Lesson 9-2: ArrayLists and Arrays

Computer Science 46A: Introduction to Programming

San José State University

Announcements

- Homework #9 will be posted soon
- Lab this Friday 4/11
- Weekly reading assignment will be posted
- Next weeks lecture will be from another Professor
- Last day of class May 8
- Midterm 2 – May 6 (might change)
- Final Exam – May 14, Wednesday 10:45-12:45

Learning Objectives

By the end of this lesson, you should be able to:

1. Write a class to modify an ArrayList
2. Construct an array of a given size and call methods on it
3. Compare and contrast simple ArrayList and array methods

ArrayLists Revisited

ArrayList Syntax

- To declare an ArrayList, use the following syntax:

```
ArrayList<type> [name] = new ArrayList<type>();
```

Common ArrayList methods (review)

- `add([new value]);`
  - The add method will append new values to the end of the ArrayList
- `set(int index, [new value]);`
  - The set method will change the value at a given index to a new value
- `remove(int index);`
  - The remove method will remove the value at a given index
  - The remove method also returns the given element
    - In other words, you can store the element in a new variable as output from remove
- `size();`
  - Gets the total amount of elements that are in the ArrayList

Common ArrayList method: get

- `get(int index);`
  - The get method will retrieve a value from an ArrayList

myNumbers				
1	2	3	4	5

Example:

```
myNumbers.get(3);
```

Output:

```
4
```

ArrayLists in Classes – an Example

- Like all Objects, we can declare ArrayLists as instance variables in Classes and declare methods to modify them
- Example:

CryptoList is a class that acts like a reference that manages a list of different **Crypto**s

Crypto class stores Symbol, price, and year

CryptoListTester is a unit test to ensure that **CryptoList** works as expected

Arrays

Array – an introduction

- An array can be used to store a collection of objects
- Consider the following array called myNumbers:

myNumbers				
1	2	3	4	5

- The array stores five integers into memory
- Arrays can only store one type of variable or object
  - .e.g ints, doubles, Strings, Rectangles, etc
- To create Arrays, there is no need to import a utils class

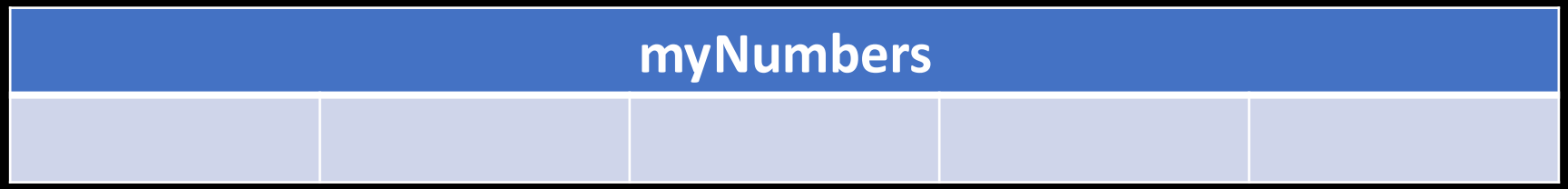
Array Syntax

- To declare an Array, use the following syntax:

```
[type][] [name] = new [type][size]
```

- For example:

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



The declaration constructs the Array – next we need to fill it

- Note: once declared, the size of on array cannot change
  - The array has a length property which we access with `myNumbers.length`

Accessing Array Elements

- To access elements in an array, we use the brackets with the index of the element we are interested in

myNumbers				
1	2	3	4	5

Examples:

- To get an element at index 2, use:

```
int thirdNumber = myNumbers[2]
```

- To set an element at index 4 to a new number (e.g. 7), use:

```
myNumbers[4] = 7
```

Array Syntax – part 2

- We can also declare and initialize an array at the same time using curly braces {} and the elements of our array:

```
int[] myNumbers = {1, 2, 3, 4, 5}
```

myNumbers				
1	2	3	4	5

Poll Everywhere: Question 1

- Which of the following lines declares the main method in a class?

```
A) public static void main()

B) public void main(String[] args)

C) public static void main(String args)

D) public static void main(String[] args)
```

PollEv.com/narayanbalasubramanian644

Printing Arrays

- For arrays, we cannot use the System.out.print() method
- Most classes have a (sometimes hidden) .toString() method but array structures do not have this
- One way we can print arrays is to use the Arrays class

```
import java.util.Arrays;
System.out.print(Arrays.toString(myNumbers));
```

The Arrays class has a toString method

See example code in **MyNumbers**

Looping over array values:

- Similar to ArrayLists, the values in an array can be accessed in succession using an "enhanced for loop"
- Syntax:

```
for ([type] [var name] : [array name])
{
    // enter code here
}
```

See example code in **MyNumbersLoop**

Adding or removing values from an array

- Once declared, arrays are a fixed size – they don't change
- In other words, we cannot add or remove a value from array since this will change its size
- If we want to add or remove a value, we need to make a copy of the array with the new size and transfer over the elements

  Option 1: Make a loop to copy the values

  Option 2: Use the arraycopy method from the System class

  Syntax:

```
System.arraycopy(oldArr, startIndex, newArr, startIndex, nItems)
```

See example code in **MyNumbersLoop**

Arrays vs ArrayLists

ArrayLists vs Arrays – a visual

ArrayList

Array

ArrayLists change size as needed, arrays are a fixed size from declaration

ArrayLists vs Arrays - operations

Example: myNumbers = [1, 2, 3, 4, 5]

Operations	ArrayLists	Arrays
Access an element at a given index	myNumbers.get(index)	myNumbers[index]
Set an element to N at a given index	myNumbers.set(index, N)	myNumbers[index] = N
Add or remove an element	myNumbers.add() or myNumbers.remove()	Create a new array with the new numbers of elements
Get the number of elements	myNumbers.size()	myNumber.length

Participation Exercise 9-2a: `CircleList`

Goal: Write a class to store Circle objects in an ArrayList and call methods on it

```
Average area: 235.0013.
Number of circles: 2.
[Circle[x=2,y=5,radius=5.51], Circle[x=-2,y=-7,radius=10.92]]
Largest area: 374.6236.
```

Output of the `CircleListTester` script (for the case of 2 circles)

Codecheck Link: HERE and on Canvas

Participation Exercise 9-2b: FlowerArray

Goal: Use an Array called garden to store a collection of flower names and experiment with Array functionality

```
Number of flowers: 3
Enter a flower: daisy
Enter a flower: poppy
Enter a flower: sunflower
daisy poppy sunflower
Enter a flower: rose
daisy poppy sunflower rose
Enter a flower: zinnia
Enter a flower: lily
daisy lily sunflower zinnia
```

Output of the FlowerArray script

Codecheck Link: HERE and on Canvas