CSCI-201: Principles of Software Development

Fall 2020

Lab02: Unbounded Arrays

Introduction:

Thinking abstractly, an unbounded array is an array in the sense that we should be able to add a new element to the end of the array and delete an element from the end of the array. This data structure will not run out of space unless we hit some hard memory limit, which is unlikely given modern operating systems.

The general implementation strategy is as follows. We maintain an array and an internal index *size* which tracks how many elements are actually in the array. When we add a new element, we increment *size*, when we remove an element, we decrement size. The tricky issue is how to proceed when we the array is full and want to add another element. At that point, we allocate a new array of a larger length (we double its length) and copy the elements we already have to the new array. Removing an element from the end is simple: we just decrement size.

Implementing Unbounded Arrays:

Implement the Uba class according to the interface provided. In the interface you'll see three functions that you'll need to implement. They have already been included in the Uba.java. Here are some general tips to guide your coding:

add:

- Remember to make sure that there is enough room in the array before adding.

- Don't forget to increment your counter!

toString:

- The string representation should begin and end with square brackets

- Elements should be separated by a comma and a space

- Example: [elem1, elem2, elem3] where elem is some element (int, String, etc.)

Good luck! Have fun!

Testing:

Expected output (you must match it exactly):

```
[]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]
49
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48]
[uba0, uba1, uba2, uba3, uba4, uba5]
uba5
[uba0, uba1, uba2, uba3, uba4]
```

Grading Criteria:

1) complete the lab following the instructions above and

2) show your understanding of the lab material by answering questions upon the following check-offs.

Check-off Questions:

1. Consider

   ```
   int [] a = {1,2,3};
   int [] b = a;
   int [] c = new int[3];
   System.arraycopy(a,0,c,0,a.length);
   ```

   Show the results of these comparisons:

   a) a == b;

   b) a == c;
2. What is a return type of a constructor?
3. What is the main purpose of a constructor?
4. What is the main difference between String and StringBuffer classes?
5. Why did we have to define the toString method? What would happen if we didn't?
6. Can we redefine our Uba object with a different given type?
7. Why do we use Generics in this lab as opposed to using the Object base class?

8. Consider

```
String s1 = new String(“cs201”);
String s2 = “cs201”;
String s3 = s1
```

Show the results of these comparisons:

a) s1 == s2;
b) s2 == s3;
c) s1 == s3;