Arrays

A byte size lesson in Java programming.

What is an array?

- In Java, an array is a collection of similar data types
- Here is an array to store the names of 100 people

```
String[] names = new String[100]
```

Breaking it down

- This is how we declare an array

```
dataType[] arrayName;
```

- And this is how we tell it how many elements it can hold

```
double[] arrayName;

arrayName = new double[10]
```

Let's test your understanding!

- How many numbers can I fit in the following arrays?

```
int[] scores = new int[10];
```

```
String[] surnames = new String[20];
```

- What is wrong with the following declaration?

```
int[] scores = new double[10];
```

Initializing arrays

- Sometimes we want to fill an array with data immediately. In this case the Java compiler automatically sets the size to 4

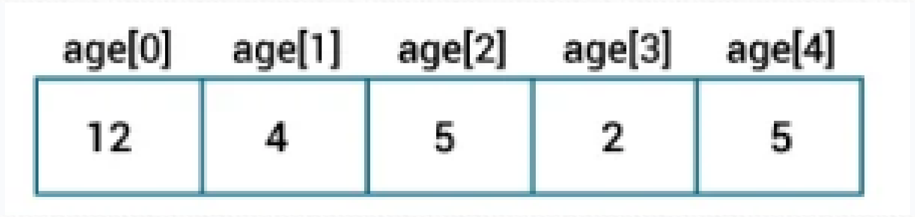
```
int[] ages = {5, 12, 8, 7};
```

Array index

- In an array, each memory location is associated with a number (or index).

```
int[] ages = {12, 4, 5, 2, 5};
```

- The index starts from 0. here are the indexes of the array declared above.



Array index

- If we want to get a particular element from an array, we can use the index

```
int[] ages = {12, 4, 5, 2, 5};

System.out.println(“First Element: ” + ages[0]);
System.out.println(“Second Element: ” + ages[1]);
```

- If the number of elements is **n**, then the last position is **n-1**. If we try and access beyond that, we will get an error!

Let’s test your understanding!

- Complete the following code

```
double[] prices = {5.20, 2.4, 37.2, 5.00};
```

```
System.out.println(“Second Element: ” + prices[   ]);
```

```
System.out.println(“          Element: ” + prices[3]);
```

- What would be the output of:

```
System.out.println(prices[4]);
```

Looping through arrays

- One of the most common uses of arrays is to loop through them and perform a function for each element in the array:

```
// create an array
int[] age = {12, 4, 5};

// loop through the array
// using for loop
System.out.println("Using for Loop:");
for(int i = 0; i < 3; i++) {
  System.out.println(age[i]);
```

Output

```
Using for Loop:
12
4
5
```

Let’s test your understanding!

- Give the output of the following

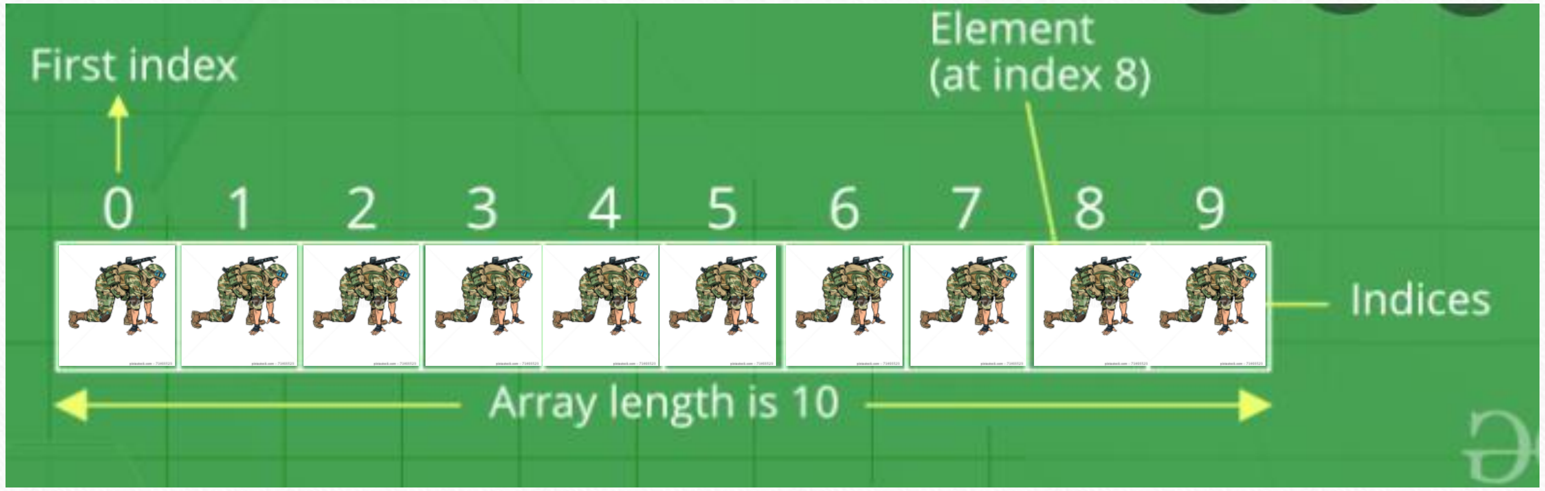
```
   // create an array
   String[] name = {“John”, “Karl”, “Max”, “Jane”};

   System.out.println("Using for Loop:");
   for(int i = 2; i < 4; i++) {
     System.out.println(name[i]);
}
```

Array of Objects

- It is perfectly possible and often necessary to create arrays that store complex types like objects.
- You will find that declaring an array of objects is not much different to creating an array of simple types.

```
Soldier[] soldiers = new Soldier[10];
```

Array of Soldier



Assigning an array value

```
// create an array
Soldier[] soldiers = new Soldier[10];

// create an object instance and store it in array
soldiers[0] = new Soldier();

// works also
Soldier john = new Soldier();
soldiers[1] = john;
```

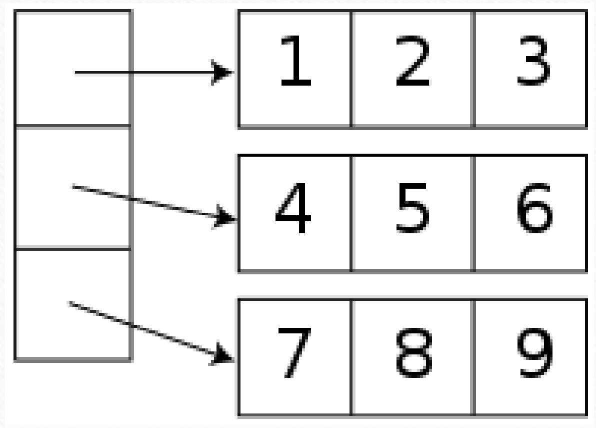
Let’s test your understanding!

- Complete the following code…

```
             accounts = new BankAccount[5];

System.out.println(“Second account: ” + accounts   .number);

System.out.println(“        account: ” + accounts[3].number);

// create and store the last account object in accounts

```

Multidimensional Arrays

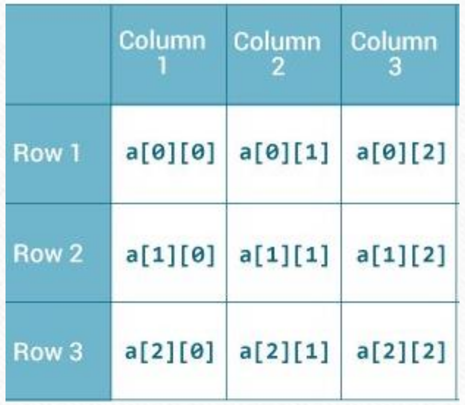
- A multidimensional array is an array of arrays. Each array element in a multidimensional array would be an array!
- We can create two dimensional arrays to organise data in a grid/table/matrix.



Initialising two dimensional arrays

- To create a 2D array we have to specify the size of each dimension. In this case, the number of rows and columns. The below can hold a maximum of 9 elements.

```
int[][] a = new int[3][3]; // 3 rows and 3 columns
```



	Column 1	Column 2	Column 3
Row 1	a[0][0]	a[0][1]	a[0][2]
Row 2	a[1][0]	a[1][1]	a[1][2]
Row 3	a[2][0]	a[2][1]	a[2][2]

Initialising two dimensional arrays

- This is an example of how we can initialise a 2D array of numbers.

```
// create an array
int[][] matrix = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
```

Let’s test your understanding!

- What is the output of the following?

```
int[][] a = {
    {1, 2, 3},
    {4, 5, 6, 7}
    {7}
};

for(int row = 0; row < 3; row++) {
    System.out.println(“Row ” + row + “ has ” + a[row].length + “ cols”);
}
```

Looping through 2D arrays

- As with 1D arrays, it is very common to loop through them and do something with each element:

```
int[][] matrix = { {1, 2, 3}, {4, 5, 6,}, {7, 8, 9} };

for(int row = 0; row < 3; row++) {
    for(int col = 0; col < 3; col++) {
        int number = matrix[row][col];
        System.out.print(number + “, ”);
    }
}
// 1, 2, 3, 4, 5, 6, 7, 8, 9
```