GEBZE TECHNICAL UNIVERSITY

DATA SCTRUCTURE AND ALGORITHM

HOMEWORK -3 REPORT

NAME: SUHA BERK KUKUK

STUDENT ID: 225008003102

1. Run Program and results:
   1. Basic Array Test 1

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated with medium confidence

* 1. Basic Array Test 2

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

* 1. Basic Array Test 3

Text

Description automatically generated

* 1. Array List Structure Test 1

Text

Description automatically generated

Text

Description automatically generated

A picture containing text

Description automatically generated

* 1. Array List Structure Test 2

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

* 1. Array List Structure Test 3

Text

Description automatically generated

* 1. Array List Structure Test 4

Text

Description automatically generated

Text

Description automatically generated

* 1. LinkedList Structure Test 1

Text

Description automatically generated

Text

Description automatically generated

* 1. LinkedList Structure Test 2

Text

Description automatically generated

Text

Description automatically generated

* 1. LinkedList Structure Test 3

Text

Description automatically generated

* 1. LinkedList Structure Test 4

Text

Description automatically generated

Text

Description automatically generated

* 1. LDLinkedList Structure Test 1

Text

Description automatically generated

Text

Description automatically generated

* 1. LDLinkedList Structure Test 2

Text

Description automatically generated

Text

Description automatically generated

* 1. LDLinkedList Structure Test 3

Text

Description automatically generated

* 1. LDLinkedList Structure Test 4

Text

Description automatically generated

Text

Description automatically generated

1. Time Complexity Analysis:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Structures | Time Complexity | | | |
| Basic Array Structure | Array List Structure | Linked List Structure | LD Linked List Structure |
| Methods |
| followCheck | O(n) | O(n) | O(n) | LinkedList Structure Test 4 |
| checkBlock | O(n) | O(n) | O(n) | O(n) |
| follow | O(4) | O(6) | O(6) | O(n) |
| blockAccount | O(1) | O(1) | O(1) | O(6) |
| sendMessage | O(n+7) | O(2n+5) | O(2n+5) | O(1) |
| checkInbox | O(n + 4) | O(2) | O(2) | O(2n+5) |
| checkOutbox | O(n + 4) | O(3) | O(3) | O(2) |
| sharingPost | O(6) | O(4) | O(4) | O(3) |
| likePost | O(4) | O(3) | O(n+2) | O(4) |
| commentPost | O(5) | O(4) | O(n+3) | O(n+2) |
| viewIntereactions | O(2) | O(2) | O(2) | O(n+3) |
| viewPost | O(n) | O(n) | O(n) | O(2) |
| viewInbox | O(n) | O(n) | O(n) | O(n) |
| viewProfile | O(3n) | O(3n) | O(2) | O(n) |
| unfollow | Not available | O(n+3) | O(n+3) | O(2) |
| unblock | O(2) | O(2) | O(n+3) |
| unlikePost | O(n+4) | O(2+n) | O(2) |
| uncommentPost | O(n+4) | O(2+n) | O(2+n) |
| viewHistory | O(n+1) | O() | O(2+n) |

1. Experimental Running time analysis:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Implementation Type | Scenario 1 | Scenario 2 | Scenario 3 | Scenario 4 |
| Basic Array Structure | 0.045 second | 0.044 second | 0.048 second | Not available |
| Array List Structure | 0.049 second | 0.048 second | 0.00 second | **0.052 second** |
| Linked List Structure | 0.047 second | **0.049 second** | 0.00 second | 0.049 second |
| LD Linked List Structure | **0.05 second** | 0.048 second | 0.00 second | 0.05 second |

When we work basic array and array list, time complexity is not too complex as well we work linkedList and LDlinkedList. Basically, it depends on the node and linked structure. Normally we just add element information we don’t keep which element come first and then continue to another element. So, arrive to specific element in this list, the time decreasing. Because we can know where element where it in list is. We don’t need to traverse all the list. As you can see table time complexity. If the method consists of get method (follow, unfollow, viewMessage etc), time complexity is increasing. We can see same result in time analysis. But in scenario 3, the time is close to zero. I think that it causes view profile which blocked by profile. That’s why the code run fast.

1. Problem Solution Approach for LDlinkedlist:

In solving this problem to create LDlinkedlist from Abstractlist. There are methods that use for arriving element, remove element, get size of list etc. In this list we should overriding. So respectively created methods as like below.

LDlinkedlist should consist of get, size, move and add. The list type is linked so create nodes for link each of element. The nodes consist of data, isDeleted information (I will mention) and next linked information node. isDeleted is boolen value for checking the node is deleted. If the node deleted sets True. When we need to physically remove the lazily deleted nodes, we check traverse the list and count of isDeleted as True. If the counter was bigger than 2, we deleted nodes physically. And then we change the next information on previous and later nodes. Overall, the LDLinkedList implementation will have the same behavior as a regular linked list, but with the added feature of lazy deletion.