CSE 222 HOMEWORK #5

Emre

BAYRAM

141044019

TABLE OF CONTENTS

1. Requirements

1.1. Overall Description

Part1

Implementing Honoi Tower Problem with iterative thinking.

Part2

Implementing the remove procedure from LinkListRec. This remove method removes all duplicated elements in the linkedlist

Part3

Implementing 3 recursive method . First method is “intersection of List” this method returns Intersection of 2 list while your are creating object that you have used , Second method is “Union Of List ” this method returns a new List of Union of 2 list in the same way you created object. Third method is “is Subset” this method returns true or false whether list1 is subset of list2 is true otherwise false.

1.2. Requirement Definitions

1.2.1. 2 list when using part3 class

1.2.2. When returning a List used ArrayList data structure.

2. Analysis and Solutions Approach

PART 1

I was solved before Honoi tower problem using recursive thinking. I examined again this solution then do some research on the Internet and play the game. Then I develop some algorithms and apply them.

PART 2

Before implementing remove method , wrote a small Linked List data Structure as in course book. Then I examined the normal remove method and I wrote algorithm for remove all duplications and apply to code.

PART 3

I did some research about these methods on Internet about what is expecting from me and I develop algorithm after that apply it I wrote wrapper method for every method.

3. Class Diagrams

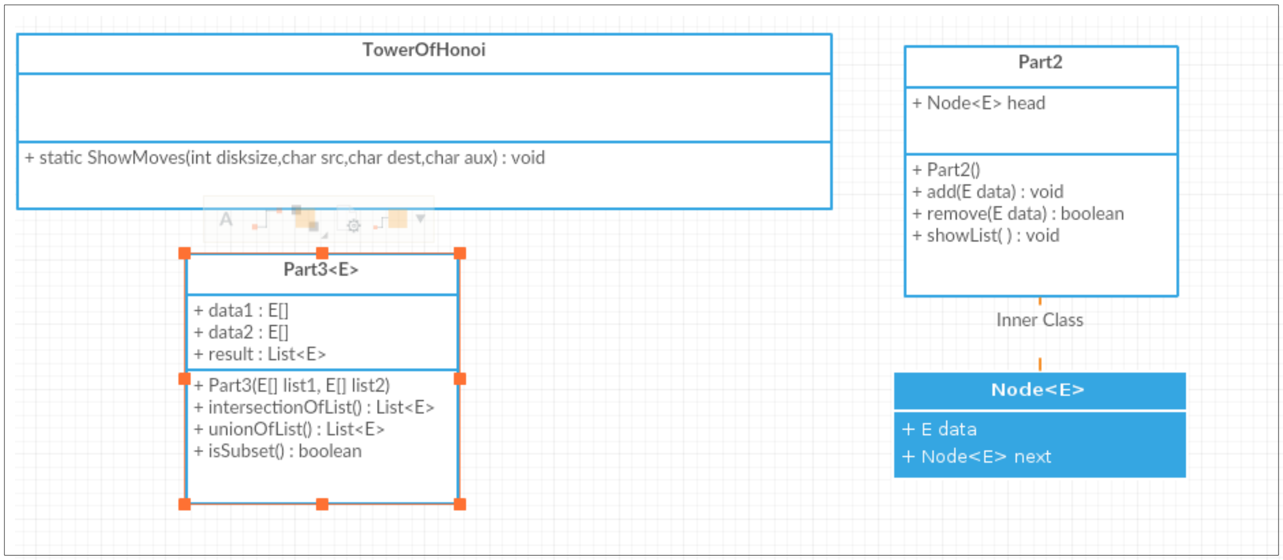
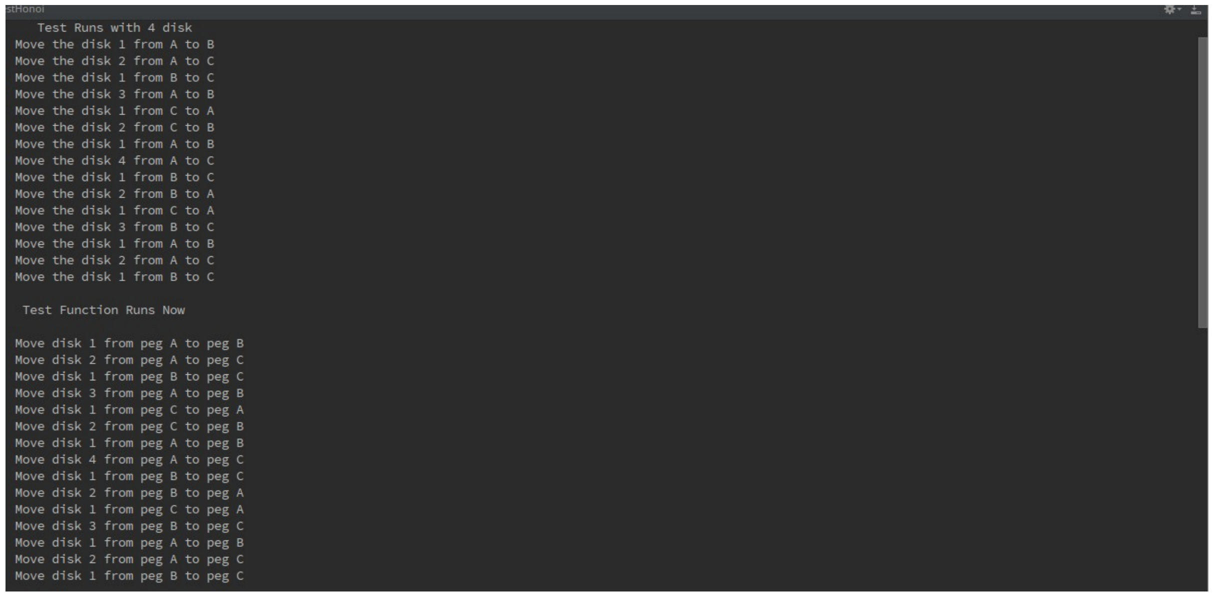


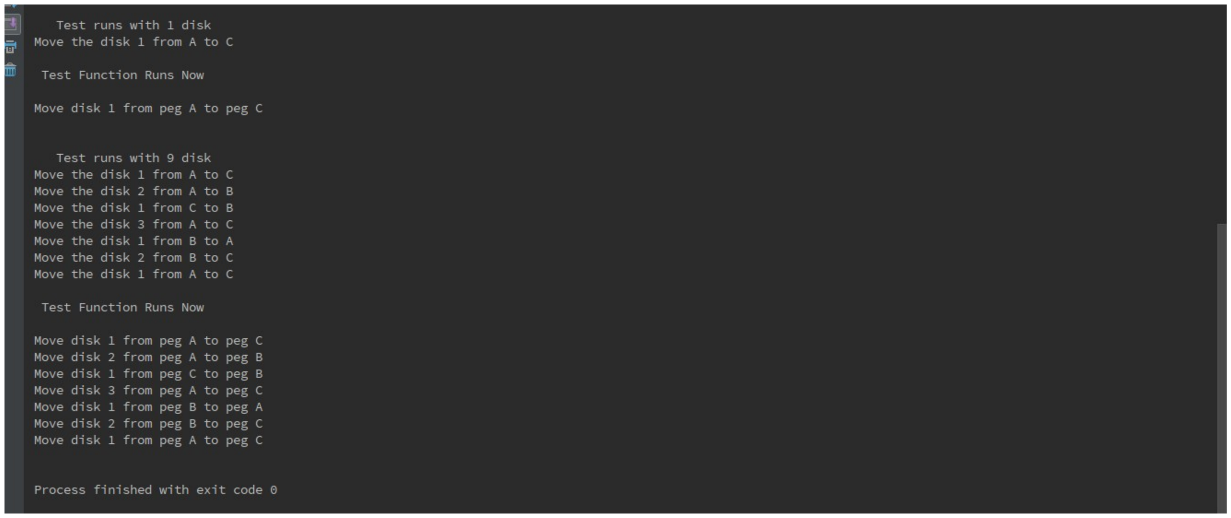
Figure 1 Class Diagram

4. Tests

Tower Of Honoi



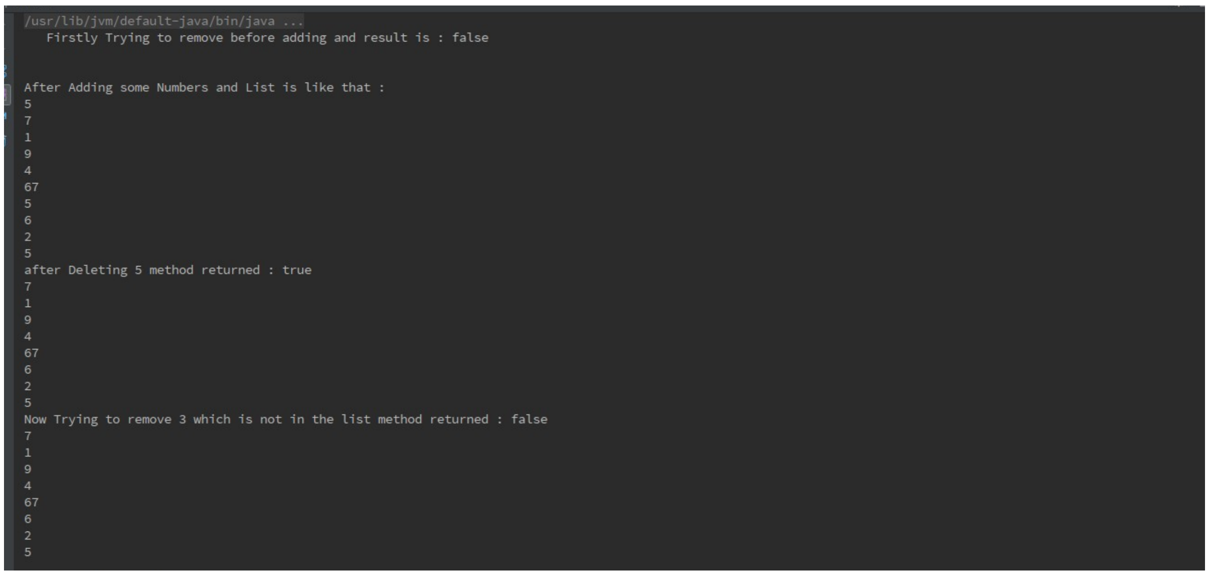
```
   Test Runs with 4 disk
Move the disk 1 from A to B
Move the disk 2 from A to C
Move the disk 1 from B to C
Move the disk 3 from A to B
Move the disk 1 from C to A
Move the disk 2 from C to B
Move the disk 1 from A to B
Move the disk 4 from A to C
Move the disk 1 from B to C
Move the disk 2 from B to A
Move the disk 1 from C to A
Move the disk 3 from B to C
Move the disk 1 from A to B
Move the disk 2 from A to C
Move the disk 1 from B to C

 Test Function Runs Now

Move disk 1 from peg A to peg B
Move disk 2 from peg A to peg C
Move disk 1 from peg B to peg C
Move disk 3 from peg A to peg B
Move disk 1 from peg C to peg A
Move disk 2 from peg C to peg B
Move disk 1 from peg A to peg B
Move disk 4 from peg A to peg C
Move disk 1 from peg B to peg C
Move disk 2 from peg B to peg A
Move disk 1 from peg C to peg A
Move disk 3 from peg B to peg C
Move disk 1 from peg A to peg B
Move disk 2 from peg A to peg C
Move disk 1 from peg B to peg C
```



```
   Test runs with 1 disk
Move the disk 1 from A to C

 Test Function Runs Now

Move disk 1 from peg A to peg C


   Test runs with 9 disk
Move the disk 1 from A to C
Move the disk 2 from A to B
Move the disk 1 from C to B
Move the disk 3 from A to C
Move the disk 1 from B to A
Move the disk 2 from B to C
Move the disk 1 from A to C

 Test Function Runs Now

Move disk 1 from peg A to peg C
Move disk 2 from peg A to peg B
Move disk 1 from peg C to peg B
Move disk 3 from peg A to peg C
Move disk 1 from peg B to peg A
Move disk 2 from peg B to peg C
Move disk 1 from peg A to peg C


Process finished with exit code 0
```

Exactly Same as solution.

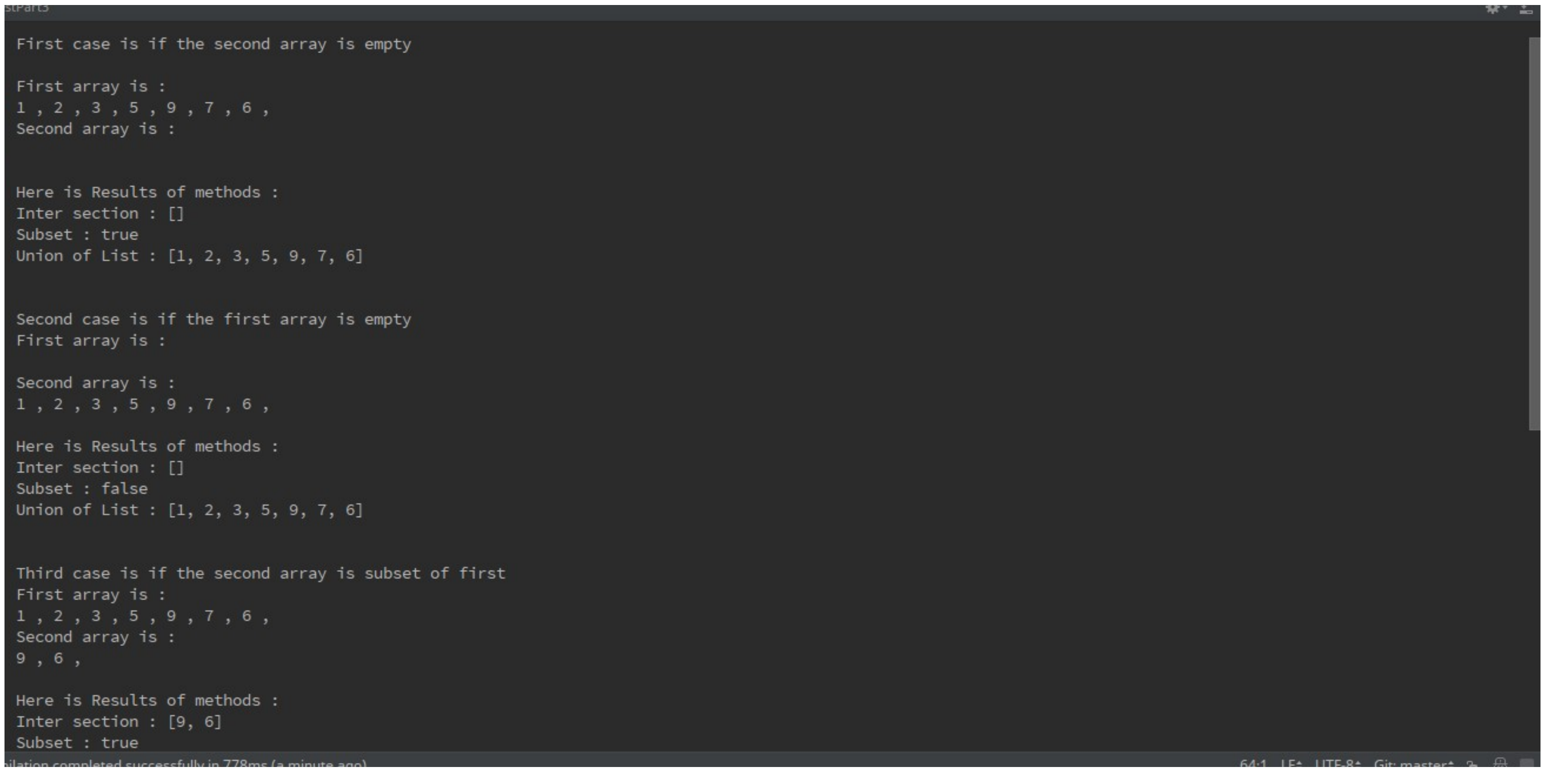
Part 2



```
/usr/lib/jvm/default-java/bin/java ...
   Firstly Trying to remove before adding and result is : false


After Adding some Numbers and List is like that :
5
7
1
9
4
67
5
6
2
5
after Deleting 5 method returned : true
7
1
9
4
67
6
2
5
Now Trying to remove 3 which is not in the list method returned : false
7
1
9
4
67
6
2
5
```

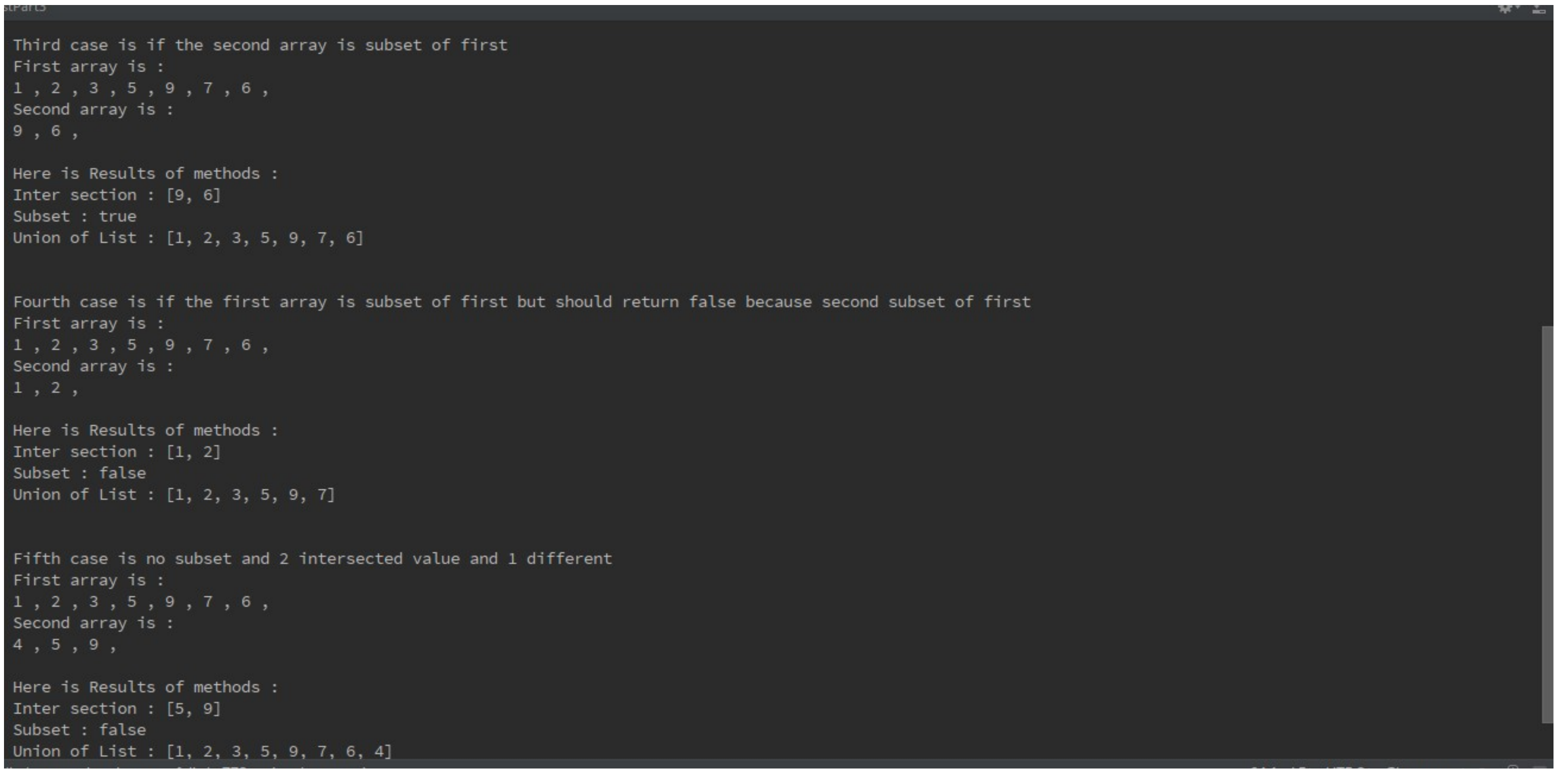
Every case is true.

Part 3



```
First case is if the second array is empty

First array is :
1 , 2 , 3 , 5 , 9 , 7 , 6 ,
Second array is :


Here is Results of methods :
Inter section : []
Subset : true
Union of List : [1, 2, 3, 5, 9, 7, 6]


Second case is if the first array is empty
First array is :

Second array is :
1 , 2 , 3 , 5 , 9 , 7 , 6 ,

Here is Results of methods :
Inter section : []
Subset : false
Union of List : [1, 2, 3, 5, 9, 7, 6]


Third case is if the second array is subset of first
First array is :
1 , 2 , 3 , 5 , 9 , 7 , 6 ,
Second array is :
9 , 6 ,

Here is Results of methods :
Inter section : [9, 6]
Subset : true
```

Every Case is demonstrated with true result all 3 methods work OK.



```
Third case is if the second array is subset of first
First array is :
1 , 2 , 3 , 5 , 9 , 7 , 6 ,
Second array is :
9 , 6 ,

Here is Results of methods :
Inter section : [9, 6]
Subset : true
Union of List : [1, 2, 3, 5, 9, 7, 6]


Fourth case is if the first array is subset of first but should return false because second subset of first
First array is :
1 , 2 , 3 , 5 , 9 , 7 , 6 ,
Second array is :
1 , 2 ,

Here is Results of methods :
Inter section : [1, 2]
Subset : false
Union of List : [1, 2, 3, 5, 9, 7]


Fifth case is no subset and 2 intersected value and 1 different
First array is :
1 , 2 , 3 , 5 , 9 , 7 , 6 ,
Second array is :
4 , 5 , 9 ,

Here is Results of methods :
Inter section : [5, 9]
Subset : false
Union of List : [1, 2, 3, 5, 9, 7, 6, 4]
```

5.Result

As a result, I have challenged writing iterative Honoi Tower problem and Part. Other Parts are not difficult for me.