Arrays

COP2250: Java Programming

Kevin Pyatt, Ph.D.

State College of Florida
Pyatt Labs

Week 4

Today's Objectives

- Understand what arrays are and why we use them
- Declare and initialize arrays
- Access elements by index (zero-indexed)
- Use `.length` property
- Loop through arrays
- Apply arrays to Rock-Paper-Scissors

The Problem: Too Many Variables

What if you need 100 test scores?

```
int score1 = 90;
int score2 = 85;
int score3 = 78;
// ... 97 more lines?!
```

This doesn't scale.

- Tedious to write
- Hard to maintain
- Can't loop through individual variables

The Solution: Arrays

An array holds multiple values in one variable.

```
int[] scores = {90, 85, 78, 92, 88};
```

Think of it as a row of boxes:

90	85	78	92	88
[0]	[1]	[2]	[3]	[4]

Each box has an **index** starting at **0**.

Declaring Arrays

Two ways to create an array:

1. Declare with initial values:

```java
int[] scores = {90, 85, 78, 92, 88};
String[] names = {"Alice", "Bob", "Charlie"};
```

2. Declare empty, fill later:

```java
String[] colors = new String[3]; // 3 empty slots
colors[0] = "Red";
colors[1] = "Green";
colors[2] = "Blue";
```

Accessing Elements

Arrays are zero-indexed — first element is at index 0.

```
String[] choices = {"Scissor", "Rock", "Paper"};

System.out.println(choices[0]); // Scissor
System.out.println(choices[1]); // Rock
System.out.println(choices[2]); // Paper
```

"Scissor"	"Rock"	"Paper"
[0]	[1]	[2]

Common Mistake: Off-By-One

Arrays start at 0, not 1!

```
String[] choices = {"Scissor", "Rock", "Paper"};

// WRONG - will crash!
System.out.println(choices[3]);
// ArrayIndexOutOfBoundsException

// RIGHT - last index is length - 1
System.out.println(choices[2]); // Paper
```

Remember: For an array of size n, valid indices are 0 to $n - 1$.

Array Length

Use `.length` to get the size of an array.

```
String[] choices = {"Scissor", "Rock", "Paper"};

System.out.println(choices.length); // 3
```

Useful for:

- Looping through all elements
- Finding the last element: `array[array.length - 1]`
- Avoiding out-of-bounds errors

Note: It's `.length` (no parentheses), not `.length()`

Looping Through Arrays

Use a for loop to visit every element:

```
int[] scores = {90, 85, 78, 92, 88};

for (int i = 0; i < scores.length; i++) {
    System.out.println("Score " + i + ": " + scores[i]);
}
```

Output:

```
Score 0: 90
Score 1: 85
Score 2: 78
Score 3: 92
Score 4: 88
```

Example: Calculate Average

```
int[] scores = {90, 85, 78, 92, 88};
int sum = 0;

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

double average = (double) sum / scores.length;
System.out.println("Average: " + average);
```

Output: `Average: 86.6`

Note: Cast to `double` to avoid integer division.

Rock-Paper-Scissors with Arrays

Store choices in an array:

```
String[] choices = {"Scissor", "Rock", "Paper"};
//                   0          1       2
```

Computer picks a random number:

```
Random rand = new Random();
int computer = rand.nextInt(3); // 0, 1, or 2

System.out.println("Computer chose: " + choices[computer]);
```

User enters a number, we use it as index:

```
int user = input.nextInt();
System.out.println("You chose: " + choices[user]);
```

Rock-Paper-Scissors: Game Logic

Index mapping:

Index	Choice
0	Scissor
1	Rock
2	Paper

User wins when:

- user == 0 AND computer == 2 (scissor cuts paper)
- user == 1 AND computer == 0 (rock smashes scissor)
- user == 2 AND computer == 1 (paper wraps rock)

The Random Class

Import and create:

```
import java.util.Random;

Random rand = new Random();
```

Generate random integers:

```
int n = rand.nextInt(3); // Returns 0, 1, or 2
int m = rand.nextInt(10); // Returns 0 through 9
int p = rand.nextInt(6) + 1; // Returns 1 through 6 (dice)
```

Pattern: `nextInt(max)` returns 0 to max-1

Summary

- **Arrays** store multiple values of the same type
- **Zero-indexed** — first element is at index 0
- **Declare:** `int[] arr = {1, 2, 3};`
- **Access:** `arr[0]`, `arr[1]`, etc.
- **Length:** `arr.length` (no parentheses)
- **Loop:** `for (int i = 0; i < arr.length; i++)`
- **Random:** `rand.nextInt(n)` gives 0 to n-1

Lab 2: Array Practice

Complete `ArrayPractice.java`**:**

1. Create integer array with test scores
2. Print first and last elements
3. Calculate average using a loop
4. Create String array for colors
5. Print each color with a loop
6. Create Rock-Paper-Scissors choices array

This prepares you for Assignment 3!

Assignment 3: Rock-Paper-Scissors

Build a working game:

1. Create choices array
2. Generate random computer choice (0-2)
3. Get user input (0-2)
4. Display both choices using the array
5. Determine winner with if/else

Starter code and reference sheet in the repo!

Questions?

Lab 2: `ArrayPractice.java`

Assignment 3: Rock-Paper-Scissors