

Objective(s):

- To understand heap insertion and removal mechanism.
- To understand the relationship between heap and priority queue.

Task 1: Create **directory named pack8_Trees** with package code inside. Study MyMinHeap's insert(int d) and remove() methods. Test L8_PQ_Main's demo1().

```
import code.*;
public class L8_PQ_Main {
    static ArrayList<Integer> least3;

    public static void main(String[] args) {
        println("-demo1---");
        demo1();
        // println("-demo2---");
        // demo2();
    }
    static void demo1() {
        least3 = new ArrayList<>();
        MyMinHeap heap = new MyMinHeap();
        heap.insert(11);    heap.insert(15);
        heap.insert(16);    heap.insert(13);
        heap.insert(17);    heap.insert(18);
        println("heap strucutre is " + heap);
        least3.add(heap.remove());
        least3.add(heap.remove());
        least3.add(heap.remove());
        println("least 3 value is " + least3);
    }
    static void demo2() {
        least3 = new ArrayList<>();
        MyPriorityQueue pq = new MyPriorityQueue();
        pq.enqueue(11);    pq.enqueue(15);
        pq.enqueue(16);    pq.enqueue(13);
        pq.enqueue(17);    pq.enqueue(18);
        pq.enqueue(19);    // <-- isFull() is true ... discard
        println("pq structure is " + pq);
        least3.add(heap.remove());
        least3.add(heap.remove());
        least3.add(heap.remove());
        println("least 3 value is " + least3);
    }
}
```

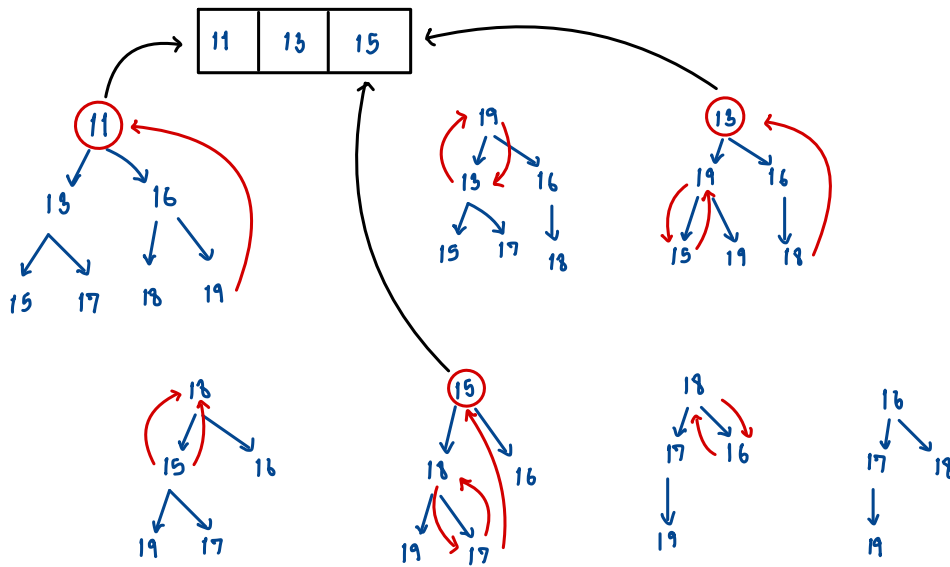
Task 2: Given an abstract class `MyQueueInterface.java`, implement `MyPriorityQueue.java` from `MyMinHeap`'s capabilities.

Note that `MyMinHeap`'s implementation is rudimentary because it does not check whether the heap is full or empty when insert or remove. Make sure that your `enqueue()` and `dequeue()` do not have such mistakes.

```
package code;

public interface MyQueueInterface {
    public void enqueue(int d);mp
    public int dequeue();
    public int front();
    public boolean isFull();
    public boolean isEmpty();
}
```

Task 3: draw / write the heap snapshot during each `dequeue()` was performed.



Submission: `MyPriorityQueue_XXXXXX.java` and this pdf.

Due date: TBA