Unit 6: Arrays
Introduction to Arrays

Adapted from:
1) Building Java Programs: A Back to Basics Approach by Stuart Reges and Marty Stepp
2) Runestone CSAwesome Curriculum

https://longbaonguyen.github.io
Can we solve this problem?

• Consider the following program (input underlined):

How many days' temperatures? 7
Day 1's high temp: 45
Day 2's high temp: 44
Day 3's high temp: 39
Day 4's high temp: 48
Day 5's high temp: 37
Day 6's high temp: 46
Day 7's high temp: 53
Average temp = 44.6
4 days were above average.

Do we want to store these in separate integer variables?
What if the user want to enter 1000 temperatures? (temp1, temp2,..temp1000?)
Arrays

- **array**: object that stores many values of the same type.
  - **value**: One value in an array.
  - **index**: A 0-based integer to access an element from an array.

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>12</td>
<td>49</td>
<td>-2</td>
<td>26</td>
<td>5</td>
<td>17</td>
<td>-6</td>
<td>84</td>
<td>72</td>
<td>3</td>
</tr>
</tbody>
</table>

- **index 0**: value is 12
- **index 4**: value is 5
- **index 9**: value is 3
Array declaration

\[
\text{type}[] \ \text{name} \ = \ \text{new} \ \text{type}[\text{length}] ;
\]

- Example:

\[
\text{int}[] \ \text{numbers} \ = \ \text{new} \ \text{int}[10] ;
\]

Note: The size of an array is established at the time of creation and cannot be changed. Array type can be primitive such as int and boolean or object reference type such as String or Point.

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
• The length can be any integer expression.

```java
int x = 2 * 3 + 1;
int[] data = new int[x % 5 + 2];
```

• Each element initially gets a "zero-equivalent" value.

<table>
<thead>
<tr>
<th>Type</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>0</td>
</tr>
<tr>
<td>double</td>
<td>0.0</td>
</tr>
<tr>
<td>boolean</td>
<td>false</td>
</tr>
<tr>
<td>String</td>
<td>null</td>
</tr>
<tr>
<td>or other object</td>
<td>(means, &quot;no object&quot;)</td>
</tr>
</tbody>
</table>
Accessing elements

`name[index]`  // access

`name[index] = value;`  // modify

Example:

```java
code
numbers[0] = 27;
numbers[3] = -6;

System.out.println(numbers[0]);  
if (numbers[3] < 0) {
    System.out.println("Element 3 is negative.");
}
```

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>27</td>
<td>0</td>
<td>0</td>
<td>-6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Arrays of other types

double[] results = new double[5];
results[2] = 3.4;
results[4] = -0.5;

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>0.0</td>
<td>0.0</td>
<td>3.4</td>
<td>0.0</td>
<td>-0.5</td>
</tr>
</tbody>
</table>

boolean[] tests = new boolean[6];
tests[3] = true;

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
Arrays of other types

String[] words = new String[5];
words[1] = "hi";
words[3] = "hello";
Arrays of other types

Point[] pts = new Point[5];
pts[1] = new Point(2, 3);
pts[4] = new Point();

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>null</td>
<td>null</td>
<td>null</td>
<td>null</td>
<td></td>
</tr>
</tbody>
</table>

Point
x = 2
y = 3

Point
x = 0
y = 0

Note that each element of the pts array store the reference to a Point object.
Out-of-bounds

• Legal indexes: between 0 and the *array's length - 1*.  
  – Reading or writing any index outside this range will throw an `ArrayIndexOutOfBoundsException`.

• Example:

```java
int[] data = new int[10];
System.out.println(data[0]);       // okay
System.out.println(data[9]);       // okay
System.out.println(data[-1]);      // exception
System.out.println(data[10]);      // exception
```

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Accessing array elements

```java
int[] numbers = new int[8];
numbers[1] = 3;
numbers[4] = 99;
numbers[6] = 2;

int x = numbers[1];
numbers[x] = 42;
numbers[numbers[6]] = 11; // use numbers[6] as index
```

<table>
<thead>
<tr>
<th>index</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>42</td>
</tr>
<tr>
<td>3</td>
<td>99</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

x

```

```
• It is common to use `for` loops to access array elements. This is called **traversing the array**.

```java
for (int i = 0; i < 8; i++) {
    System.out.print(numbers[i] + " ");
}
System.out.println(); // output: 0 3 11 42 99 0 2 0
```

• Sometimes we assign each element a value in a loop.

```java
for (int i = 0; i < 8; i++) {
    numbers[i] = 2 * i;
}
```

```
index 0 1 2 3 4 5 6 7
value 0 2 4 6 8 10 12 14
```
• An array's `length` field stores its number of elements.

```java
for (int i = 0; i < numbers.length; i++) {
    System.out.print(numbers[i] + " ");
}
// output: 0 2 4 6 8 10 12 14
```

– It does not use parentheses like a String's `.length()`.

• What expressions refer to:
  – The last element of any array? `numbers.length - 1`
  – The middle element? `numbers.length/2`, if length is even, this element is the first element of the second half of the array.
Arrays and while loops

While loop can also be used to traverse the array. The following compute the sum of an array using a while loop and a for loop. Note the difference. In both cases, this requires the elements of the array to be accessed by the indices(i).

**While Loop:**

```java
int sum = 0;
int i = 0;
while(i < numbers.length){
    sum += numbers[i];
    i++;
}
```

**For Loop:**

```java
int sum = 0;
for(int i = 0; i < numbers.length; i++){
    sum += numbers[i];
}
```

Typically, traversing an array is best done with a for loop.
Quick array initialization

Arrays can be used created quickly using *initializer lists*.

```
type[] name = {value, value, ... value};
```

- Example:
  ```
  int[] numbers = {12, 49, -2, 26, 5, 17, -6};
  ```

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>12</td>
<td>49</td>
<td>-2</td>
<td>26</td>
<td>5</td>
<td>17</td>
<td>-6</td>
</tr>
</tbody>
</table>

- Useful when you know what the array's elements will be
- The compiler figures out the size by counting the values
"Array mystery" problem

What element values are stored in the following array?

```java
int[] a = {1, 7, 5, 6, 4, 14, 11};
for (int i = 0; i < a.length - 1; i++) {
    if (a[i] > a[i + 1]) {
        a[i + 1] = a[i + 1] * 2;
    }
}
```

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>1</td>
<td>7</td>
<td>10</td>
<td>12</td>
<td>8</td>
<td>14</td>
<td>22</td>
</tr>
</tbody>
</table>
Limitations of arrays

- You cannot resize an existing array:

```java
int[] a = new int[4];
a.length = 10;  // error
```

- You cannot compare arrays with `==` or `equals`:

```java
int[] a1 = {42, -7, 1, 15};
int[] a2 = {42, -7, 1, 15};
if (a1 == a2) {  ...  }  // false!
if (a1.equals(a2)) {  ...  }  // false!
```

- An array does not know how to print itself:

```java
int[] a1 = {42, -7, 1, 15};
System.out.println(a1);  // [I@98f8c4]
```
public static void main(String[] args) {
    int[] a = {0, 14, 4, 6, 8};
    System.out.println(a);
}

Output: I@674f1c67 6. (14 pts)

Prints out the address not the contents of a.

Arrays.toString accepts an array as a parameter and returns a String representation of its elements.

- Must import java.util.*;
- Arrays is one of the classes in the util’s package.
import java.util.*;

public class Example
public static void main(String[] args) {
    int[] a = {0, 14, 4, 6, 8};
    System.out.println("a is "+Arrays.toString(a));
}

Output:
a is [0, 14, 4, 6, 8]
Arrays

String[] a = {"hip", "hip"};

// hip hip arrays!
Why did the programmer quit his job?

Because he didn’t get arrays.

He didn’t get arrays and he didn’t get a raise.
Write the method `average` which accepts an int array and returns the average of the values.

Write the method `countAboveAve` which accepts an int array and returns the number of values that are above the average. You must call `average`.

Write the method `largest` which accepts an int array and returns the largest value of the array.

Write the method `indexOfsmallest` which accepts an int array and returns the index of the smallest value. If there are multiple smallest values, return the index of the first one.
Also write the main method with an array and check to make sure your methods work!

```java
public static double average(int[] array){}
public static int countAboveAve(int[] array){}
public static int largest(int[] array){}
public static int indexOfsmallest(int[] array){}
```
This lab is on Processing.

Write the Ball class with attributes center_x, center_y, change_x, change_y, radius(floats) and a color.

Methods:
update()
display()

Make it bounce on the screen. We have done this before but the ball was not an object.
Create an array of ball objects and make it move on the screen!

Write a method largestBall that returns the largest Ball and set it to a different color.

There's a .pde template file on my github website if you need some help.