Filipe de Oliveira Ataíde – Mat.: 20181014040022

Fila de Prioridade com Heap e Array

Main.java

```
public class Main{
	public static void main(String[] args) {
		Array heap = new Array(99);

		heap.insert(17);
		heap.insert(18);
		heap.insert(25);
		heap.insert(13);
		heap.insert(1);
		heap.insert(5);

		heap.removeMin();
		heap.removeMin();
		heap.removeMin();
		heap.removeMin();

		heap.print();
	}
}
```

FilaPrioridade.java

```
public interface FilaPrioridade{
	public int size();
	public int min();
	public void insert(int o);
	public int removeMin();
	public boolean isEmpty();
}
```

Array.java

```
public class Array implements FilaPrioridade{
	int heap[];
	int t = 0;

	public Array(int t){
		heap = new int[t];
	}

	public int size(){
		return t+1;
	}

	public boolean isEmpty(){
		return t == 0;
	}

	public boolean isFull(){
		return t == heap.length;
	}

	public int getParent(int i){
		return (i -1)/2;
	}

	public int getChild(int i, boolean l){
		return 2 * i + (l ? 1 : 2);
	}

	public void insert(int o){
		if(isFull()){
			throw new IndexOutOfBoundsException("Heap is full");
		}
		heap[t] = o;
		UpHeap(t);
		t++;
	}

	private void UpHeap(int i){
		int o = heap[i];
		while (i > 0 && o > heap[getParent(i)]){
			heap[i] = heap[getParent(i)];
			i = getParent(i);
		}
		heap[i] = o;
	}

	private void DownHeap(int i, int last){
		int child; // child a trocar
		while (i <= last){
			int lChild = getChild(i, true);
			int rChild = getChild(i, false);
			if(lChild <= last){
				if(rChild > last){
					child = lChild;
				}
				else{
					child = (heap[lChild] > heap[rChild] ? lChild : rChild);
				}
				if (heap[i] < heap[child]){
					int aux = heap[i];
					heap[i] = heap[child];
					heap[child] = aux;
				}
				else{
					break;
				}
				i = child;
			}
			else{
				break;
			}
		}
	}

	public int min(){
		int o = heap[0];
/*		for(int i = 0; i <= t; i++){
			if(i.key < m.key){
				m = i;
```

```
            }
        }*/
        return o;
    }

    public int removeMin(){
        int o = heap[0];
        heap[0] = heap[t - 1];
        heap[t - 1] = o;
        t--;
        DownHeap(0, t-1);
        return o;
    }

    // print

    public void print(){
        for(int i = 0; i < t; i++){
            System.out.print(heap[i]);
            System.out.print(", ");
        }
        System.out.println();
    }
}
```