

CSE222 – Homework (3) – Question1
Report

Nevzat Seferoglu
171044024

Problem Solution Approach

Problem Description:

Implement a `LinkedListArray` class which implements `List` interface and extends `AbstractList` class. Your class should keep a linked list where each node contains an array of elements with constant capacity. These arrays should be partially filled arrays. Arrays in the nodes of the linked list, may contain different number of elements. When you remove an element in the list, you should find the array containing the elements, first. During remove operation, you should shift the elements (coming after the removed element) in that array only. The other arrays should not be affected. Of course, if the number of elements in an array becomes zero, the node containing this array should also be removed from the linked list. When you add a new element, you need to add the elements into the corresponding array. If the capacity of the array is exhausted, you should create a new node (containing an empty array) in the linked list instead of incrementing the size of the array (you are not allowed to reallocate). Of course, you may want to move some of the elements in the exhausted array into the new array. You are not allowed to use any class in `Collection` hierarchy as a member. So, you need to define your own `Node` class to build a linked list and use basic array structure in Java. Note that you need to implement all required methods and `ListIterator` class.

Approach:

There would be a list that is designed node relating array allocation. There is also a **listiterator** that I need to implement. There would be node which has java array with constant size. Node has reference to next and previous of itself. Thanks to the references I could move throughout the node cluster which points each other. There should have been a head reference to handle those also.

There are different types of inner class in java. Every single version adds some feature to java environment such as shadowing. Shadowing is feature that you can declare exactly same type and name, field in nested class, that field will be handled in its own scope.

Why I used nested class?

Node class is only related to my outer class, there will be no class that is concerning Node class. I also increase the encapsulation level which is directly supported by OOP. Thanks to encapsulation code will be more maintainable and readable. I used nested class as static nested class because there is no need that Node class should reach its own outer class. Static provide that property.

There are several different nested classes also;

Inner Class -> That is associated with an instance of its enclosing class and has direct access to that object's methods and fields. That does not work for me because I do not any direct access to my outer class.

- **Local Class**
- **Anonymous Class**

These are two different type of inner classes also.


Note: [Exception cases explained in Javadoc output of the project.](#)

Result:








At the end, we get a `LinkedListArray` collection that works exactly as same as `LinkedList`. There is no changing for the customer of the collection.

Test Cases

'LinkedList', special constructor that takes 'java.util. collection', testing.

| TEST-ID | SCENERIO | TEST DATA | Expected Result | Actual Result | Pass | Fail |
|---------|---|---|---|---------------------------|---|------|
| T01 | 'LinkedList' have been initialized with constructor that takes collection as an argument. | There is LinkedList which is already filled with some elements. | All elements should be copied to 'LinkedList' | Expected result occurred. |  | |

'LinkedList', non-iterator methods testing.

| TEST-ID | SCENERIO | TEST DATA | Expected Result | Actual Result | Pass | Fail |
|---------|---|--------------------|---|---------------------------|---|------|
| T01 | Creating with no parameter constructor. | | 'LinkedList' would be initialized. | Expected result occurred. |  | |
| T02 | -add (E element) E1, E2, E3, E4 will be added. | E1, E2, E3, E4 | Elements would be added to the list respectively. | Expected result occurred. |  | |
| T03 | -get (int index) Getting 0 th element from the list. | E1, E2, E3, E4 | E1 would be returned. | Expected result occurred. |  | |
| T04 | -get (int index) Getting element in (length -1) index. | E1, E2, E3, E4 | E4 would be returned. | Expected result occurred. |  | |
| T05 | -set (int index, element) Setting 0 th element to E5 | E1, E2, E3, E4, E5 | E1 would be E5. | Expected result occurred. |  | |
| T06 | -set (int index, element) Setting index less than zero or larger or equal than size of the list. | | Throws, 'IndexOutOfBoundsException' | Expected result occurred. |  | |
| T06 | -remove(element) Removing E2 from the list. | E5, E2, E3, E4 | E2 would be removed. | Expected result occurred. |  | |

| | | | | | | |
|-----|---|--------------------|---|---------------------------|---|--|
| T07 | - indexOf (element) Getting index of E4. | E5, E3, E4 | Return index of E4 as 2. | Expected result occurred. | ✓ | |
| T08 | -add (int index, element) Adding E6 to 1th index of the list. | E5, E6, E3, E4 | E6 would be added to specified position. | Expected result occurred. | ✓ | |
| T09 | -remove (int index) Remove the 1th index of the list. | E5, E6, E3, E4 | E6 would be deleted from the list. | Expected result occurred. | ✓ | |
| T10 | -add (element) Adding E3 to the list. | E5, E3, E4, E3 | E3 would be added at the end of the list. | Expected result occurred. | ✓ | |
| T11 | -add (int index, element) Adding E7 to 0 th index of the list. | E7, E5, E3, E4, E3 | E7 would be added at the beginning of the list. | Expected result occurred. | ✓ | |
| T12 | -add (int index, element) Adding element to less than 0 th or larger than size index. | | Throws, 'IndexOutOfBoundsException' | Expected result occurred. | ✓ | |
| T13 | -lastIndexOf (element) Returning the last occurrence of E3. | E7, E5, E3, E4, E3 | Return 4 as a last occurrence index of E3 | Expected result occurred. | ✓ | |
| T14 | -size () Return the current size of the list. | | Return 5 would be a size. | Expected result occurred. | ✓ | |
| T14 | -clear () Clear the content of list. | | Clear all content of list. | Expected result occurred. | ✓ | |

'LinkedList', listIterator functionality testing.

| TEST-ID | SCENERIO | TEST DATA | Expected Result | Actual Result | Pass | Fail |
|---------|---|-----------|------------------------------------|---------------------------|------|------|
| T01 | Creating with no parameter constructor. | | 'LinkedList' would be initialized. | Expected result occurred. | ✓ | |
| T02 | -listIterator (); ListIterator has been Initialized with 0 th position. | | Initialized successfully. | Expected result occurred. | ✓ | |

| | | | | | | |
|-----|---|---|--|---------------------------|---|--|
| T03 | -add (element) Adding 5 different element to list respectively. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 5 | Iterator position is at 5. | Expected result occurred. | ✓ | |
| T04 | -hasNext (); Query the iterator whether it has element next to it. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 5 | There is no more element, return false. | Expected result occurred. | ✓ | |
| T05 | -hasPrevious (); Query the iterator whether it has element before to it. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 5 | There is an element, return true. | Expected result occurred. | ✓ | |
| T06 | -next (); Return the next element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 5 | If there is no element to the next of the iterator, Throws, NoSuchElementException | Expected result occurred. | ✓ | |
| T07 | -previous (); Return the previous element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 4 | There is an element, so it returns that. Returned Element = C5 | Expected result occurred. | ✓ | |
| T08 | -previous (); Return the previous element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 3 | There is an element, so it returns that. Returned Element = C4 | Expected result occurred. | ✓ | |
| T09 | -previous (); Return the previous element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 2 | There is an element, so it returns that. Returned Element = C3 | Expected result occurred. | ✓ | |
| T10 | -previous (); Return the previous element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 1 | There is an element, so it returns that. Returned Element = C2 | Expected result occurred. | ✓ | |
| T11 | -previous (); Return the previous element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 0 | There is an element, so it returns that. Returned Element = C1 | Expected result occurred. | ✓ | |
| T11 | -previous (); Return the previous element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 0 | If there is no element to the before the iterator, Throws, NoSuchElementException | Expected result occurred. | ✓ | |
| T12 | -hasPrevious (); Query the iterator whether it has element before to it. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 0 | There is no more element, return false. | Expected result occurred. | ✓ | |
| T13 | -next (); Return the next element of iterator. | C1 C2 C3 C4 C5 0 1 2 3 4 5 Current Position: 1 | There is an element, so it returns that. Returned Element = C1 | Expected result occurred. | ✓ | |
| T14 | -add (element) Adding C6 to list. | C1 C6 C2 C3 C4 C5 0 1 2 3 4 5 6 Current Position: 2 | Iterator position is at 2. | Expected result occurred. | ✓ | |

| | | | | | | |
|-----|---|---|---|---------------------------|---|--|
| T15 | -next (); Return the next element of iterator. | <div> <div>C1 C6 C2 C3 C4 C5 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 3</p> | There is an element, so it returns that. Returned Element = C2 | Expected result occurred. | ✓ | |
| T16 | -next (); Return the next element of iterator. | <div> <div>C1 C6 C2 C3 C4 C5 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 4</p> | There is an element, so it returns that. Returned Element = C3 | Expected result occurred. | ✓ | |
| T17 | -add (element) Adding C7 to list. | <div> <div>C1 C6 C2 C3 C7 C4 C5 </div> <div>0 1 2 3 4 5 6 7</div> </div> <p>Current Position: 5</p> | Iterator position is at 5. | Expected result occurred. | ✓ | |
| T18 | -next (); Return the next element of iterator. | <div> <div>C1 C6 C2 C3 C7 C4 C5 </div> <div>0 1 2 3 4 5 6 7</div> </div> <p>Current Position: 6</p> | There is an element, so it returns that. Returned Element = C4 | Expected result occurred. | ✓ | |
| T19 | -add (element) Adding C8 to list. | <div> <div>C1 C6 C2 C3 C7 C4 C8 C5 </div> <div>0 1 2 3 4 5 6 7 8</div> </div> <p>Current Position: 7</p> | Iterator position is at 7. | Expected result occurred. | ✓ | |
| T20 | -previous (); Return the previous element of iterator. | <div> <div>C1 C6 C2 C3 C7 C4 C8 C5 </div> <div>0 1 2 3 4 5 6 7 8</div> </div> <p>Current Position: 6</p> | There is an element, so it returns that. Returned Element = C8 | Expected result occurred. | ✓ | |
| T21 | -remove (); Remove the last returned element by the next or previous. | <div> <div>C1 C6 C2 C3 C7 C4 C5 </div> <div>0 1 2 3 4 5 6 7</div> </div> <p>Current Position: 6</p> | C8 has been removed from the list. | Expected result occurred. | ✓ | |
| T22 | -next (); Return the next element of iterator. | <div> <div>C1 C6 C2 C3 C7 C4 C5 </div> <div>0 1 2 3 4 5 6 7</div> </div> <p>Current Position: 7</p> | There is an element, so it returns that. Returned Element = C5 | Expected result occurred. | ✓ | |
| T23 | -remove (); Remove the last returned element by the next or previous. | <div> <div>C1 C6 C2 C3 C7 C4 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 6</p> | C5 has been removed from the list. | Expected result occurred. | ✓ | |
| T24 | -previous (); Return the previous element of iterator. | <div> <div>C1 C6 C2 C3 C7 C4 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 5</p> | There is an element, so it returns that. Returned Element = C4 | Expected result occurred. | ✓ | |
| T25 | -previous (); Return the previous element of iterator. | <div> <div>C1 C6 C2 C3 C7 C4 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 4</p> | There is an element, so it returns that. Returned Element = C7 | Expected result occurred. | ✓ | |
| T26 | -nextIndex(); Return the next returned element index if there is. | <div> <div>C1 C6 C2 C3 C7 C4 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 4</p> | Return 4 as an output. | Expected result occurred. | ✓ | |
| T27 | -previousIndex (); Return the next returned element index if there is. | <div> <div>C1 C6 C2 C3 C7 C4 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 4</p> | Return 3 as an output. | Expected result occurred. | ✓ | |
| T28 | -set (element); Set the last return element with C9 | <div> <div>C1 C6 C2 C3 C9 C4 </div> <div>0 1 2 3 4 5 6</div> </div> <p>Current Position: 4</p> | There is an element which is returned. It changed from C7 to C9. | Expected result occurred. | ✓ | |

Class Diagram:

*Attached to the homework file as a png. *

Running command and results:

*Attached to the homework file as a txt. *