README FILE: Project 1 Parv Bhardwaj

Project Description:

This project is a basic implementation of an Abstract Data Structure(ADT). The main class NumArrayList can be used to store doubles. As opposed to an array, it has no fixed size. The capacity automatically increases when the NumArrayList reaches its limit. NumArraylist has other features like removing duplicates, inserting values, deleting values, looking them up, etc. The project is an alternate implementation of the JavaAPI ArrayList.

Implementation(Logic):

The logic used to implement the ADT. The most important bit is figuring out how can we add, removeDuplicates, and insert elements in the NumArrayList.

1. add method:

For adding doubles to the list, I first created a double arrayList called ParentDoudleArray and gave it a size. When users call the add method five things happen:

1. The index variable is reassigned to 0. To store the first element in the arrayList.

2. A temporary array can store one more element than the parentDoubleArray. This space is for the element that is about to get added to the array. The temporary array copies element from the ParentDoubleArray and the loop runs till the length of ParentDoubleArray.

3. A new array called newarr[] is created, and it is assigned to the tempArray i.e it is now a copy of the tempArray.

4. We add the element to newarr and then increment the index.

5. Now we assign the parentDoubleArray to newarr i.e it has the previous elements and the newly added element.

NOTE: All steps are repeated when the add method is called.

2. print method:

Print method takes the parentDoubleArray and prints it. It runs till(but not including) the length of the array-1 because ParentDoubleArray has extra space for storing one more element in the last index.

3. insert method:

insert method takes integer i(index) and a value(double).

1) index entered is greater than the length of the parent array:

I just call the add method as the double entered will be added at the end of the list

2) the normal case where i entered is within the bounds:

I created a tempArray first to copy all th elements from the parentDoubleArray. The temp Array has a size of 1 + parentDoubleArray.length. I then make a for-loop to copy elements in the temp array.

if case:

I have a variable to track the index called indexTemp it increments with the local variable k until the integer i found. indexTemp decrements by 1 when i is found and the value is added to tempArray.

else case:

indexTemp increments with the local variable and tempArray[k] = parentDoubleArra[indexTemp]

The overall logic of the increment method. When the index i is found, original value is changed with the new value and the original value is shifted to next index using indexTemp(by decreasing it by 1 when i is found).

4. lookup method:

The lookup method first checks whether the index entered is within the array-bounds or not. If it's not it prints an ArrayOutOfBoundsException. I intentionally decided to print the statement instead of throwing an exception. When an exception is thrown the exception is returned instead of the foundNUmber(double). I made the lookup in way that it returns -1.0410 as default when the index is not fund. and the tests are made keeping that in mind. This way all the tests pass. I tried testing for the exception as the input value but my IDE(IntelliJ) showed errors and the test code didn't compile.

NOTE: By printing the error and be returning a default value the method works perfectly as it still handles the exception

5. remove:

Very similar to the insert method. The only difference is when the index to be removed is found, we don't add the element to temp array. The rest continues as expected.

6. toString:

Returns "" when the NumArrayList is null. Uses StringBuilder and its toString method to append the elements into the StringBuilder separated by space.

NOTE: Since it wasn't explicitly mentioned in the project. This particular toString doesn't remove the extra space at the end. If I have time by the end of this project to handle the last space situation, I'll update my logbook and code.
27th Jan 2022: UPDATE: It removes now

7. removeDuplicates:

removeDuplicates uses the contains method to logic to check if the element that's about to be inserted into the tempArray has already been added into the tempArray. If it hasn't been already added, the element is added to the array. If it has been added before the index just increments without adding the element. The main loop is a while loop and the inner loop is a for-loop. This way it's easier to see the functionality of the method.

8. contains:

checks for the element in the parentDoubleArray and returns a boolean accordingly.

9. capacity:

checks whether the index variable is greater than the capacity at the current moment and returns the bigger value.

10. size:

returns the index variable.

NOTE FOR METHODS 9 AND 10:

I use the variable index in a way that it tells us where the next element will be stored as well as the size of the array the arrayList increases by 1 its size whenever the index reaches the capacity.

11. equals:

It first checks whether the sizes of both the numLists are equal. if they are equal then it proceeds to check the
individual elements using lookup method and if the values don't match it sets the isEqual boolean to false and returns
it at the end of the method. If the sizes are different it simply returns false.

Technology used:

Developed in IntelliJ using Java jdk 16. Since it was an ADT implementation no additional library was used. The tests were created using JUNIT TEST 4.

How to run the Tests:

Simply run the file. The test cases are already present in the file.

IMPORTANT: You need to have JUnit TEST 4 lib configured in your IDE.

Development log:

Jan 19 2022 - Started the project

Jan 20 2022 - Made the NumList interface

Jan 21 2022 - Made getter and setter methods in the NumArrayList class

Jan 22 2022 - Made a JUnit test for the getter setter methode and started coding the rest of the required functions.

Created JUnit test for the isEmpty method.

Jan 23 2022 - Added the add method. Explained the logic of add method in README file. Submitted the project with 30% progress.

Jan 23 2022 after submission:- Coded the contains, lookup and partially the removeDuplicates method and made test cases for the methods

Jan 24 2022 - Coded the loopkup method and its test. Explained the logic of coded methods in the README file.

Submitting the code with 505 progress. Copied the development log written in the README file to LogBook

Jan 25 2022 - Coded the .equals, remove, and removeDuplicate methods and created tests for the methods. Added methods to README file. Updated the logbook. Submitted the project with 90% progress.

Jan 26 2022 - Converted .txt to pdf. Refined the methods. Added exceptions. Fixed last space bug in toString

Jan 27 2022 - Submitted the project

How to use the Project:

This project is an alternate implementation of the Java API ArrayList. So, it can be used just like the ArrayList. It has some methods that Java API ArrayList doesn't have and Java API ArrayList has many methods that this Arraylist doesn't have.

Methods in this ArrayList:

- size:
- setCapacity:
- capacity:
- isEmpty:
- add:
- insert:
- lookup:
- remove:
- removeDuplicates:
- contains:
- printArray:
- equals:
- toString:
- main:

I'll be adding more methods in this ArrayList soon. Encountered a bug? email: *pxb410@case.edu*

UPDATE 27th Jan 2022: Several exceptions are thrown when appropriate. Main method is complete, Test & methods are modified and the last space problem in toString is fixed.

Credits:

Michael S Lewicki(professor)

Parv Bhardwaj(self)