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

Name : Md Sarwar Hossain
Student Num : 161044121

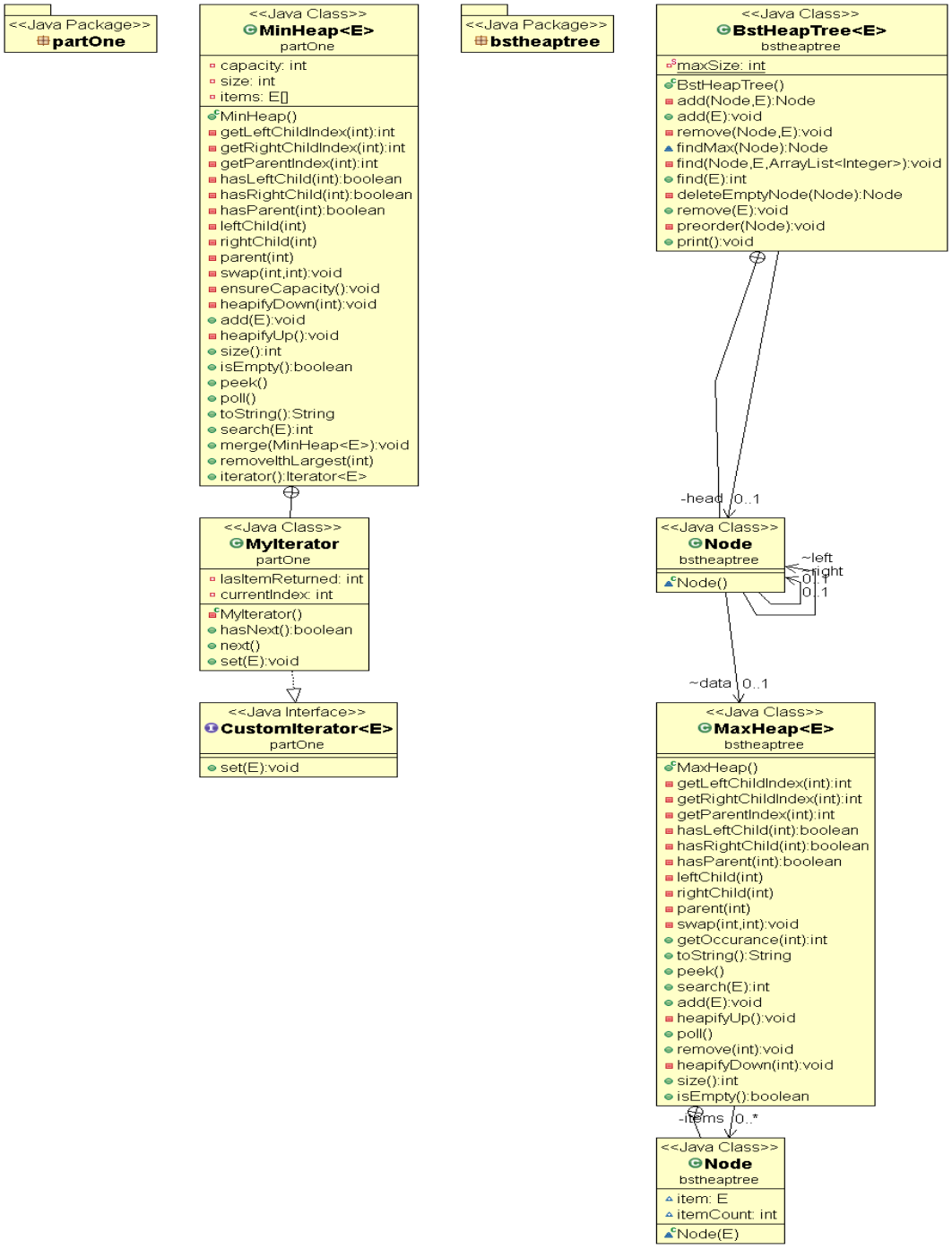
Part One

1. SYSTEM REQUIREMENTS

In part one we are asked to design a system where we should implement a heap data structure. The systems needs to behave the data structure like a heap where we can find essentials of a heap structure. Besides there are some specific method that the systems wants . Those are search for a item. Merge a heap with another heap . Removing the ith large element and also a iterator with special modification.

For the second part, it's a modified binary search tree where the nodes of the tree holds a heap. The heap has a maximum size and the datas are to filled in the heaps first and then if the heap is full , we should create another node. This tree structure will behave like normal bst but with slight modification. For duplicate items we should increase their occurrence size and while deleting we should decrease it's occurrence.

2. Class Diagram



3. PROBLEM SOLUTION APPROACH

For the heap design I used normal array in the first part and an arraylist in the second part. Basically both of them will work the same with slight difference in time complexity.

For the heap ,

The relation between a parent a child nodes are , `parentIndex*2+1(left child), parentIndex*2+2(right child) and the relation between child and parent is (childIndex-1)/2.`

`Based on this relation I designed both the max and min heap.`

`For the bstheap tree , keeping the special requirement in head, I used the peek of the heap to balance the tree.`

4. TEST CASES and Result

Case 1:

```
MinHeap<Integer> x = new MinHeap<>();
x.add(37);
x.add(23);
x.add(10);
x.add(16);
x.add(19);
x.add(9);
x.add(3);
System.out.println(x);
```

```
<terminated> Driver (3) [Java Application] C:\
[3, 16, 9, 37, 19, 23, 10]
```

Case 2:

```
{
    MinHeap<Integer> x = new MinHeap<>();
    x.add(37);
    x.add(23);
    x.add(10);
    x.add(16);
    x.add(19);
    x.add(9);
    x.add(3);
    System.out.println(x);

    MinHeap<Integer> y = new MinHeap<>();

    y.add(31);
    y.add(15);
    y.add(29);
    y.add(13);
    System.out.println(y);
    x.merge(y);
    System.out.println(x);
```

```
<terminated> Driver (3) [Java Application] C:\Users\swr\.p2\poo
[3, 16, 9, 37, 19, 23, 10]
[13, 15, 29, 31]
[3, 13, 9, 15, 19, 23, 10, 37, 16, 29, 31]
```

Case 3:

```
{
    MinHeap<Integer> x = new MinHeap<>();
    x.add(37);
    x.add(23);
    x.add(10);
    x.add(16);
    x.add(19);
    x.add(9);
    x.add(3);
    System.out.println(x);
    x.removeIthLargest(3);
    System.out.println(x);
```

```
<terminated> Driver (3) [Java Application] C:\U
[3, 16, 9, 37, 19, 23, 10]
[3, 16, 9, 37, 10, 23]
```

Case 4:

```
{
    MinHeap<Integer> x = new MinHeap<>();
    x.add(37);
    x.add(23);
    x.add(10);
    x.add(16);
    x.add(19);
    x.add(9);
    x.add(3);
    CustomIterator<Integer> it = (CustomIterator<Integer>) x.iterator();
    while(it.hasNext()) {
        System.out.println(it.next());
    }
    it.set(200);
    System.out.println(x);
```

```
<terminated> Driver (3) [Java Application] C:\Users\sv
3
16
9
37
19
23
10
[3, 16, 9, 37, 19, 23, 200]
```

Case 5:

```
{
    BstHeapTree<Integer> x = new BstHeapTree<>();
    x.add(37);
    x.add(23);
    x.add(10);
    x.add(16);
    x.add(19);
    x.add(9);
    x.add(3);

    x.add(31);
    x.add(15);
    x.add(29);
    x.add(13);

    x.add(124);
    x.add(52);
    x.add(98);
    x.add(51);
    x.add(38);
    x.add(87);
    x.add(80);

    x.add(60);
    x.add(57);
    x.add(43);
    x.add(54);
    x.add(39);

    x.print();
```

```
[37-1, 23-1, 10-1, 16-1, 19-1, 9-1, 3-1]
[31-1, 15-1, 29-1, 13-1]
[124-1, 52-1, 98-1, 51-1, 38-1, 87-1, 80-1]
[60-1, 57-1, 43-1, 54-1, 39-1]
```

Case 6:

```
x.print();
System.out.println("----------after removing---------");
x.remove(124);
x.remove(52);
x.remove(98);
x.remove(51);
x.remove(38);
x.remove(87);
x.remove(80);
x.print();
```

```
<terminated> Driver (3) [Java Application] C:\Users\swr\.p2\pool\plu
[37-1, 23-1, 10-1, 16-1, 19-1, 9-1, 3-1]
[31-1, 15-1, 29-1, 13-1]
[124-1, 52-1, 98-1, 51-1, 38-1, 87-1, 80-1]
[60-1, 57-1, 43-1, 54-1, 39-1]
----------after removing---------
[37-1, 23-1, 10-1, 16-1, 19-1, 9-1, 3-1]
[31-1, 15-1, 29-1, 13-1]
[60-1, 57-1, 43-1, 54-1, 39-1]
```

Case 7:

```
{
    BstHeapTree<Integer> x = new BstHeapTree<>();
    x.add(37);
    x.add(23);
    x.add(10);
    x.add(16);
    x.add(19);
    x.add(9);
    x.add(3);
    x.add(10);
    x.add(10);
    x.add(10);

}
```

```
<terminated> Driver (3) [Java Application] C:\Users\swr\.p2\pool\plu
[37-1, 23-1, 10-4, 16-1, 19-1, 9-1, 3-1]
```

Case 8:

```
BstHeapTree<Integer> x = new BstHeapTree<>();
x.add(37);
x.add(23);
x.add(10);
x.add(16);
x.add(19);
x.add(9);
x.add(3);
x.add(10);
x.add(10);
x.add(10);
x.print();
x.remove(7);
x.remove(10);
x.print();
```

```
<terminated> Driver (3) [Java Application] C:\Users\swr\.p
[37-1, 23-1, 10-4, 16-1, 19-1, 9-1, 3-1]
[37-1, 23-1, 10-3, 16-1, 19-1, 9-1, 3-1]
```

Case 9:

```
{
    BstHeapTree<Integer> x = new BstHeapTree<>();
    x.add(37);
    x.add(23);
    x.add(10);
    x.add(16);
    x.add(19);
    x.add(9);
    x.add(3);
    x.add(10);
    x.add(10);
    x.add(10);
    x.print();
    System.out.println("occurance of 10 : "+x.find(10));


}

}
```

```
<terminated> Driver (3) [Java Application] C:\Users\swr\.p2\p
[37-1, 23-1, 10-4, 16-1, 19-1, 9-1, 3-1]
occurance of 10 : 4
```