CS 302: Introduction to Programming in Java

Lecture 12

Review

- What is the 3-step processing for using Objects (think Scanner and Random)?
- Do objects use static methods or non-static (how do you know)?

THE UNIVERSITY of WISCONSIN MADISON

Null

- Used to indicate a reference variable that points to nothing (i.e. a null memory address)
- Only can be used with reference variables!
- Different from the empty String!!!

```
String x = null;
String y = "";
```

Handling Null

- Null is a possible value for a memory address
- Therefore check with == or !=
- Example:

```
String x = null;
if (x == null) {
  S.o.pln("This will print out");
}
```

Different from how we usually compare Strings!!!

Array Limitations

- Fixed length
  - What if we get a new test score but we have filled up the array?
  - No way to simply add a new "box" for the score
  - Must create a new array that is long enough for all the old scores and the new score and copy all the values over
- Solutions
  - Use of partially-filled arrays
  - ArrayLists

THE UNIVERSITY of WISCONSIN MADISON

Array Lists

- Array Lists can grow and shrink as needed
- ArrayList class already has methods for inserting elements as specific indexes, removing elements, etc.
- Using Array Lists:
  - 3 Steps to using an Object
    - import the package: import java.util.ArrayList
    - Instantiate an object: ArrayList<String> names = new ArrayList<String>();
    - Use the object: data.add(“Yinggang");

Using an ArrayList Object

```
ArrayList<type> variableName = new ArrayList<type>();
```

Must be a reference type (ex. String) – cannot be primitive (ex. int, double, boolean, char)

variableName.size() - returns the size of the ArrayList as an int

variableName.add(element) - appends element to the end of the list and automatically increases its size

variableName.set(i, element)

0 <= i < variableName.size() - sets variableName → i= element

More ArrayList Benefits

- ArrayList<String> names;
- Easy to print out
  - System.out.println(names);
- Easy to copy
  - ArrayList<String> copyOfNames = new ArrayList<String>(names);

Pass array to copy into constructor

  - What if instead had done:
    - ArrayList<String> copyOfNames = names;

More Methods

- indexOf
- lastIndexOf
- remove
- isEmpty
- More...
- How would I know these?
- Answer: Java API (also control-space / apple-space...)

THE UNIVERSITY of WISCONSIN MADISON

Array Lists and Methods

- Array Lists can be method parameters or return types (like arrays)

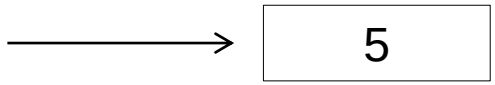
```
public static ArrayList<String> reverse(ArrayList<String> names)
{
   ArrayList<String> reversedNames = new ArrayList<String>();
   for (int i = names.size() - 1; i >= 0; i--)
   {
      reversedNames.add(names.get(i));
   }
  return reversedNames;
}
```

Wrapper Classes

- Problem – type for ArrayList can only be reference variables – what if we wanted an ArrayList of ints or doubles?
- Solution: Wrapper classes
  - int -> Integer, double -> Double, char -> Character, boolean -> Boolean
  - ArrayList<Integer> intArrayList = new ArrayList<Integer>();
  - Auto-Boxing = conversion between primitive and wrapper classes (int -> Integer)

THE UNIVERSITY of WISCONSIN MADISON

Auto-Boxing

- Conversion between primitives and their respective wrapper class goes on "behind the scenes"
- Integer x = 5; // x 

- int y = x; //y = 5

Array List Example

- Find the largest value

```
ArrayList<Integer> vals = new ArrayList<Integer>();
//fill vals with random values
for (int i = 0; i < rand.nextInt(100); i++)
   { val.add(rand.nextInt()); }
int largest = vals.get(0);
for(int i = 1; i < vals.size(); i++) //find largest
{
    if (vals.get(i) > largest) largest = vals.get(i);
}
```

THE UNIVERSITY of WISCONSIN MADISON

Reading Input

```
ArrayList<Double> testScores = new ArrayList<Double>();
while (in.hasNextDouble())
{
   testScores.add(in.nextDouble());
}
```

Length with Arrays, ArrayLists, and Strings

- Strings -> stringName.length();
- Arrays -> arrayName.length;
- ArrayLists -> arrayList.size();

ArrayList Practice 1 (maybe take tome)

- Write a method to sort an ArrayList of Integers using Selection Sort:

public static void selectionSort(ArrayList<Integer> data)

- Selection Sort (basically what humans usually do):
  - For each index in the array:
    - Find the current smallest element from [index...end]
    - Swap its value with the element currently in index

ArrayList Practice 2 (take home)

- Write a method to sort an ArrayList of Integers using the BubbleSort method:

public static void bubbleSort(ArrayList<Integer> data)

- Bubble Sort:
  - Iterate through all elements until no swaps occur
  - In each interation, compare every pair of adjacent elements and swap them if they are out of order