Java Arrays Part 3 Notes

Team Treehouse

May 27, 2020

1 Adding and Removing Items Means Copying

- Adding / Removing elements → new array must be declared and copy into it
- Adding
  - **Syntax:** *System.arraycopy(Object src, int srcPos, Object, dest, int destPos, int length);*
    - **src:** Is the source array
    - **srcPos:** Is the starting position of the source array
    - **dest:** Is the destination array
    - **destPos:** Is the starting position in the destination data
    - **length:** Is the number of array elements to be copied

Example:

```
1    String[] classmates = {"Ben", "Johnny", "Pasan"};
2    String[] classmatesAndMe = new String[4];
3
4    System.arraycopy(classmates, 0, classmatesAndMe, 0, classmates.
     length);
5
6    // Returns [Ben, Johnny, Pasan, null]
7
```

Listing 1: lesson_01/Explore.java

Notes:

- Files can be compiled and displayed by typing *javac Explore.java && java Explore* in terminal

2 Quiz 2

1. Since you can add elements to an array by making a new array and copying values into it, how do you suppose you go about removing items?

   A. You still make a new copy, one element smaller than the current one, and simply don't copy over the value that you want to remove.

   B. I've given up all hope.

   C. You call the method deleteItemAt and pass the index you want deleted.

   Answer: A

2. Why can you not simply just add an item to an array?

   A. Polymorphism doesn't allow for growth of objects that are statically typed.

   B. The interface does provide an proper method that allows for this. It is due to method access levels.

   C. An array's length is immutable and it requires elements to be located in a contiguous order in memory.

   Answer: C

3 Sorting

- **Syntax:** *Arrays.sort(DATA_TYPE[] arr, int from_index, int to_index)*
- **Syntax 2:** *Arrays.sort(DATA_TYPE[] arr, Comparator c)*
  - Comparator is like *lambda* in python
  - *Arrays.sort* is in *java.util.Arrays*
  - *Comparator* is in *java.util.Comparator*

```
1  import java.util.Arrays;
2  import java.util.Comparator;
3
4  public class Explore {
5      public static void main(String[] args) {
6          String[] classmates = {"Ben", "Johnny", "Pasan"};
7
```

```
8             Arrays.sort(classmates, Comparator.comparing(String::
  length)); // <- sorts based on length of string
9             System.out.println(Arrays.toString(classmates));
10
11            // Returns ["Ben", "Pasan", "Johnny"]
12        }
13    }
14
```

Listing 2: lesson_03/Explore.java

4 Quiz 3

1. Assume that you have a class representing Planet. It has a method named *getDistanceInAstromicalUnits* that helps figure out the distance from the Sun.

   Instances of the Planet class are created and placed in an Planet[] array named planets.

   Which code snippet would allow you to sort the planets array by distance?

   A. Arrays.sort(planets);
   B. Planet.sortBy(getAstronomicalUnits());
   C. Arrays.sort(planets, Comparator.comparing(Planet::getDistanceInAstronomicalUnits));

   Answer: C