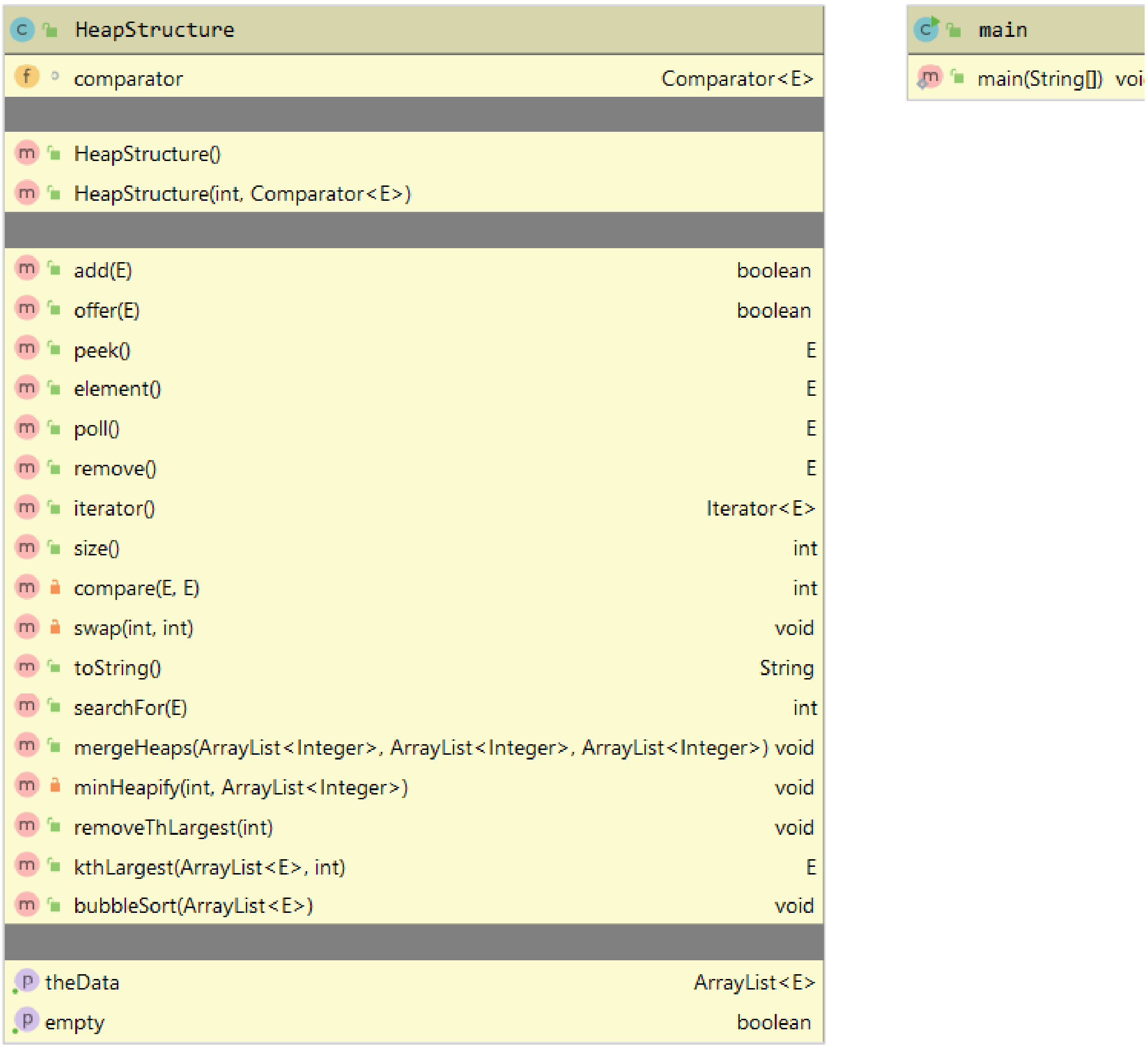
GTU Department of Computer Engineering CSE 222/505 - Spring 2021 Homework 4 Report

Refik Orkun Arslan
151044063

Class Diagram



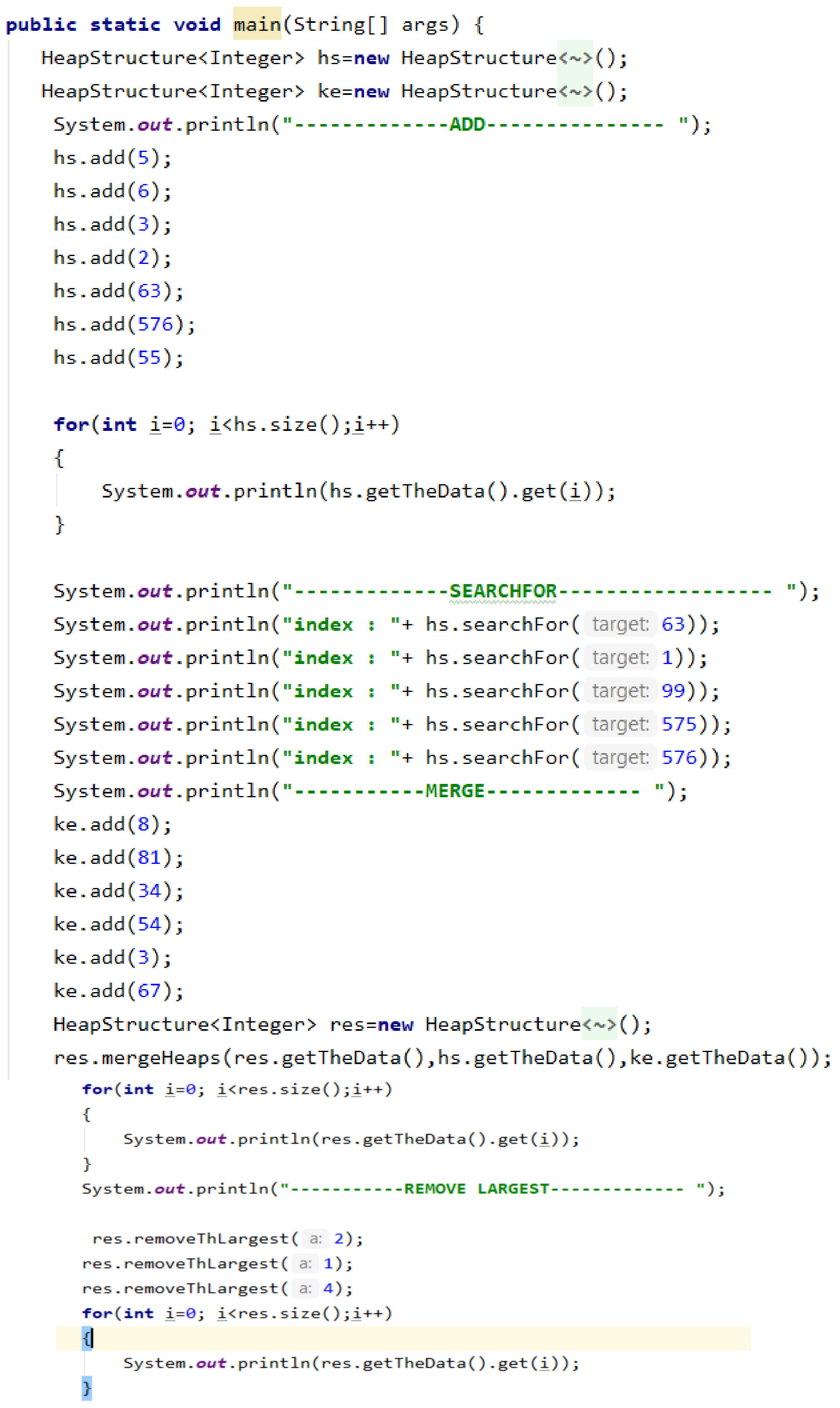
HeapStructure	
comparator	Comparator<E>
HeapStructure()	
HeapStructure(int, Comparator<E>)	
add(E)	boolean
offer(E)	boolean
peek()	E
element()	E
poll()	E
remove()	E
iterator()	Iterator<E>
size()	int
compare(E, E)	int
swap(int, int)	void
toString()	String
searchFor(E)	int
mergeHeaps(ArrayList<Integer>, ArrayList<Integer>, ArrayList<Integer>)	void
minHeapify(int, ArrayList<Integer>)	void
removeThLargest(int)	void
kthLargest(ArrayList<E>, int)	E
bubbleSort(ArrayList<E>)	void
theData	ArrayList<E>
empty	boolean

main	
main(String[])	voi

Problem solutions approach

Using the heap structure, arranging the elements thrown into the arraylist and solving the problem of subtracting the minimum to the heap. Find the searched element in an array and join 2 arrays and apply the heap structure. We first arrange the deletion of the largest element in the th row and then we do the deletion.

Test cases X Running command and results



```java
public static void main(String[] args) {
    HeapStructure<Integer> hs=new HeapStructure<~>();
    HeapStructure<Integer> ke=new HeapStructure<~>();
     System.out.println("-------------ADD--------------- ");
     hs.add(5);
     hs.add(6);
     hs.add(3);
     hs.add(2);
     hs.add(63);
     hs.add(576);
     hs.add(55);

     for(int i=0; i<hs.size();i++)
     {
         System.out.println(hs.getTheData().get(i));
     }

     System.out.println("-------------SEARCHFOR----------------- ");
     System.out.println("index : "+ hs.searchFor( target: 63));
     System.out.println("index : "+ hs.searchFor( target: 1));
     System.out.println("index : "+ hs.searchFor( target: 99));
     System.out.println("index : "+ hs.searchFor( target: 575));
     System.out.println("index : "+ hs.searchFor( target: 576));
     System.out.println("-----------MERGE------------- ");
     ke.add(8);
     ke.add(81);
     ke.add(34);
     ke.add(54);
     ke.add(3);
     ke.add(67);
     HeapStructure<Integer> res=new HeapStructure<~>();
     res.mergeHeaps(res.getTheData(),hs.getTheData(),ke.getTheData());
        for(int i=0; i<res.size();i++)
        {
            System.out.println(res.getTheData().get(i));
        }
        System.out.println("-----------REMOVE LARGEST------------- ");

         res.removeThLargest( a: 2);
        res.removeThLargest( a: 1);
        res.removeThLargest( a: 4);
        for(int i=0; i<res.size();i++)
        {
            System.out.println(res.getTheData().get(i));
        }
```

```
-------------ADD---------------
2
3
5
6
63
576
55
-------------SEARCHFOR-----------------
index : 4
index : -1
index : -1
index : -1
index : 5
-----------MERGE-------------
2
3
5
3
34
54
55
6
8
63
81
576
67
```

```
-----------REMOVE LARGEST-------------
2
3
3
5
6
8
54
55
63
576
```