Assignment #6
CSE110 - Arizona State University

Topics

- Arrays
- Classes

Coding Guidelines:

- Give identifiers semantic meaning and make them easy to read (examples numStudents, grossPay, etc).
- Keep identifiers to a reasonably short length.
- User upper case for constants. Use title case (first letter is upper case) for classes. Use lower case with uppercase word separators for all other identifiers (variables, methods, objects).
- Use tabs or spaces to indent code within blocks (code surrounded by braces). This includes classes, methods, and code associated with ifs, switches and loops. Be consistent with the number of spaces or tabs that you use to indent.
- Use white space to make your program more readable.

Part #1 - Programming

Your assignment is to create a class called `NumberCollection` in a file called `NumberCollection.java` (there is no `main` method in this class). The `NumberCollection` class will act as a resizable array of unique integers and provide methods to act on the collection. Therefore the `NumberCollection` class needs the following instance variables:

- `numberArray` - an array of integers to hold the collection.
- `count` an int to hold how many integers have been stored in the array so far.

Note: The instance variable count will usually be different than the capacity of the array (`numberArray.length`).

In order to provide access to the collection, the class `NumberCollection` must include the following constructor and methods. (If your class does not contain any of the following methods, points will be deducted.)

- `public NumberCollection(int arraySize)` - It constructs an empty `NumberCollection` object with an array capacity specified by the integer parameter `arraySize`. That is, it stores a new int array with size `arraySize` in the `numberArray` instance variable.
- `private int indexOf(int searchingNum)` - It returns the index of the number specified by the parameter is located. If the number is not found, it returns -1. This is a helper method.
- `public boolean addNumber(int numberToAdd)` - The method will add `numberToAdd` at the smallest available index in the array, if the number is not in already in the array. Be sure to check if the number is already in the array:
  - Hint: This can be done using the `indexOf` method to see if it returns -1 or not.

  - If the number is in the array, then it will not be added again and the method returns `false` to indicate that nothing was added.
  - If the number is NOT in the array, continue.

  If the number is not in the array, be sure to check if the array has reached its capacity.

  - Hint: What is the relationship between the instance variable `count` and the length of `numberArray` if the capacity is reached.
  - If the capacity is reached, double the length of `numberArray`.
  - Hint: You can use the `Arrays.copyOf()` method for this as described in Section 6.3.9 of the book.
  - After updating the capacity, if necessary, continue.

  If the number is not in the array and after updating the capacity, if necessary, then add `numberToAdd` to `numberArray` at the smallest available index.

  - Hint: What's the smallest available index's relationship to `count`?
  - Hint: Be sure to increase count after adding a new element.

  If the number is added successfully, then the method returns `true`.

- `public int findMax()` - It finds the maximum number among the numbers stored so far (at the time when this method is called), and returns it. If the array is empty, return 0.
- `public int findMin()` - It finds the minimum number among the numbers stored so far (at the time when this method is called), and returns it. If the array is empty, return 0.
- `public int computeSum()` - It computes and returns the sum of numbers stored in `numberArray` so far (at the time when this method is called.) If the array is empty, return 0.
- `public String toString()` - Returns a String containing a list of numbers stored in `numberArray`. An example of such string can be:

  {3, 6, -1, 3, 23, -50, 43}

  The string should start with a '{' and end with a '}'.

  Note: `Arrays.toString(numberArray)` cannot be used since will not use the required delimiters. That is, you have to write your own loop.

Hint: An example of most of these methods can be found in the subsections of Section 6.3 in the book beginning on page 258.

Helpful Hints

- Work on it in steps - write one method, test it with a test driver and make sure it works before going on to the next method.
- Always make sure your code compiles before you add another method.
- Your methods should be able to be called in any order.

Save the `NumberCollection` class in a file called `NumberCollection.java` and use the following program stored in `Assignment6.java`, which has the main method to create new `NumberCollection` objects and to test your class. You do NOT need to modify `Assignment6.java`.

The program will ask a user to enter a size for the array. Then it will show the following menu to a user:

```
Command Options
-----------------------------------
a: add an integer in the array
b: display the array
c: compute and display the maximum
d: compute and display the minimum
e: compute and display the sum
?: display the menu again
q: quit this program
```

Sample Outputs

(user input is in bold)

Please enter a size for the array.
3

Command Options
---

a: add an integer in the array
b: display the array
c: compute and display the maximum
d: compute and display the minimum
e: compute and display the sum
?: display the menu again

Please enter a command or type ?
b

{}

Please enter a command or type ?
c

The maximum is: 0

Please enter a command or type ?
d

The minimum is: 0

Please enter a command or type ?
e

The sum is: 0

Please enter a command or type ?
a

Please enter an integer to add.
5

5 successfully added.

Please enter a command or type ?
a

Please enter an integer to add.
12

12 successfully added.

Please enter a command or type ?
a

Please enter an integer to add.
-2

-2 successfully added.

Please enter a command or type ?
a

Please enter an integer to add.
41

41 successfully added.

Please enter a command or type ?
a

Please enter an integer to add.
9

9 successfully added.

Please enter a command or type ?
a

Please enter an integer to add.
-21

-21 successfully added.

Please enter a command or type ?
b

{5, 12, -2, 41, 9, -21}

Please enter a command or type ?
c

The maximum is: 41

Please enter a command or type ?
d

The minimum is: -21

Please enter a command or type ?
e

The sum is: 44

Please enter a command or type ?
a

Please enter an integer to add.
41

41 is already in the array. 41 was not added.

Please enter a command or type ?
b

{5, 12, -2, 41, 9, -21}

Please enter a command or type ?
?

Command Options

---

a: add an integer in the array
b: display the array
c: compute and display the maximum
d: compute and display the minimum
e: compute and display the sum
?: display the menu again
q: quit this program

Please enter a command or type ?
q

Submission

- Go to the course web site (my.asu.edu), and then click on the on-line Submission tab.
- Submit your **Assignment6.java and NumberCollection.java** file on-line. Make sure to choose Hw6 from drop-down box.
- `Assignment6.java` should have the following, in order:
  - In comments, the assignment header.
  - The working Java code requested in Part #1.
- The `NumberCollection.java` and `Assignment6.java` files must compile and run as you submit it. You can confirm this by viewing your submission results.

Important Note: You may resubmit as many times as you like until the deadline, but we will only mark your last submission.

NO LATE ASSIGNMENTS WILL BE ACCEPTED.