CISC 3115 EWQ6

Array, ArrayList, Arrays, and Collections

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Discussed in previous modules
  - Math, Random, Point2D
  - Wrapper classes of the pyrimitate data types
  - BigInteger, BigDecimal
  - String, StringBuilder, StringBuffer
- ArrayList, array, arrays, and Collections

ArrayList: Motivation

- What is an array?
  - A collection of items, typically indexed, with equal random access time.
- What are the limitations of array?
  - The size of an array must be known when the array is being created, and is fixed.
- Vector
  - (Mostly, if not all) random access takes equal amount of time
  - It can grow and shrink as needed
  - Java's java.util.ArrayList is the realization of the vector data structure.

The ArrayList Class

java.util.ArrayList<E>	
+ArrayList()	Creates an empty list.
+add(o: E) : void	Appends a new element o at the end of this list.
+add(index: int, o: E) : void	Adds a new element o at the specified index in this list.
+clear(): void	Removes all the elements from this list.
+contains(o: Object): boolean	Returns true if this list contains the element o.
+get(index: int) : E	Returns the element from this list at the specified index.
+indexOf(o: Object) : int	Returns the index of the first matching element in this list.
+isEmpty(): boolean	Returns true if this list contains no elements.
+lastIndexOf(o: Object) : int	Returns the index of the last matching element in this list.
+remove(o: Object): boolean	Removes the element o from this list.
+size(): int	Returns the number of elements in this list.
+remove(index: int) : boolean	Removes the element at the specified index.
+set(index: int, o: E) : E	Sets the element at the specified index.

Remark: Generic Type

- ArrayList is known as a generic class with a generic type E.
  - One can specify a concrete type to replace E when creating an ArrayList.
  - This type indicates the type of objects to which the list contains references to
  - Example:

    ```
    ArrayList<String> cities = new ArrayList<String>();
    ```

  - This ArrayList object can be used to store (the references to) strings

Differences and Similarities between Arrays and ArrayList

Operation	*Array*	*ArrayList*
Creating an array/ArrayList	`String[] a = new String[10]`	`ArrayList<String> list = new ArrayList<>();`
Accessing an element	`a[index]`	`list.get(index);`
Updating an element	`a[index] = "London";`	`list.set(index, "London");`
Returning size	`a.length`	`list.size();`
Adding a new element		`list.add("London");`
Inserting a new element		`list.add(index, "London");`
Removing an element		`list.remove(index);`
Removing an element		`list.remove(Object);`
Removing all elements		`list.clear();`

Iterating ArrayList

- Use the for loop
  - Example

```
for(int i=0; i<arrayList.size(); i++) {
…
}
```

- Use the enhanced for loop
  - Example

```
for (Fruit fruit: fruitList) {
…
}
```

- Use the forEach method of the ArrayList
  - To be discussed in the future

Arrays to Array Lists

- Creating an ArrayList from an array of objects:
  - Using the java.util.Arrays class
  - Example

```
String[] array = {"red", "green", "blue"};

ArrayList<String> list =

        new ArrayList<String>(Arrays.asList(array));
```

Array Lists to Arrays

- Creating an array of objects from an ArrayList:
  - Using ArrayList's size() toArray method
  - Example

```
ArrayList<String> list = new ArrayList<String>();
……
String[] array1 = new String[list.size()];
list.toArray(array1);
```

Using the Collections Class

- The java.util.Collections class consists exclusively of static methods that operate on or return collections, e.g., ArrayList
- min, max, shuffle an Array List

ArrayList: min and max

- Example: min

```
String[] array = {"red", "green", "blue"};
System.out.println(Collections.max(
  new ArrayList<String>(Arrays.asList(array))));
```

- Example: min

```
String[] array = {"red", "green", "blue"};
System.out.println(Collections.min(
  new ArrayList<String>(Arrays.asList(array))));
```

ArrayList: Shuffle

- Example: shuffle

```
Integer[] array = {3, 5, 95, 4, 15, 34, 3, 6, 5};
ArrayList<Integer> list = new
    ArrayList<>(Arrays.asList(array));
java.util.Collections.shuffle(list);
System.out.println(list);
```

Example: Design Stack Using ArrayList

- Observe the animation
  - https://liveexample.pearsoncmg.com/dsanimation/StackeBook.html

Questions

- Concept of array and vector
- Java's ArrayList
- Two utility classes
    - Arrays and Collections

Exercise

- This exercise builds from the previous exercise where we implemented a hierarchy of fruits.
  - Create subdirectory/subfolder for the exercise
  - Make sure each class in the hierarchy has its own implementation of the toString() method
  - Add an equal method and a hashCode method to each class, and make sure the contract of the hashCode method is maintained.
  - Revise the GoldenDelicious and McInosh classes so that they cannot be subclassed.
  - In your client class, create an ArayList of fruits of various kind.
    - Print the list of fruits
    - Take two random fruits out of the list, and compare whether they are equal