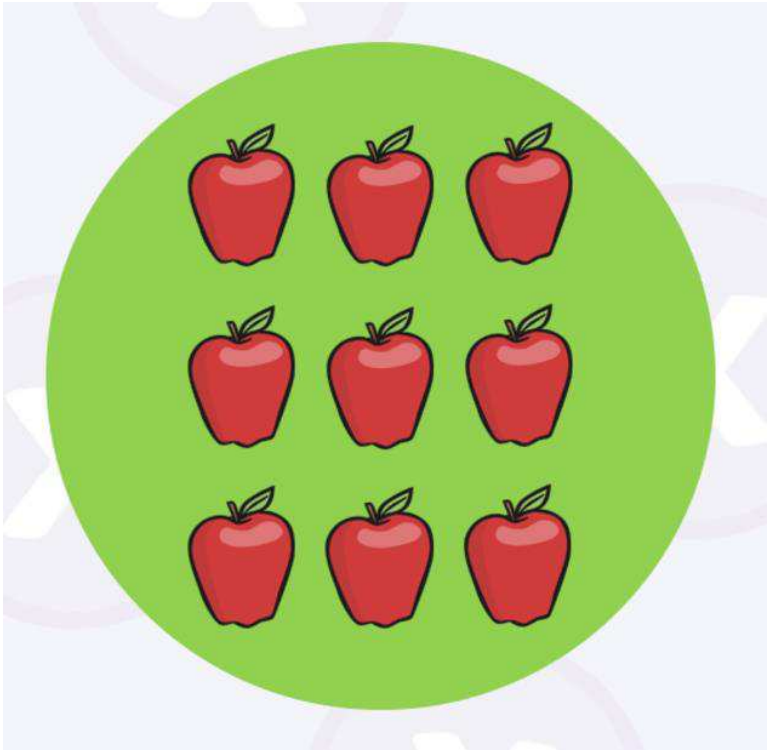
Resource Software Solution

# Session 04: Arrays



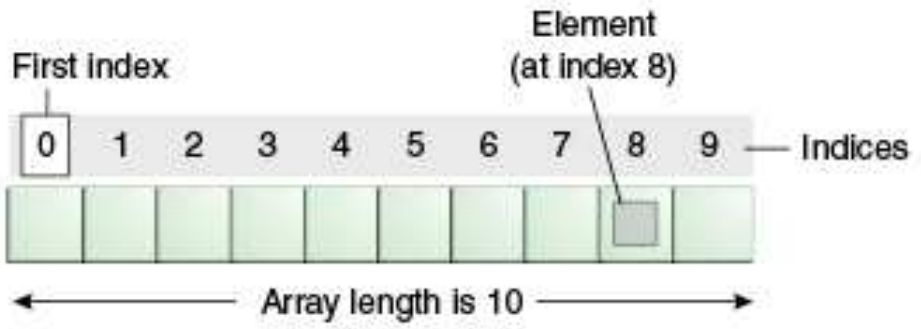
# Objectives

1. Overview
2. Types of Array
3. Array Statements

# Overview (1)

## What is an Array

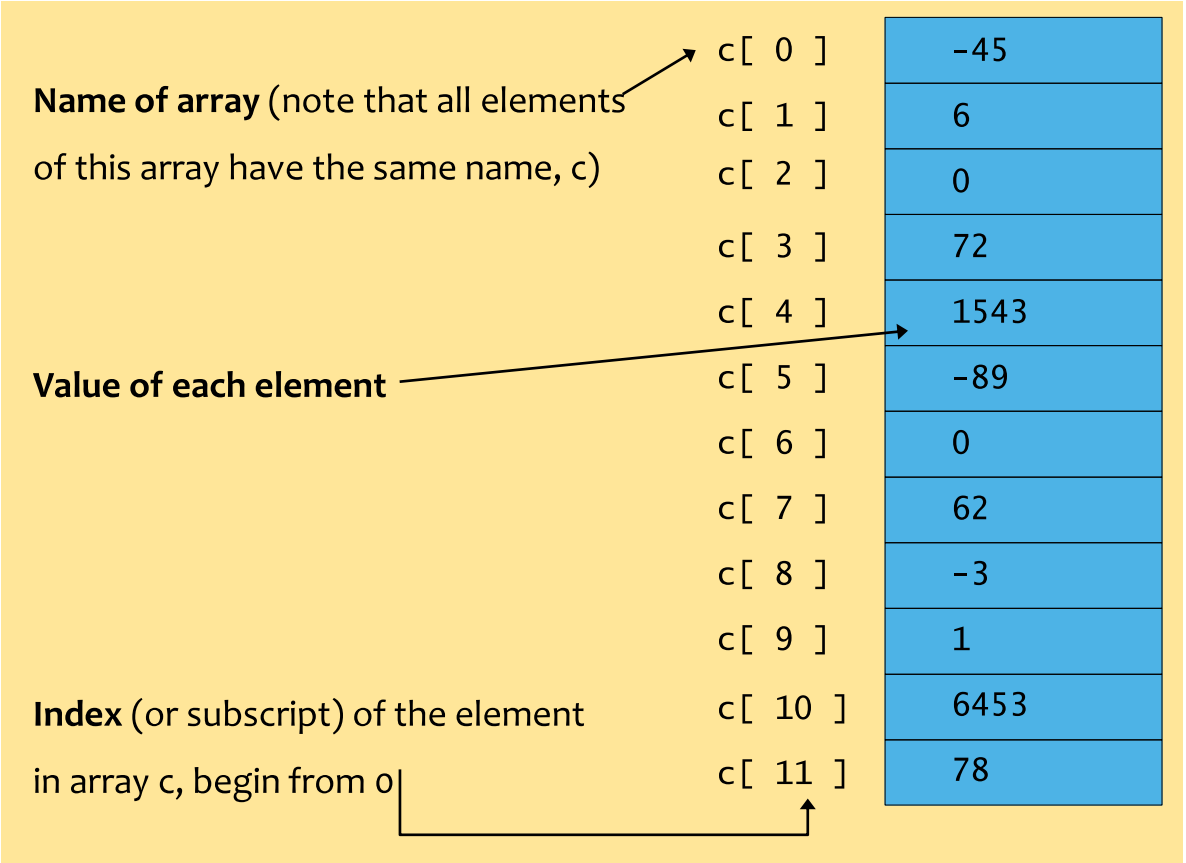
- **Java array** is an object which contains elements of a **similar data type.** The elements of an array are stored in a **contiguous memory location**
- It is a data structure where we store similar elements:
  - We can store only a fixed set of elements in a Java array
  - Array in Java is index-based, the first element of the array is stored at the **0th index,** 2nd element is stored on 1st index and so on
  - We can store **primitive values** or **objects** in an array in Java



# Overview (2)

## Types of Array

- There are two types of array
  - Single Dimensional Array
  - Multidimensional Array



Resource Software Solution

2

# Types of Array

# Single Dimensional Array (1)

## Array Declarations

- Syntax

```
datatype [] identifier = new datatype[size];

datatype [] identifier = {value1,value2,... valueN};
```

- Example

```
float [] fArray = new float [20];

int [] iArray = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };
```

# Single Dimensional Array (2)

## Examine

- **fArray, iArray** is the array *name*
- **fArray**.length accesses array's *length*
- **iArray** has 10 *elements*
  - **iArray**[0], **iArray** [1], … , **iArray** [9]
  - The *value* of **iArray** [0] is 32

```
float [] fArray = new float [20];
int [] iArray = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };
```

# Single Dimensional Array (3)

## Array Index

- Also called subscript
- Position number in **square brackets**
- Always begin from zero: **Must >= 0 and < array's length**
- Example

```
for (int counter = 0; counter < iArray.length; counter++) {
    System.out.print(iArray[counter]);
}
```

# Single Dimensional Array (4)

## ArrayIndexOutOfBoundsException

- The Java Virtual Machine (JVM) throws an **Exception** if length of the array in **negative, equal to the array size or greater than** the array size while traversing the array
- Example

```
public class TestArrayException {
  public static void main(String[] args) {
    int arr[] = { 50, 60, 70, 80 };
    for (int i = 0; i <= arr.length; i++) {
       System.out.println(arr[i]);
    }
  }
}
```

**Output:**

```
50
60
70
80
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 4 at
TestArrayException.main(TestArrayException.java:15)
```

# Two Dimensional Array (1)

## Array Declarations

- In such case, data is stored in **row** and **column** based index (also known as matrix form)
- Syntax

```
datatype [][] identifier = new datatype[size][size];
```

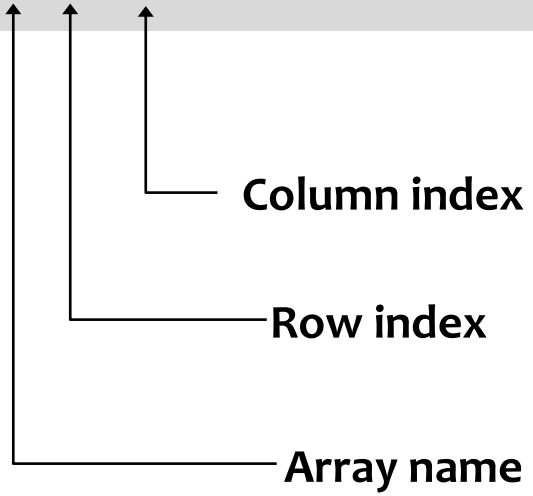
- Example

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

# Two Dimensional Array (2)

## Array structure

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



# Two Dimensional Array (3)

## Example

- Let's see the simple example to ***declare***, ***instantiate***, ***initialize*** and ***print***

```
public class Test2Dimensional {
        public static void main(String[] args) {
            // Declaring and initializing 2D array
            int arr[][] = { { 1, 2, 3 }, { 2, 4, 5 }, { 4, 4, 5 } };

            // Printing 2D array
            for (int i = 0; i < 3; i++)  {
              for (int j = 0; j < 3; j++)  {
                  System.out.print(arr[i][j] + "\t");
              }

              System.out.println();
            }
        }
}
```

**Output:**

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

# Two Dimensional Array (4)

## Jagged Array

- **Jagged Array** is an array of arrays with **different** number of columns

```
public class TestJaggedArray {
    public static void main(String[] args) {
        // Declaring a jagged array
        int[][] jagArray = new int[3][];

        jagArray[0] = new int[3];
        jagArray[1] = new int[5];
        jagArray[2] = new int[2];

        Random random = new Random(2);
        // Initializing a jagged array
        for (int i = 0; i < jagArray.length; i++) {
          for (int j = 0; j < jagArray[i].length; j++) {
            jagArray[i][j] = random.nextInt(100);
          }
        }
    }
}
```

Resource Software Solution

3

# Array Statements

# Copying a Java Array

- We can copy an array to another by the **arraycopy**() method of **System** class
- Syntax **arraycopy**(Object src, **int** srcPos,Object dest, **int** destPos, **int** length)
- Example

```
public class TestArrayCopyDemo {
  public static void main(String[] args)
    // Declaring a source array
    char[] copyFrom = {'R', '2', 'S', 'S', 'o', 'f', 't', 'w', 'a', 'r', 'e', 'A', 'c', 'a', 'd', 'e', 'm', 'y' };

    // Declaring a destination array
    char[] copyTo = new char[10];

    // Copying array using System.arraycopy() method
    System.arraycopy(copyFrom, 3, copyTo, 0, 8);

    // Printing the destination array
    System.out.println(String.valueOf(copyTo));
  }
}
```

**Output:**

Software

# Cloning an Array

- We can clone an array by the **clone**() method
- Example

```
public class TestCloneArray {
        public static void main(String[] args) {
           int arr[] = { 12, 5, 18, 8, 6 };
           System.out.println("Printing original array:");
           for (int value : arr) {
             System.out.println(value);
           }

           System.out.println("Printing clone of the array:");
           int carr[] = arr.clone();
           for (int value : carr) {
             System.out.println(value);
           }

           System.out.println(arr == carr);
         }
}
```

Resource Software Solution

# Summary

1. Single Dimensional Array
2. Two Dimensional Array/Jagged Array
3. ArrayIndexOutOfBoundsException
4. Copying a Java Array
5. Cloning an Array in Java

# Thankyou!