GIT Department of Computer Engineering

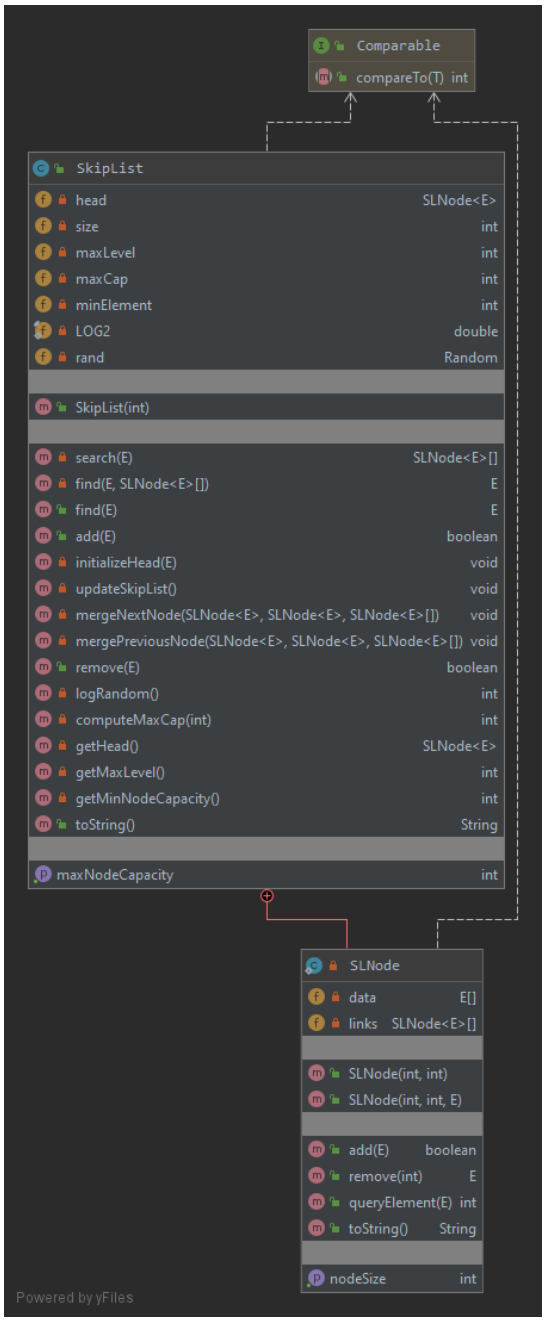
CSE 222/505 – Spring 2020

Homework #07 Part 2 Report

Abdullah ÇELİK

171044002

Class Diagram



Problem Solution Approach

First, the skip list should be sent as the parameter of the maximum number of elements in a node as each node will hold the elements. The minimum number of elements in a node is half the maximum number of elements. In other words, the number of elements in a node is greater than the minimum number of elements and greater than the maximum number of elements in a node. This rule does not apply if there is only one node. If the node exceeds the maximum number of node elements while adding to a node, this node is splitted and a new node will be created next to this node and the elements will be shifted. While doing this, it is necessary to link the newly added node links very carefully. In this way, the structure of the skip list will be preserved. If an element wants to be deleted and the size of the node after deletion is smaller than the minimum number of node elements, if there is a node to the previous of this node, it is merged with previous node. If it does

not exist and there is a next node, it is merged with next node. If there is no node next, this skip list has only one node left and nothing is done. If the size of the merged node is greater than the maximum number of elements after the shifting, that node is splitted again. The last operation is to link the links of the newly added node.

Test Cases

Test ID	Scenerio	Test Data	Expected Results	Actual Results	Pass/Fail
TEST01	Constructor testing when maximum number of elements in a node is lower than 1	max: -5 max: 0	IndexOutOfBoundException to be thrown	As expected	Pass
TEST02	Constructor testing when maximum number of elements in a node is equal or greater than 1	max: 1 max: 4	Successfully created	As expected	Pass
TEST03	boolean add(E item) method called when skip list is empty	item: 10 item: 20 item: 5	Successfully added	As expected	Pass
TEST04	boolean add(E item) method called when node exceeds maximum node capacity	item: 30 item: 40 item: 25 item: 1 item: 3 item: 6 item: 8 item: 7 item: 45	Successfully node was splitted and added item	As expected	Pass
TEST05	boolean add(E item) method called when item is null	item: null	NullPointerException to be thrown	As expected	Pass
TEST06	E find(E target) method called when skip list contains items	item: 5 item: 25 item: 45	Successfully found and returned item	As expected	Pass
TEST07	E find(E target) method called when skip list doesn't contain items	item: 0 item: 50	Successfully returned null	As expected	Pass
TEST08	E find(E target) method called when target is null	item: null	NullPointerException to be thrown	As expected	Pass
TEST09	boolean remove(E item) method called when a node contains greater than minimum elements in a node	item: 10 item: 40	Successfully removed and didn't merge the node	As expected	Pass

TEST10	boolean remove(E item) method called when a node contains equal than minimum elements in a node	item: 1 item: 7 item: 45	Successfully merged nodes and removed item	As expected	Pass
TEST11	boolean remove(E item) method called when item is null	item: null	NullPointerException to be thrown	As expected	Pass

Running and Results

```
TEST01
When maximum number of element in a node is lower than 1
Maximum number of element: -5
Invalid size!
IndexOutOfBoundException was caught!

Maximum number of element: 0
Invalid size!
IndexOutOfBoundException was caught!

TEST02
When maximum number of element in a node is greater or equal than 1
Maximum number of element: 1
Creating an skip list is successful!

Maximum number of element: 4
Creating an skip list is successful!

TEST03
When skip list is empty, method will be called respectively as
        list.add(10),list.add(20),list.add(5)
Before adding, list:
List: Empty
Adding 10: true
Adding 20: true
Adding 5: true
After adding, list:
List: Head: 1-->5,10,20|1|

TEST04
When adding elements to skip list and node exceeds maximum node capacity
Before adding, list:
List: Head: 1-->5,10,20|1|
Adding 30: true
List: Head: 1-->5,10,20,30|1|
Adding 40: true
List: Head: 2-->5,10|1|-->20,30,40|1|
Adding 25: true
List: Head: 2-->5,10|1|-->20,25,30,40|1|
Adding 1: true
List: Head: 2-->1,5,10|1|-->20,25,30,40|1|
```

```
List: Head: 2-->1,5,10|1|-->20,25,30,40|1|
Adding 3: true
List: Head: 2-->1,3,5,10|1|-->20,25,30,40|1|
Adding 6: true
List: Head: 2-->1,3|1|-->5,6,10|1|-->20,25,30,40|1|
Adding 8: true
List: Head: 2-->1,3|1|-->5,6,8,10|1|-->20,25,30,40|1|
Adding 7: true
List: Head: 3-->1,3|1|-->5,6|2|-->7,8,10|1|-->20,25,30,40|1|
Adding 45: true
List: Head: 3-->1,3|1|-->5,6|2|-->7,8,10|1|-->20,25|3|-->30,40,45|1|

TEST05
When adding null element to skip list, method will be called as
        list.add(null)
NullPointerException was caught!

TEST06
When skip list contains item, method will be called as
        list.find(5), list.find(25), list.find(45)
List: Head: 3-->1,3|1|-->5,6|2|-->7,8,10|1|-->20,25|3|-->30,40,45|1|

Find 5: 5
Find 25: 25
Find 45: 45

TEST07
When skip list doesn't contain item, method will be called as
        list.find(0), list.find(50)
List: Head: 3-->1,3|1|-->5,6|2|-->7,8,10|1|-->20,25|3|-->30,40,45|1|

Find 0: null
Find 50: null

TEST08
When searching null element, method will be called as
        list.find(null)
NullPointerException was caught

TEST09
When a node contains greater than minimum element in a node, method will be called as
        list.remove(10), list.remove(40)
Before removing, list:
Head: 3-->1,3|1|-->5,6|2|-->7,8,10|1|-->20,25|3|-->30,40,45|1|

Removing 10: true
After removing, list:
Head: 3-->1,3|1|-->5,6|2|-->7,8|1|-->20,25|3|-->30,40,45|1|

Removing 40: true
After removing, list:
Head: 3-->1,3|1|-->5,6|2|-->7,8|1|-->20,25|3|-->30,45|1|
```

```
TEST10
When a node contains equal than minimum element in a node, method will be called respectively as
        list.remove(1), list.remove(7), list.remove(45)
Before removing, list:
Head: 3-->1,3|1|-->5,6|2|-->7,8|1|-->20,25|3|-->30,45|1|

Removing 1: true
After removing, list:
Head: 3-->3,5,6|2|-->7,8|1|-->20,25|3|-->30,45|1|
Removing 7: true
After removing, list:
Head: 3-->3,5,6,8|2|-->20,25|3|-->30,45|1|
Removing 45: true
After removing, list:
Head: 3-->3,5,6,8|2|-->20,25,30|3|

TEST11
When null elements to be removed, remove method will be called as
        list.remove(null)
NullPointerException was caught!
```