DLPriorityQueue Psudocode

```
public DLPriorityQueue()
  - set head & tail to null
```

```
public void add(T dataItem, dble priority)
  - boolean value 'sorted' to check if node was sorted
  - curr = head, get priority, prev
  - while (curr.getNext != null && sorted = false)
      ↳ if (dataItem.priorty ≤ curr.priority)
          ↳ dataItem.prev = curr.prev
          - curr prev = dataItem
          - dataItem.next = curr
          - sorted = True or try break;
    - else
        ↳ curr = curr.getNext()
```

```
public void updatePriority(T dataItem, dble priority) throws InvalidElementException
    - boolean value  dataItemExists = false
    - while (curr.getNext() != null)
        ↳ If (dataItem == curr.getDataItem())
            ↳ dataItem Exists = true
            - break;
        - else
            ↳ curr.getNext()
    - if (dataItemExists != True
        ↳ throw InvalidElementException
    - create temp dataItem, get next & prev
    - curr = head, getnext & prev
    - sorted = false
    - while (curr.getNext() != null and sorted = false)
        ↳ if (dataItem.priority ≤ curr.priority)
            ↳ dataItem.prev = curr.prev
            - curr prev = dataItem
            - dataItem.next = curr
            - sorted = True
        - else
            ↳ curr = curr.getNext()
```

```
public T removeMin() Throws EmptyPQException
  - if (head = null)
      ↳ Throw EmptyPQException
  - RmvdDataItem = head
  - (RmvdDataItem.next).setPrev = null
  - RmvdDataItem setNext = null
  - return RmvdDataItem.getDataItem
```

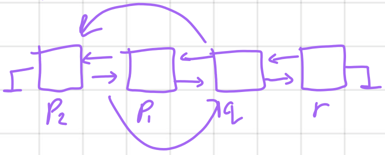
```
public boolean isEmpty()
  - if (front == null and rear == null)
      ↳ return true
  - else
      ↳ return false
```

```
public int size()
  - counter = 0
  - while (curr.getNext() != null)
      ↳ counter ++
      - curr = curr.getNext()
  - return counter
```

```
public String toString()
```

```
public DLinkedNode<T> getRear()
  - return rear
```

p.getPrev setNext(p getNext)
p.getNext.setPrev(p.getPrev)



```
if (front = null)
    front = newNode
    rear = newNode
```