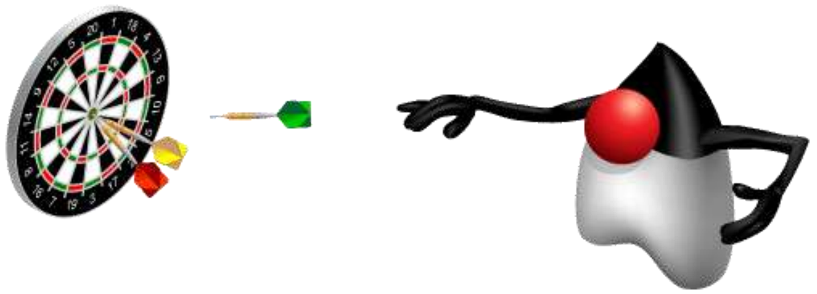
Managing Multiple Items

Objectives

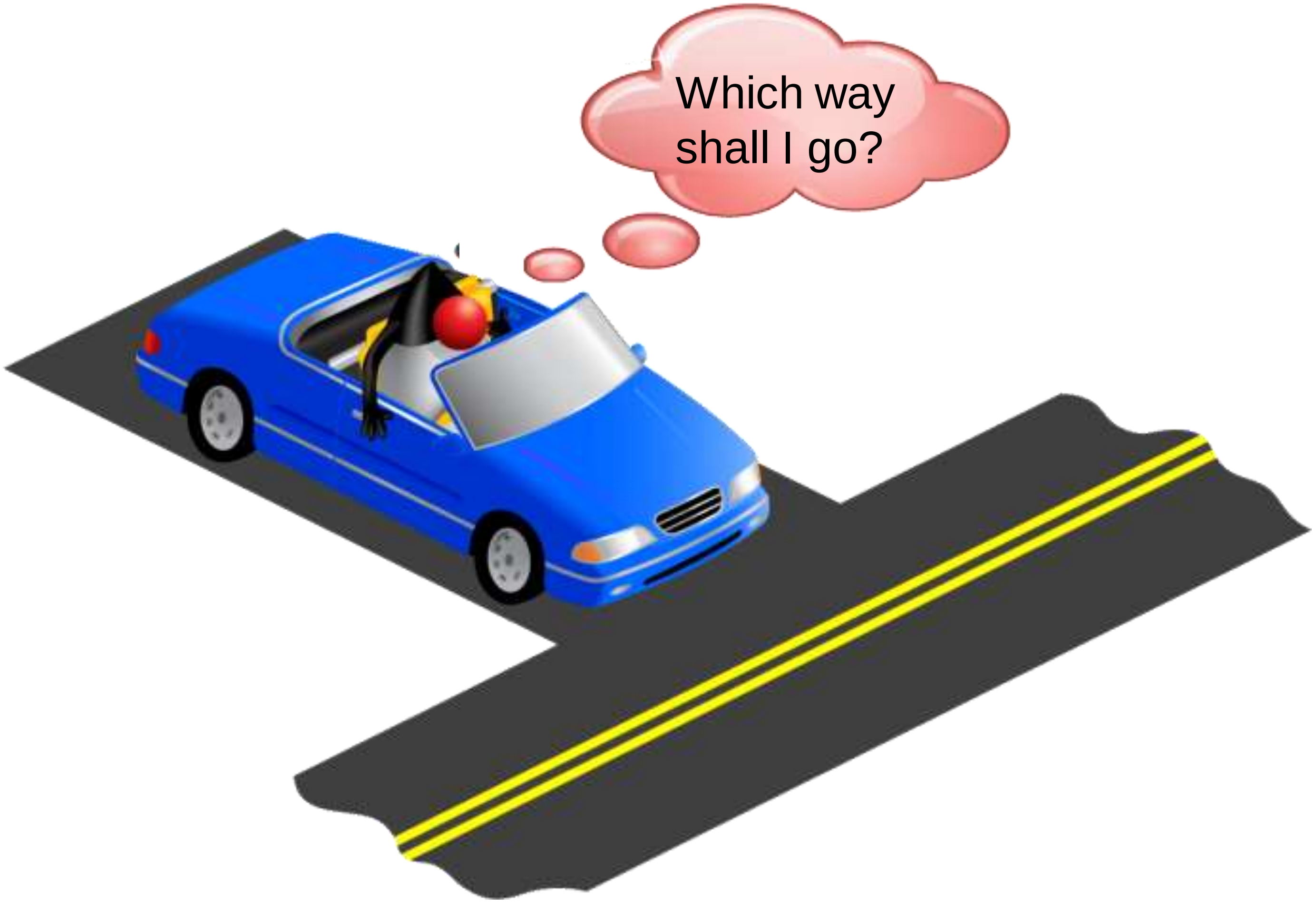
- After completing this lesson, you should be able to:
  - Explain what a boolean expression is
  - Create a simple `if/else` statement
  - Describe the purpose of an array
  - Declare and initialize a `String` or `int` array
  - Access the elements of an array
  - Explain the purpose of a `for` loop
  - Iterate through a `String` array using a `for` loop



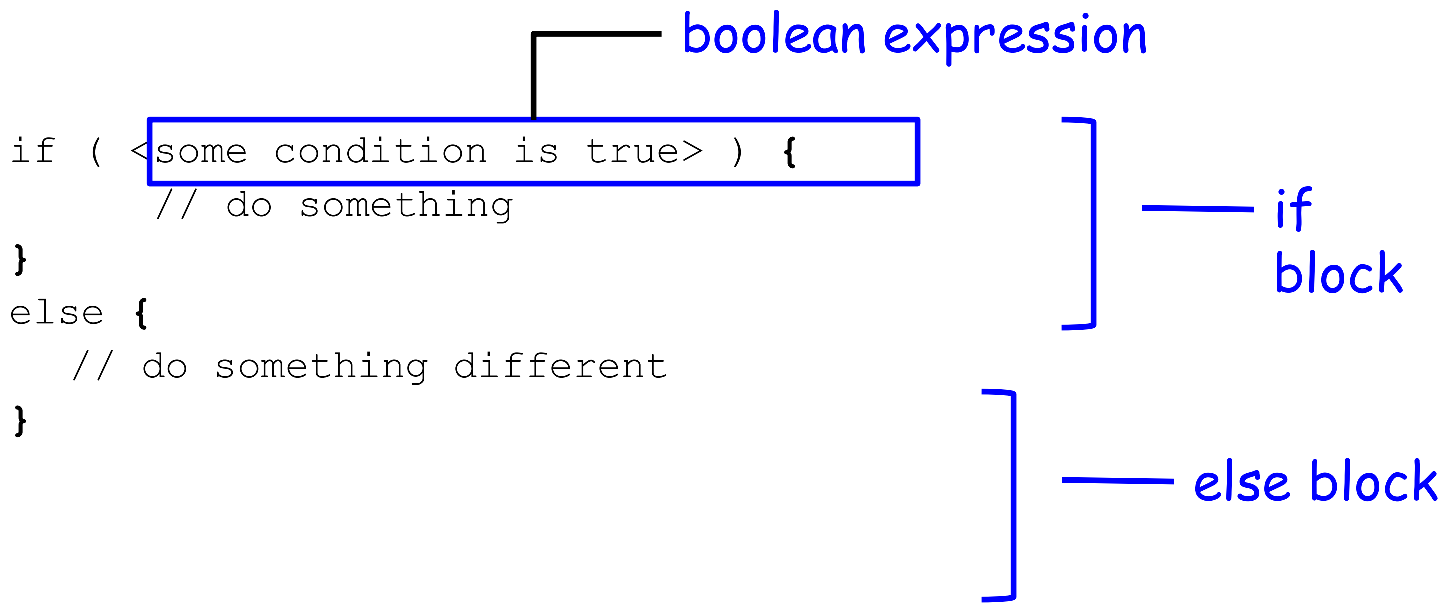
Topics

- Working with conditions
- Working with an array of items
- Processing an array of items

Making Decisions



The `if/else` Statement



```
if ( <some condition is true> ) {
        // do something
}
else {
   // do something different
}
```

Boolean Expressions

Review:

- `boolean` data type has only two possible values:
  - `true`
  - `false`

A boolean expression is a combination of variables, values, and operators that evaluate to `true` or `false`.

- `length > 10;`
- `size <= maxSize;`
- `total == (cost * price);`

Relational operators

Relational Operators

Condition	Operator	Example

Is equal to	`==`	`int i=1;`
		`(i == 1)`

Is not equal to	`!=`	`int i=2;`
		`(i != 1)`

Is less than	`<`	`int i=0;`
		`(i < 1)`

Is less than or equal to	`<=`	`int i=1;`
		`(i <= 1)`

Is greater than	`>`	`int i=2;`
		`(i > 1)`

Is greater than or equal to	`>=`	`int i=1;`
		`(i >= 1)`

Examples

Sometimes there is a quicker way to meet your objective. boolean expressions can be used in many ways.

```
int attendees = 4;
boolean largeVenue;

// if statement example
if (attendees >= 5){
    largeVenue = true;
}
else {
    largeVenue = false;
}

// same outcome with less code
largeVenue = (attendees >= 5);
```

Assign a boolean by using an `if` statement.

Assign the boolean directly from the boolean expression.

Exercise 5-1: Using `if` Statements

- In this exercise, you use an `if` and an `if/else` statement:
  - Declare a `boolean`, `outOfStock`.
  - `if quantity` > 1
    - Change the `message` variable to indicate plural
  - `if/else`:
    - `if` item is out of stock:
      - Inform the user that the item is unavailable
    - `else`
      - Print the `message`
      - Print the total cost

Quiz

What is the purpose of the `else` block in an `if/else` statement?

a. To contain the remainder of the code for a method
b. To contain code that is executed when the expression in an `if` statement is false
c. To test if an expression is false

Topics

- Working with conditions
- Working with an array of items
- Processing an array of items

What If There Are Multiple Items in the Shopping Cart?

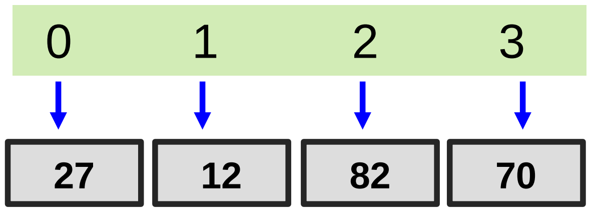
```
// Without an array
String itemDesc1 = "Shirt";
String itemDesc2 = "Trousers";
String itemDesc3 = "Scarf";

// Using an array
String[] items = {"Shirt","Trousers","Scarf"};
```

Not realistic if 100s of items!

Much better!

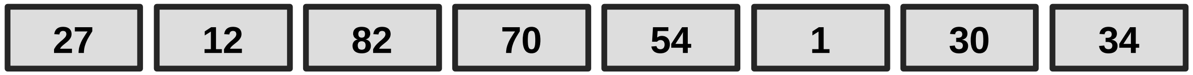
Introduction to Arrays

- An array is an indexed container that holds a set of values of a single type.
- Each item in an array is called an *element*.
- Each element is accessed by its numerical index.
- The index of the first element is 0 (zero).
  - A four-element array has indices: 0, 1, 2, 3.



Array Examples

Array of `int` **types**

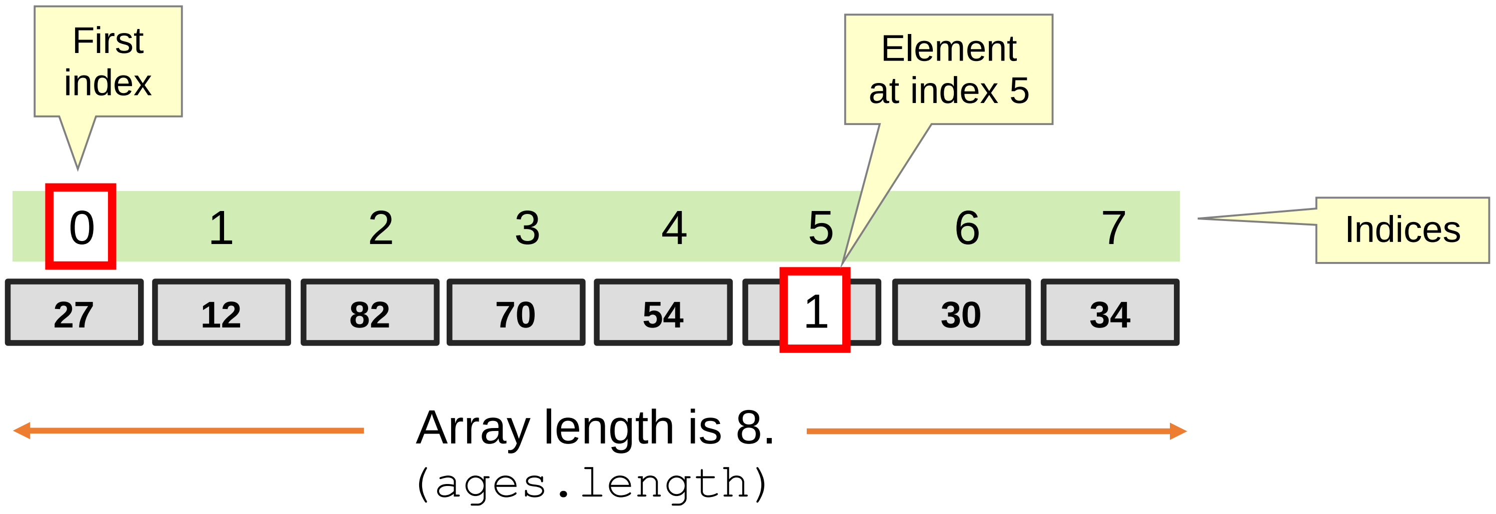


Array of `String` **types**

Hugh Mongus **Aaron Datires** **Stan Ding** **Albert Kerkie** **Carrie DeKeys** **Walter Mellon** **Hugh Morris** **Moe DeLawn**

Array Indices and Length

The `ages` array has eight elements.



Declaring and Initializing an Array

- Syntax:

```
type[] arrayIdentifier = {comma-separated list of
  values};
```

- Declare arrays of types `String` and `int`:

```
String[] names = {"Mary", "Bob", "Carlos"};
int[] ages = {25, 27, 48};
```

All in one line

Declaring and Initializing an Array

- Examples:

```java
int[] ages = new int[3];
ages[0] = 19;
ages[1] = 42;
ages[2] = 92;

String[] names = new String[3];
names[0] = "Mary";
names[1] = "Bob";
names[2] = "Carlos";
```

Multistep approach

Multistep approach

Accessing Array Elements

- Get values from the `ages` array:

```
int[] ages = {25, 27, 48};

int myAge = ages[0];
int yourAge = ages[1];
System.out.println("My age is " + ages[0]);
```

- Set values from the `names` array:

```
String[] names = {"Mary", "Bob", "Carlos"};

names[0] = "Gary";
names[1] = "Rob";
```

Exercise 5-2: Using an Array

In this exercise, you declare and initialize a `String` array to hold names. Then you experiment with accessing the array:

- Declare a `String` array, `names`, and initialize it with four `String` values.
- Print the number of items the customer wants to buy.
- Print one of the array elements.

Quiz

- Why does the following code not compile? Select all that apply.
- `int[] lengths = {2, 4, 3.5, 0, 40.04};`

a. `lengths` cannot be used as an array identifier.

b. All of the element values should have the same format (all using `double` values, or all using `int` values).

c. The array was declared to hold `int` values. `double` values are not allowed.

Quiz

- Given the following array declaration, which of the following statements are true?
- `int[] classSize = {5, 8, 0, 14, 194};`

a. `classSize[0]` is the reference to the first element in the array.

b. `classSize[5]` is the reference to the last element in the array.

c. There are 5 integers in the `classSize` array.

d. `classSize.length = 5`

Topics

- Working with conditions
- Working with an array of items
- Processing an array of items

Loops

Loops are used in programs to repeat blocks of statements

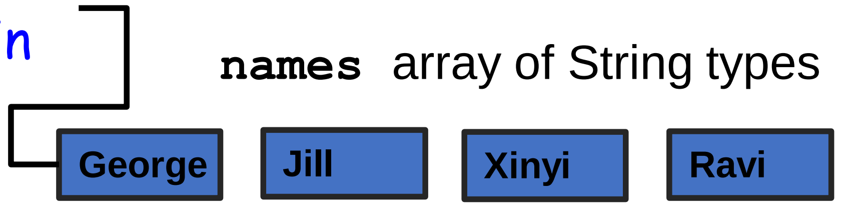
- Until an expression is false

  or

- For a specific number of times:
  - I want to print each element of an array.
  - I want to print each element of an `ArrayList`. (The `ArrayList` class is covered in the lesson titled “Working with Arrays, Loops, and Dates.”

Processing a String Array

Loop accesses each element in turn.



Each iteration returns the next element of the array.

```
for (String name : names ) {
    System.out.println("Name is " + name);
}
```

Output:

```
Name is George
Name is Jill
Name is Xinyi
Name is Ravi
```

Using `break` with Loops

`break` example:

```
int passmark = 12;
boolean passed = false;
int[] scores = {4,6,2,8,12,35,9};
for (int unitScore : scores){
    if (unitScore >= 12){
        passed = true;
        break;
    }
}
System.out.println("At least one passed? " +passed);
```

No need to go through the loop again, so use break.

Output:

```
At least one passed? true
```

Exercise 5-3: Using a Loop to Process an Array

- In this exercise, you loop through an array called `itemPrices` to print a message indicating each item price.

Quiz

Given the following code,

```
int[] sizes = {4, 18, 5, 20};
for (int size : sizes){
    if (size > 16){break;}
    System.out.println("Size: "+size + ", ");
}
```

which option below shows the correct output?

a. `Size: 4,`

b. `Size: 4`

c. `Size: 4,`
  - `Size: 5,`

d. There is no output.

Summary

In this lesson, you should have learned how to:

- Use a boolean expression
- Create a simple `if/else` block
- Describe the purpose of an array
- Declare and initialize a `String` or `int` array
- Access the elements of an array
- Explain the purpose of a `for` loop
- Iterate through a `String` Array using a `for` loop