Gebze Technical University

Computer Engineering

CSE222-2021-SPRING

Homework-4-Report

Part1

Ferdi Sönmez

161044046

1)INTRODUCTİON

1.1)Problem Definition

In this homework , we are asked to implement the heap structure and add some functions and write them. These functions have different requests.

i. ii. iii. iv.	Search for an element Merge with another heap Removing i_{th} largest element from the Heap Extend the Iterator class by adding a method to set the value (value passed as parameter) of the last element returned by the next methods.

1.2- System Requirements

The necessity of a PriorityQueue structure and a Heap structure that includes it emerges from the problem definition. Combining all these components creates a heap structure.

1.2.1-Users of the System

PriorityQueue structure is stored inside the heap structure and all necessary operations are performed using this PriorityQueue structure.

These requirements create a need for a menu.

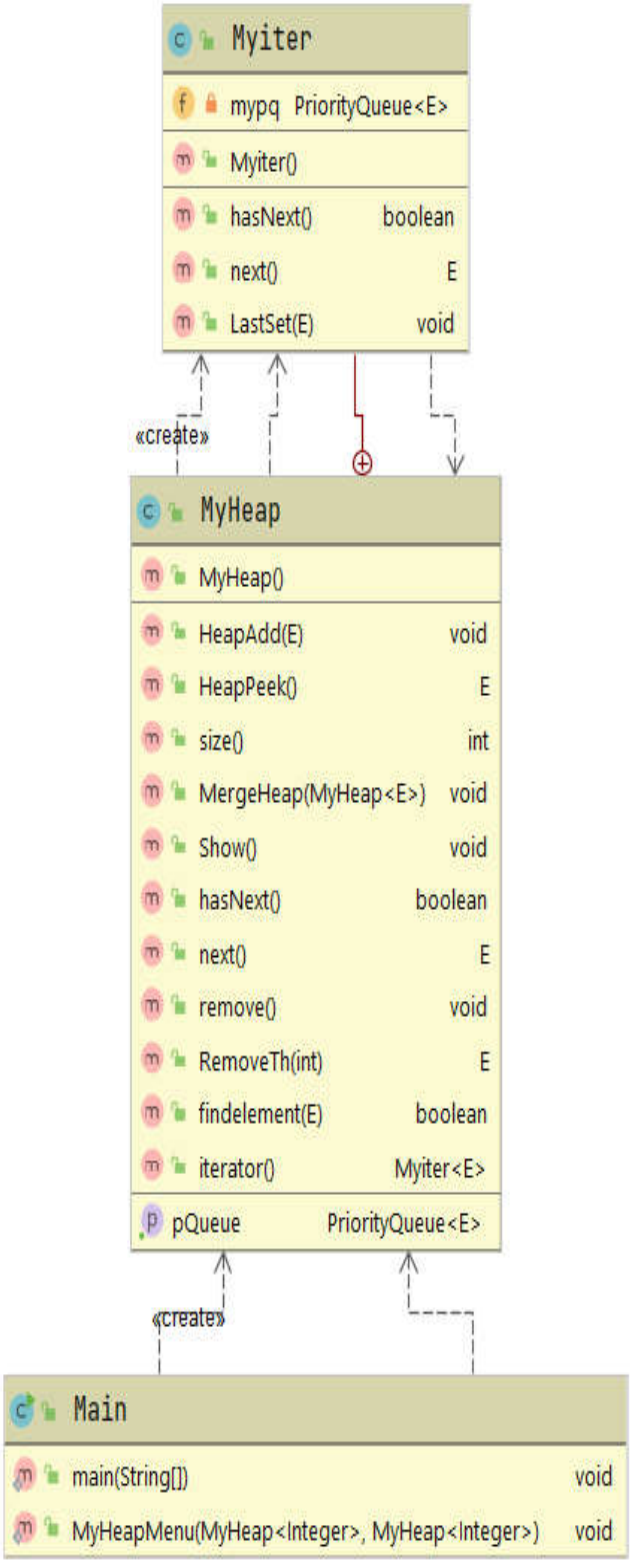
1.2.2- Solution Of The Problem

Keeping the PriorityQueue structure in Myheap, the desired functions are solved in this class.

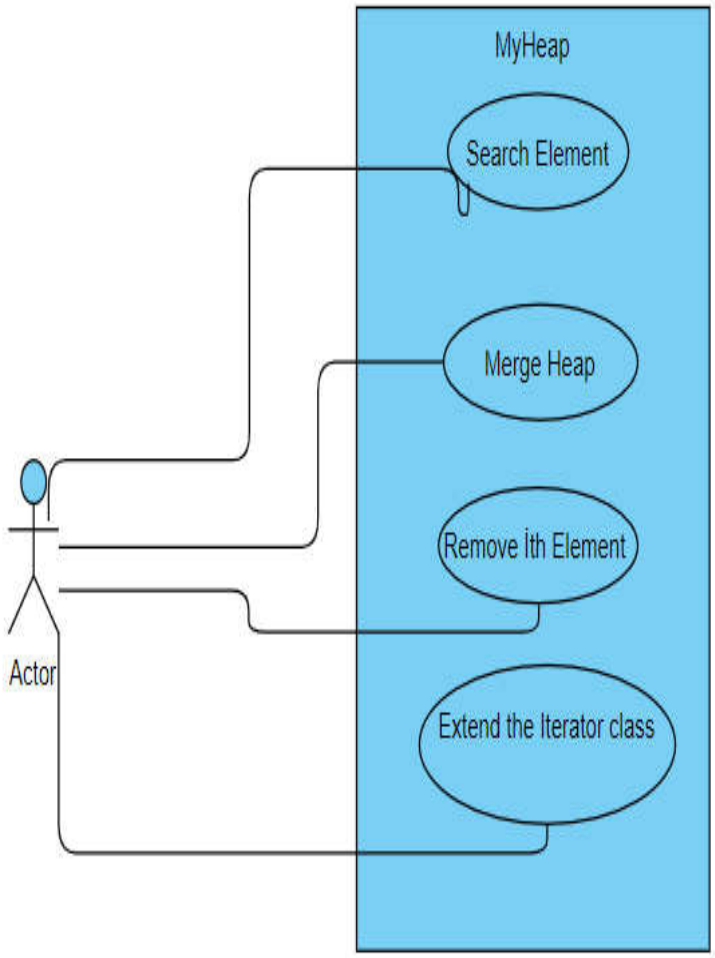
```
public class MyHeap<E> implements Iterator<E> {
    private PriorityQueue<E> pQueue;

    public MyHeap() { pQueue = new PriorityQueue<E>(); }
```

2) Class Diagram



3) Use Case Diagram



4)Test Case

a) The menu designed for this system

```
1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
```

b) Show All Element in Heap

```
1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
5
***Show The All Element***
40 130 150
```

c) Search for an element

```
1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
4
***Search Element***
Enter Element For Search
150
Number is Founded
```

d) Removing ith largest element from the Heap

```
1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
3
***Remove ith Element***
Enter The Number For Removing ith Largest Element
0
150 is Delete

1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
5
***Show The All Element***
20 40 130
```

e)Add Element in Heap.

```
1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
1
***Add Element System***
Enter The Element For Add
20
Number is Added

1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
5
***Show The All Element***
20 40 150 130
```

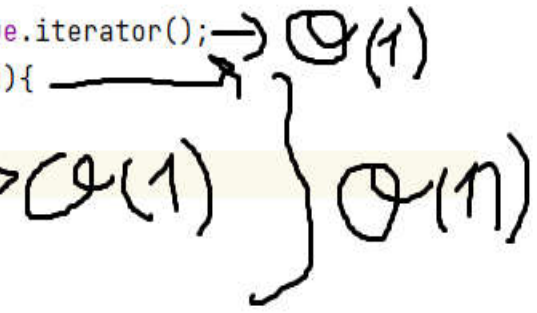
f) Merge with another heap.

```
1-Add Element
2-Merge Another Heap
3-Remove ith Element
4-Search Element
5-Show All Element
6-Exit...
2
***Merge Another Heap Structure***
20 40 130
35 45 55 Do you want to this two heap?(Y/N)
Y
Two Heap Merged
20 35 55 40 45 130
```

Time Complexity:

1)

```
*/
public int size(){
    int count=0;
    Iterator itr = pQueue.iterator();
    while (itr.hasNext()){
        count++;
        itr.next();
    }
    return count;
}
```



$$T(n) = O(n)$$

2)

```
public void MergeHeap(MyHeap<E> temp){
    Iterator itr = temp.getpQueue().iterator();
    while (itr.hasNext()){
        this.pQueue.add((E) itr.next());
    }
}
```

$O(1)$

$O(n)$

$T(n) = O(n)$

3)

```
    */
    public void LastSet(E temp2){
        E temp1=null;
        E y=(E) temp2;
        int i=0;
        Iterator itr = mypq.iterator();
        while (itr.next()!=null){
            temp1= (E) itr.next();
            if (!itr.hasNext()){
                break;
            }
            i++;
        }
        getpQueue().remove(temp1);
        mypq.offer(y);
    }
}
```

$O(1)$

$O(n)$

$O(1)$

$O(\log n)$

$T(n) = O(n)$

4)

```
public E RemoveTh(int index){
    if (index>this.pQueue.size()){
        throw new IndexOutOfBoundsException();
    }
    else {
        ArrayList<E> arrayList = new ArrayList<E>(this.pQueue);
        Collections.sort(arrayList, Collections.reverseOrder());
        this.pQueue.remove(arrayList.get(index));
        E tmp=arrayList.remove(index);

        return tmp;
    }
}
```

$O(1)$

$O(n)$

$O(n)$

$O(n)$

$T(n) = O(n)$