

Mind the Index

Array creation and access



Five quizzes, five variables

PAUSE AND TRY

The user types 3. How do you print quiz 3's score?

WHAT WE CAN DO SO FAR

```
int quiz1 = 88;  
int quiz2 = 92;  
int quiz3 = 75;  
int quiz4 = 81;  
int quiz5 = 90;
```

Talk with a partner. How much code would you need for 5 quizzes? For 30? Is there a way to turn the number 3 into the name quiz3?

By the end of class you can...

- 1 Create an array with new and with an initializer list.
- 2 State the default value every element starts with.
- 3 Read and change one element using its index in square brackets.
- 4 Use the length attribute, and explain why an array's length can never change.
- 5 Predict when an index throws an `ArrayIndexOutOfBoundsException`.

A large, white, rounded square containing the number '1' in a bold, black, sans-serif font.

PART 1

One name, many values

An array stores many values of one type

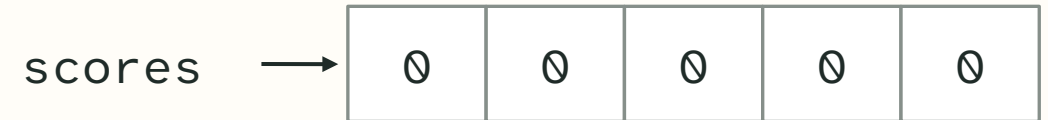


An **array** stores multiple values of the same type. Each value is an **element**. Elements can be primitive values or object references.

```
int[] scores = new int[5];
```

Piece	Meaning
<code>int[]</code>	the variable's type: array of int
<code>scores</code>	the name; it holds a reference
<code>new int[5]</code>	a new array with room for 5 ints

IN MEMORY



One object, five elements, all int.

Arrays of different element types

```
double[] prices = new double[3];  
boolean[] lightsOn = new boolean[4];  
String[] songs = new String[2];
```

Variable	Element type	Number of elements	Kind of element
prices	double	3	primitive value
lightsOn	boolean	4	primitive value
songs	String	2	reference to a String

Type on both sides must match. The size goes only after new.

A large, white, rounded square containing the number '2' in a bold, teal, sans-serif font.

2

PART 2

Index and square brackets

Square brackets and an index

temps	68	71	65	74
index	0	1	2	3



arr[index] reads or changes one element. Valid indexes run from 0 to one less than the length.

READ, THEN CHANGE

```
int[] temps = {68, 71, 65, 74};  
System.out.println(temps[0]);  
temps[2] = 66;  
System.out.println(temps[2]);
```

OUTPUT

```
68  
66
```

An index can be any int expression

```
int[] temps = {68, 71, 65, 74};  
int day = 3;  
System.out.println(temps[day]);  
System.out.println(temps[day - 1]);  
System.out.println(temps[0] + temps[1]);
```

OUTPUT

```
74  
65  
139
```

Expression	Index used	Value
temps[day]	3	74
temps[day - 1]	3 - 1 = 2	65
temps[0] + temps[1]	0 and 1	68 + 71 = 139

Trace the cost array

PAUSE AND TRY

Write the exact output.

```
double[] cost = {2.5, 4.0, 1.25};  
cost[1] = cost[0] * 2;  
cost[2] += 0.75;  
System.out.println(cost[1]);  
System.out.println(cost[2]);
```

YOUR OUTPUT

A large, bold, teal number '3' is centered within a white rounded square that has a slight tilt.

PART 3

Defaults and initializer lists

new fills every element with a default

Element type	Default value
int	0
double	0.0
boolean	false
a reference type, like String	null



When **new** creates an array, every element starts at the default for its type.

```
int[] counts = new int[3];
double[] rates = new double[2];
boolean[] lightsOn = new boolean[4];
String[] songs = new String[2];
System.out.println(counts[0]);
System.out.println(rates[1]);
System.out.println(lightsOn[3]);
System.out.println(songs[0]);
```

OUTPUT

```
0
0.0
false
null
```

An initializer list creates and fills

WORKS: LIST IN THE DECLARATION

```
String[] colors = {"red", "teal", "gold"};  
System.out.println(colors.length);  
System.out.println(colors[1]);
```

OUTPUT

```
3  
teal
```

DOES NOT COMPILE: LIST ON A LATER LINE

```
String[] colors;  
colors = {"red", "teal", "gold"};
```

javac: illegal start of expression



An **initializer list** sets the length and the values in one step, starting at index 0.

A large, bold, teal-colored number '4' is centered within a white square with rounded corners. The square is slightly tilted to the right.

4

PART 4

Length is fixed

length: an attribute, not a method

```
String[] colors = {"red", "teal", "gold"};  
System.out.println(colors.length);  
System.out.println(colors[colors.length - 1]);  
System.out.println(colors[1].length());
```

OUTPUT

```
3  
gold  
4
```

You have...	Its size	Why
an array	<code>colors.length</code>	<code>length</code> is an attribute of the array
a String	<code>"teal".length()</code>	<code>length()</code> is a String method

The last element is always at `arr.length - 1`.

"Making the array bigger"



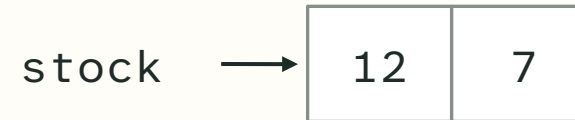
An array's length is set when it is created and **cannot be changed**. `new` makes a different array; it never stretches the old one.

```
int[] stock = {12, 7};  
stock = new int[4];  
System.out.println(stock.length);  
System.out.println(stock[0]);
```

OUTPUT

4
0

BEFORE LINE 2



AFTER LINE 2



old array, unchanged, no variable

A large, white, rounded square containing a bold, dark teal number '5'.

5

PART 5

Out of bounds, and objects

One past the end



Valid indexes are **0 through length - 1**. Any other index throws an **`ArrayIndexOutOfBoundsException`** while the program runs.

```
int[] temps = {68, 71, 65, 74};  
System.out.println(temps[3]);  
System.out.println(temps[temps.length]);  
System.out.println("finished");
```

OUTPUT

74

ERROR WHILE RUNNING (WRAPPED)

```
Exception in thread "main"  
java.lang.  
ArrayIndexOutOfBoundsException:  
Index 4 out of bounds for length 4
```

An array of objects holds references

THE CLASS

```
class Pet {  
    private String name;  
  
    public Pet(String petName) {  
        name = petName;  
    }  
  
    public String getName() {  
        return name;  
    }  
}
```

IN MAIN

```
Pet[] shelter = new Pet[3];  
shelter[0] = new Pet("Biscuit");  
System.out.println(shelter[0].getName());  
System.out.println(shelter[1]);
```

OUTPUT

```
Biscuit  
null
```

ref	null	null
0	1	2



shelter[1].getName() would throw a **NullPointerException**.

Elements are Strings

PAUSE AND TRY

Write the exact output. Which length is which?

```
String[] words = {"apple", "kiwi", "fig"};  
System.out.println(words[1].length());  
System.out.println(words.length);  
String last = words[words.length - 1];  
System.out.println(last.substring(0, 1));
```

YOUR OUTPUT

Words to know

Array

An object that stores a fixed number of values of one type.

Element

One value stored in an array.

Index

An element's position, from 0 to length - 1.

Default value

0, 0.0, false, or null: what new puts in each element.

Initializer list

Values in braces that create and fill an array at once.

length

The number of elements; an attribute, fixed at creation.

Back to the five quizzes



Rewrite the five quiz variables as one array. Then explain to your partner how to print quiz 3 and the last quiz.

SENTENCE STARTER

Quiz 3 is at index ____ because _____. The last quiz is at index _____.

Next: the Mind the Index puzzle worksheet.