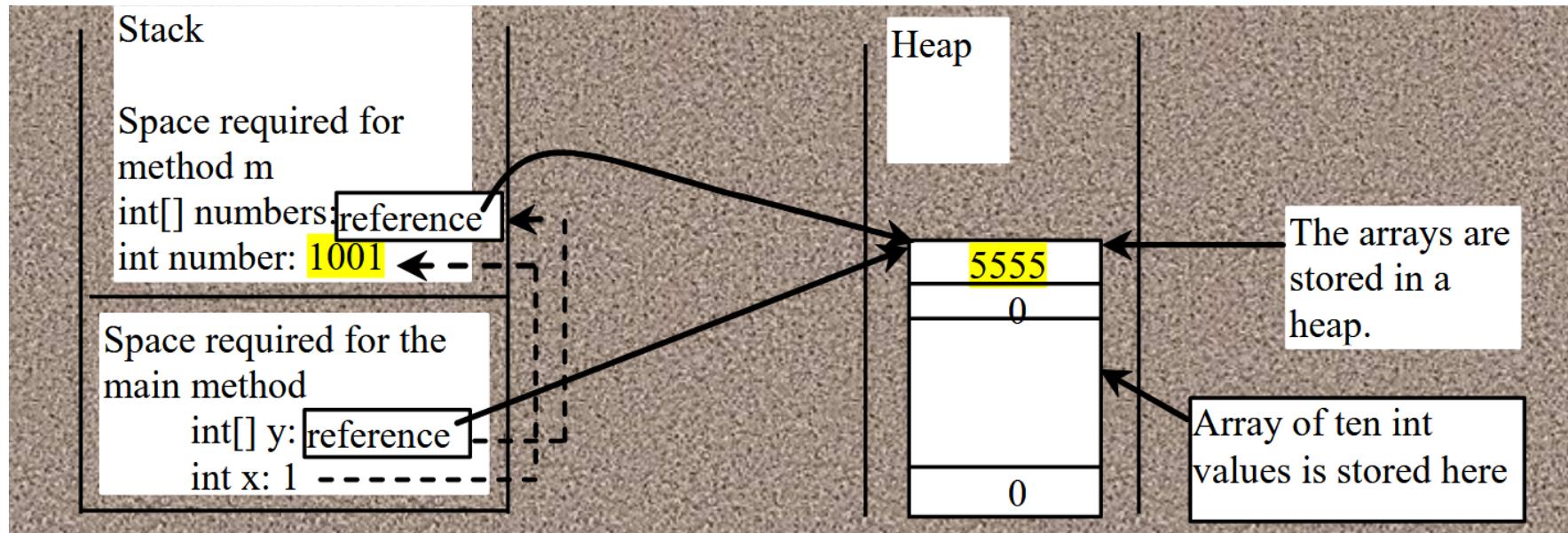
در ادامه مثالی ساده از این بخش می بینیم.

```
In [ ]: public class Test {  
    public static void main(String[] args) {  
        int x = 1; // x represents an int value  
        int[] y = new int[10]; //y represents an array of int values  
        m(x, y); // Invoke m with arguments x and y  
        System.out.println("x is " + x);  
        System.out.println("y[0] is " + y[0]);  
    }  
    public static void m(int number, int[] numbers) {  
        number = 1001; // Assign a new value to number  
        numbers[0] = 5555; // Assign a new value to numbers[0]  
    }  
}
```

پشته فراخوانی مثال قبل

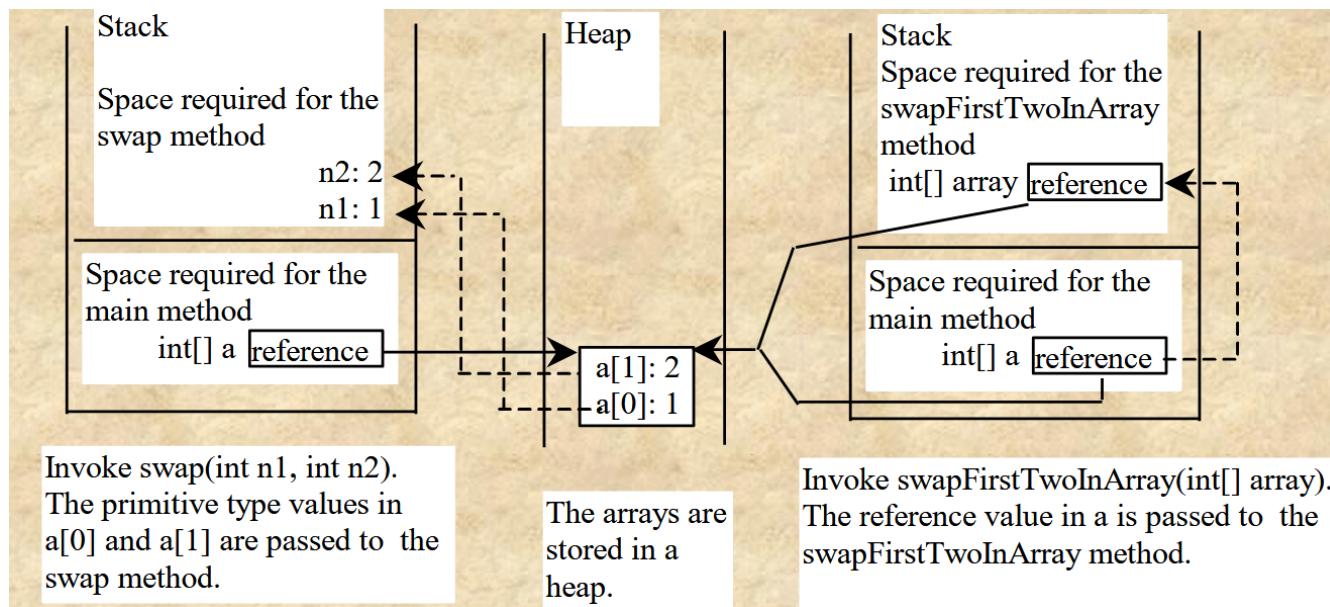
به همان آرایه numbers، حاوی مقدار ارجاعی به آرایه است y ارسال می شوند. چون numbers و number در هنگام فراخوانی m(x, y) ترتیب به y, x مقادیر، به آن ارجاع دارد y ای ارجاع می کند که





جابجایی دو عنصر آرایه

را بررسی می کنیم swap و reference swap دو روش



```
In [ ]: public class SwapArray {  
    public static void main(String[] args) {  
        int[] a = {1, 2};  
  
        // Primitive swap  
        int n1 = a[0];  
        int n2 = a[1];  
        swap(n1, n2);  
    }  
    void swap(int n1, int n2) {  
        int temp = n1;  
        n1 = n2;  
        n2 = temp;  
    }  
}
```

```

System.out.println("After swap(int n1, int n2): " + a[0] + ", " + a[1]);
// Output: 1, 2 → unchanged

// Reference swap
swapFirstTwoInArray(a);
System.out.println("After swapFirstTwoInArray(a): " + a[0] + ", " + a[1]);
// Output: 2, 1 → array is modified
}

public static void swap(int n1, int n2) {
    int temp = n1;
    n1 = n2;
    n2 = temp;
}

public static void swapFirstTwoInArray(int[] array) {
    int temp = array[0];
    array[0] = array[1];
    array[1] = temp;
}
}

```

متد معکوس آرایه

در ادامه کدی حاوی متد معکوس آرایه را به همراه ردگیری آن بررسی می کنیم.

```
In [ ]: public class ReverseArray {
    public static void main(String[] args) {
        int[] list1 = {1, 2, 3, 4, 5, 6};
        int[] list2 = reverse(list1);
        for(int i = 0; i < list2.length; i++)
            System.out.print(list2[i] + " ");
    }
}
```

```
        System.out.print(list2[i] + " ");
    }
    public static int[] reverse(int[] list) {
        int[] result = new int[list.length];
        for (int i = 0, j = result.length - 1;
             i < list.length; i++, j--) {
            result[j] = list[i];
        }
        return result;
    }
}
```

ردگیری متد معکوس آرایه

Declare result and create array.

list

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|

result

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|

i (=0) is less than 6 and j = 5

```
result[5] = list[0] = 1
```

list

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|

result

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 |
|---|---|---|---|---|---|

After $i++$ and $j--$, i becomes 1 and j becomes 4.

i (=1) is less than 6 and $j = 4$

```
result[4] = list[1] = 2
```

list

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|

result

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 2 | 1 |
|---|---|---|---|---|---|

After $i++$ and $j--$, i becomes 2 and j becomes 3.

i (=2) is less than 6 and $j = 3$

$\text{result}[3] = \text{list}[2] = 3$

| | | | | | | |
|--------|---|---|---|---|---|---|
| list | 1 | 2 | 3 | 4 | 5 | 6 |
| result | 0 | 0 | 0 | 3 | 2 | 1 |

After $i++$ and $j--$, i becomes 3 and j becomes 2.

i (=3) is less than 6 and $j = 2$

$\text{result}[2] = \text{list}[3] = 4$

| | | | | | | | |
|--------|---|---|---|---|---|---|---|
| list | <table border="1"><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table> | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 2 | 3 | 4 | 5 | 6 | | |
| result | <table border="1"><tr><td>0</td><td>0</td><td>4</td><td>3</td><td>2</td><td>1</td></tr></table> | 0 | 0 | 4 | 3 | 2 | 1 |
| 0 | 0 | 4 | 3 | 2 | 1 | | |

After $i++$ and $j--$, i becomes 4 and j becomes 1.

i ($=4$) is less than 6 and $j = 1$

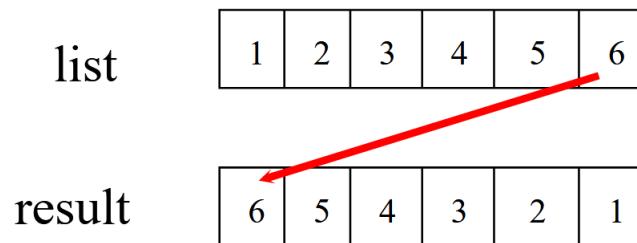
$result[1] = list[4] = 5$

| | | | | | | | |
|--------|---|---|---|---|---|---|---|
| list | <table border="1"><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table> | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 2 | 3 | 4 | 5 | 6 | | |
| result | <table border="1"><tr><td>0</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td></tr></table> | 0 | 5 | 4 | 3 | 2 | 1 |
| 0 | 5 | 4 | 3 | 2 | 1 | | |

After $i++$ and $j--$, i becomes 5 and j becomes 0.

i ($=5$) is less than 6 and $j = 0$

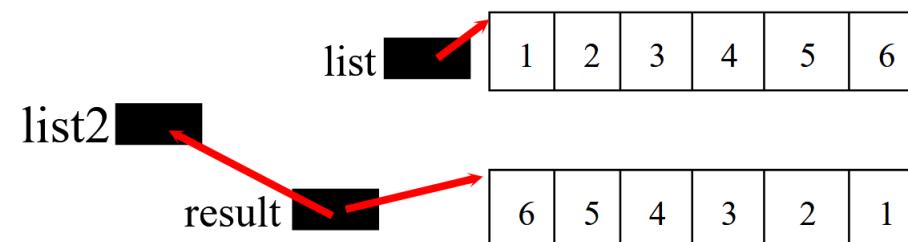
`result[0] = list[5] = 6`



After `i++` and `j--`, `i` becomes 6 and `j` becomes -1.

`i` ($=6$) < 6 is false \rightarrow Exit the loop.

Return result



مسئله: شمارش تعداد مشاهدات یک حرف

تعداد مشاهده هر حرف در آرایه دلخواه را شمارش کنید

```


    /**
     * Count the occurrences of each letter
     */
    public static int[] countLetters(char[] chars) {
        // Declare and create an array of 26 int
        int[] counts = new int[26];
        // For each lowercase letter in the array, count it
        for (int i = 0; i < chars.length; i++) {
            counts[chars[i] - 'a']++;
        }
        return counts;
    }
    /**
     * Display counts
     */
    public static void displayCounts(int[] counts) {
        for (int i = 0; i < counts.length; i++) {
            if ((i + 1) % 10 == 0)
                System.out.println(counts[i] + " " + (char) (i + 'a'));
            else
                System.out.print(counts[i] + " " + (char) (i + 'a') + " ");
        }
    }
}


```

خودمون رو بسنجیم

این بخش برای این طراحی شده که در پایان مطالعه این اسلاید، بتونی خودت رو محک بزنی و ببینی آیا مفاهیم رو به خوبی یاد گرفتی یا نه. سوالات زیر رو مرور کن و سعی کن بدون نگاه کردن به متن درس، به اون ها پاسخ بدی.

- متدى بنويسيد که عناصر يك آرایه از نوع عدد صحیح را مرتب کند
- برنامه ای بنويسید که در آرایه دلخواه (مثلا دارای 6 عنصر)، عنصر جدیدی را (مثلا عدد 35) در آرایه اضافه کند و آرایه جدید را چاپ کند

- متدی بنویسید که عناصر تکراری یک آرایه را پیدا و چاپ کند.
 - متدی بنویسید که عناصر مشترک دو آرایه را پیدا و چاپ کند.
-

پایان

در صورت هرگونه سوال یا پیشنهاد میتوانید با من در
(: ارتباط باشید
telegram: @nana_f83