CSE222 – Homework (3) - Question2 Report

NEVZAT SEFEROGLU

171044024

Problem Solution Approach:

Problem Definition:

The problem was creating a simple text editor. Editors have taken part all over our lives recently. There are so many kinds of editors and each editor has some properties which can distinguish from others. In this project our editors have some functionalities which are like popular ones.

Desired Functionality:

- Reading:
  *Read the file which is given with filepath. *
  - With pure collection functionality except iterator feature.
  - With pure collection functionality with **iterator** feature.
- Adding:
  *Add the given string to desired location as index. *
  - With pure collection functionality except iterator feature.
  - With pure collection functionality with **iterator** feature.
- Finding:
  *Find the given string, return the first occurrence of index in collection. *
  - With pure collection functionality except iterator feature.
  - With pure collection functionality with **iterator** feature.
- Replacing:
  Replace the given character with given as new one
  - With pure collection functionality except iterator feature.
  - With pure collection functionality with **iterator** feature.

Approach:

There were two different implementations for each function, I thought need to implement in a different view because there are two unique collection features in our hand. These two methods behave differently to their abstracted data type. Iteration has a functionality which provides us to getting the element which has index easily. I took an advantage of features of these different programming concept.

*Note: Exceptions have been already clarified and explained in Javadoc of the project file. *

Test Cases:

Test ID	Scenario	Test Data	Expected Result	Actual Result	Pass	Fail
T01	Initializing editor class		Initialized editor class will be ready	Expected result occurred.	✓	
T02	-read (String filepath) read (filepath)	filepath	File will be read.	Expected result occurred.	✓	
T03	-readWithIterator (String filepath) readWithIterator(filepath)	filepath	File will be read by the iterator.	Expected result occurred.	✓	
T04	-add (int, String) add (5, GTU) Added to the given index.	5, GTU	At index 5 if there is, GTU string will be inserted.	Expected result occurred.	✓	
T05	-addWithIterator (int, String) addWithIterator (5, GTU) Added to the given index.	18, CSE	At index 18 if there is, CSE string will be inserted by the iterator.	Expected result occurred.	✓	
T06	-find (String filepath) -find (GTU) Find the beginning index of GTU if there is.	GTU	If there is a string which matches GTU, return the index of it, otherwise return -1	Expected result occurred.	✓	
T07	-findWithIterator (String filepath) - findWithIterator (CSE) Find the beginning index of CSE if there is.	CSE	If there is a string which matches CSE, return the index of it, otherwise return -1 by the iterator.	Expected result occurred.	✓	
T08	-replace (Character 1, Character 1) -replace (all number occurrence with, _) Replay the all occurrence of given character with new one.	_	All number occurrence in the internal text will be _.	Expected result occurred.	✓	
T09	-replaceWithIterator (Character 1, Character 1) replaceWithIterator (all number occurrence with, _) Replay the all occurrence of given character with new one.	_	All number occurrence in the internal text will be _.	Expected result occurred.	✓	

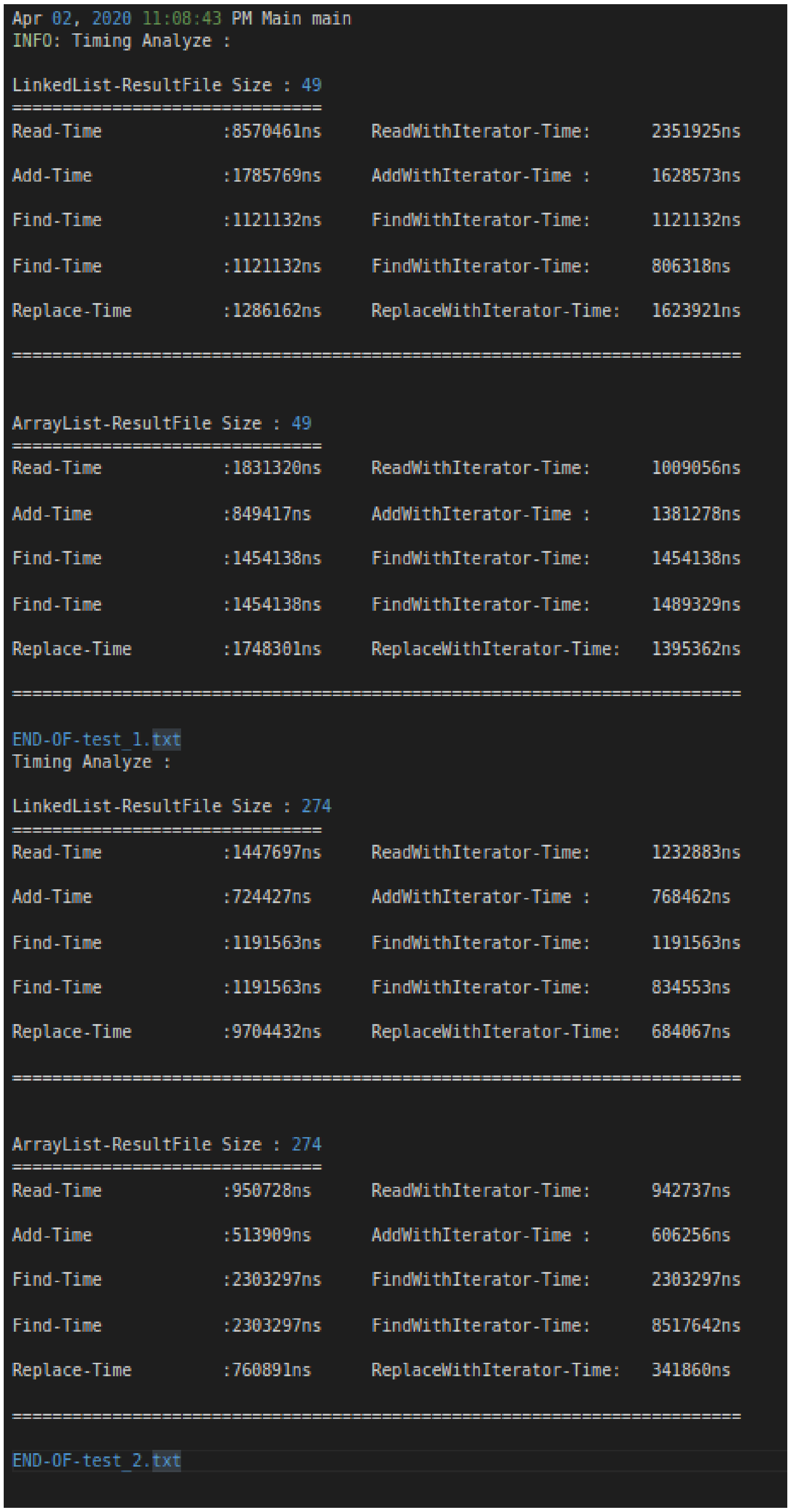
*Note: This test is valid for both LinkedList and ArrayList. *

Theoretical Analysis:

Method	LinkedList	ArrayList
read ()	This method read the given file. In its internal implementation, there is first degree for loop and in this for loop, add method which has O (1) big(o) time time-complexity. Total time-complexity will be O(n)	This method read the given file. In its internal implementation, there is first degree for loop and in this for loop, add method which has O (N) big(o) time time-complexity. Total time-complexity will be $O(n^2)$
readWithIterator ()	There is listIterator method which is add that has O (1) big o complexity. There is also loop for going all over the nodes which is O(n). Total time-complexity should be O(n).	There is listIterator so there is no constant time and node implementation also, add work in a linear time-complexity which is O(n). There is also for loop again and total complexity is $O(n^2)$
add ()	This method adds the string to specific position so, there would be a string which but there is another add method in it which takes index and character, string length is determinant. LinkedList add method time complexity O(L) L is length of our string. There is a for loop again which depends on index value, let say N for that. Total complexity will be O(NL).	This method adds the string to specific position so, there would be a string which but there is another add method in it which takes index and character, string length is determinant. ArrayList add method time complexity O(L) L is length of our string. There is a for loop again which depends on index value, let say N for that. Total complexity will be O(NL).
addWithIterator ()	There is an iteration add method which works in constant time that is O (1). There is also a loop which depends on length of the string so total time-complexity is O(N).	There is no constant-time complexity because there is no iteration and nodes. Resizing and shifting can occur so total time-complexity will be $O(N^2)$

find	This method finds the string in text. There are different purposed if statement in my algorithm. Bu in the worst case, string would not exist in the list and there are two different for loop inside one another. And there is get method of linked list which is work constant time O (1), For loop size would be different so that total complexity will be O(NL)	This method finds the string in text. There are different purposed if statement in my algorithm. Bu in the worst case, string would not exist in the list and there are two different for loop inside one another. And there is get method of array list which works in linear time O (N), For loop size would be different so that total complexity will be O(N^2L) , N^2 come from outer for loop and also from get().L is an another size of inner for loop.
findWithIterator ()	There are several iterator functions in this implementation and all of them works constant time, as it occurs before, there should be two different size for loop which inside one another one so total time-complexity will be O(NL)	There are several iterator functions in this implementation and all of them works constant time, as it occurs before, there should be two different size for loop which inside one another one so total time-complexity will be O(NL)
replace	There is loop again. Inside the loop , in worst case all character would be changed , LinkedList get and set method will be used and each method works in constant time so there 2 different constant time and outer of all of them , there is an for loop .All time complexity will be O(N).	There is loop again. Inside the loop , in worst case all character would be changed , ArrayList get and set method will be used and each method works in linear time so there 2 different linear time which are not inside by inside and outer of all of them , there is an for loop .All time complexity will be O(N^2).
replaceWithIterator ()	There is only just for loop, inside for loop all operations are in constant time. Total complexity O(N).	There is only just for loop, inside for loop all operations are in constant time. Total complexity O(N).

Experimental Result:



```
Apr 02, 2020 11:08:43 PM Main main
INFO: Timing Analyze :

LinkedList-ResultFile Size : 49
===============================
Read-Time            :8570461ns     ReadWithIterator-Time:      2351925ns

Add-Time             :1785769ns     AddWithIterator-Time :      1628573ns

Find-Time            :1121132ns     FindWithIterator-Time:      1121132ns

Find-Time            :1121132ns     FindWithIterator-Time:      806318ns

Replace-Time         :1286162ns     ReplaceWithIterator-Time:   1623921ns

=========================================================================


ArrayList-ResultFile Size : 49
===============================
Read-Time            :1831320ns     ReadWithIterator-Time:      1009056ns

Add-Time             :849417ns      AddWithIterator-Time :      1381278ns

Find-Time            :1454138ns     FindWithIterator-Time:      1454138ns

Find-Time            :1454138ns     FindWithIterator-Time:      1489329ns

Replace-Time         :1748301ns     ReplaceWithIterator-Time:   1395362ns

=========================================================================

END-OF-test_1.txt
Timing Analyze :

LinkedList-ResultFile Size : 274
===============================
Read-Time            :1447697ns     ReadWithIterator-Time:      1232883ns

Add-Time             :724427ns      AddWithIterator-Time :      768462ns

Find-Time            :1191563ns     FindWithIterator-Time:      1191563ns

Find-Time            :1191563ns     FindWithIterator-Time:      834553ns

Replace-Time         :9704432ns     ReplaceWithIterator-Time:   684067ns

=========================================================================


ArrayList-ResultFile Size : 274
===============================
Read-Time            :950728ns      ReadWithIterator-Time:      942737ns

Add-Time             :513909ns      AddWithIterator-Time :      606256ns

Find-Time            :2303297ns     FindWithIterator-Time:      2303297ns

Find-Time            :2303297ns     FindWithIterator-Time:      8517642ns

Replace-Time         :760891ns      ReplaceWithIterator-Time:   341860ns

=========================================================================

END-OF-test_2.txt
```

Class Diagram:

*Class diagram is inserted to the homework file.

Running commend and result:

***Running command and result is inserted to the homework file.