Yunus Emre Avcı -111044040

CSE 222 - Data Structures HW05 Report

Part - 1 : Towers Of Hanoi iteratively

- Towers of hanoi iterative olarak implement edilmiştir.
- http://www.geeksforgeeks.org/iterative-tower-of-hanoi/ adresinden iterative algoritması alınarak implement edilmiştir.
- Adım sayısı 2^disksize -1 olarak bulunarak 1'den 2^disksize -1 değerine kadar her bir değer için mod 3 alınarak hangi peg ikilisinin kullanılacağı belirlenerek problem implement edildi.
- Disk sayısının çift olma durumunda auxaliary ile destination karakterleri swap edilmiştir.

* 3 disk sayısı için:

```
Tower of hanoi for 3 disk
Move Disk 1 from peg S to peg D

Move Disk 2 from peg S to peg A

Move Disk 1 from peg D to peg A

Move Disk 3 from peg S to peg D

Move Disk 1 from peg A to peg S

Move Disk 2 from peg A to peg D

Move Disk 1 from peg S to peg D
```

* 4 disk sayısı için:

```
Tower of hanoi for 4 disk
Move Disk 1 from peg A to peg S

Move Disk 1 from peg S to peg D

Move Disk 1 from peg D to peg A

Move Disk 1 from peg A to peg S

Move Disk 1 from peg S to peg D

Move Disk 1 from peg D to peg A

Move Disk 1 from peg A to peg S

Move Disk 1 from peg S to peg D

Move Disk 1 from peg D to peg A

Move Disk 1 from peg A to peg S

Move Disk 1 from peg S to peg D

Move Disk 1 from peg D to peg A

Move Disk 1 from peg A to peg S

Move Disk 1 from peg S to peg D

Move Disk 1 from peg D to peg A

Tower of hanoi for 5 disk
```

Part - 2 : Remove procedure of LinkedListRec class bulunan tüm duplicate elemanları kaldır.

- Kitapta bulunan LinkedListRec classı alınarak ilaveten add ve toString methodları da recursive olarak implement edilmiştir.

- Integer ve String değerler girilerek methodlar test edilmişir;

```
* Linked list before remove: ali ==> mehmet ==> lale ==> bursa ==> leyla ==> lale ==> il
* Removed: lale
* After removing method: ali ==> mehmet ==> bursa ==> leyla ==> il
* Linked list before remove: 2 ==> 5 ==> 24 ==> 22 ==> 1 ==> 2 ==> 27
* Removed: 2
* Aftre removing method: 5 ==> 24 ==> 22 ==> 1 ==> 27
```

Part - 3 : intersection, union and isSubset

- Class içerisinde tutulan iki sorted list yardımıyla iki listenin intersectionı unionı ve isSubseti recursive olarak implement edilmiştir.

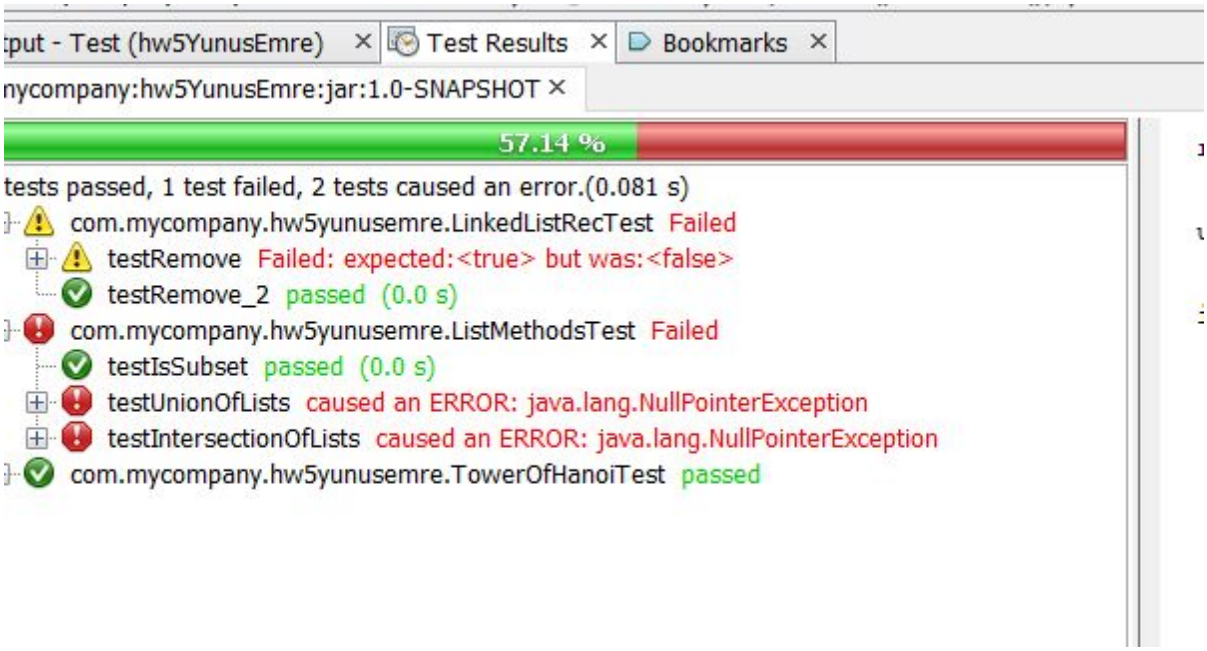
- Union bulurken kod patlamaktadır.

```
------------------------------------------------------------------------

]--- exec-maven-plugin:1.2.1:exec (default-cli) @ hw5YunusEmre ---
~~~~~~> For integer numbers List Methods:
------>List 1: [8, 114, 1, 114, 8, 21]
------>List 2: [1, 11, 1, 114, 71, 114]
------>Intersection of two lists: [114, 1]
------>List 2 Is subset of list 1: true
------>Union of two lists: [8, 1, 114, 11, 1, 114, 8, 71, 21, 114]
-------------------

~~~~~~> For Strings List Methods:
------>List 1: [ahmet, */da, üüasd, izmir, kavak, batık]
------>List 2: [*/, izmir, batıl, art]
------>Intersection of two lists: [izmir]
------>List 2 Is subset of list 1: false
------>Union of two lists: [ahmet, */, */da, izmir, üüasd, batıl, izmir, art]
```

JUnit


put - Test (hw5YunusEmre)
Test Results
Bookmarks
nycompany:hw5YunusEmre:jar:1.0-SNAPSHOT
57.14 %
tests passed, 1 test failed, 2 tests caused an error.(0.081 s)
com.mycompany.hw5yunusemre.LinkedListRecTest Failed
testRemove Failed: expected:<true> but was:<false>
testRemove_2 passed (0.0 s)
com.mycompany.hw5yunusemre.ListMethodsTest Failed
testIsSubset passed (0.0 s)
testUnionOfLists caused an ERROR: java.lang.NullPointerException
testIntersectionOfLists caused an ERROR: java.lang.NullPointerException
com.mycompany.hw5yunusemre.TowerOfHanoiTest passed