

UnorderedListPriorityQueue.java

```
1  /* Vincent Chan
2   * masc0264
3   */
4  package data_structures;
5
6  import java.util.Iterator;
7
8  public class UnorderedListPriorityQueue<E> implements PriorityQueue<E> {
9      /* Functions Included
10       * ====PUBLIC====
11       * constructors          //Ln. 26
12       * insert(object)        //Ln. 31
13       * remove()              //Ln. 37
14       * peek()                //Ln. 43
15       * contains(Object)      //Ln. 63
16       * size()                //Ln. 68
17       * clear()               //Ln. 73
18       * isEmpty()             //Ln. 78
19       * isFull()              //Ln. 83
20       * iterator()            //Ln. 89
21       */
22
23     //Variable declarations
24     UnorderedList<E> list;
25
26     //Constructor
27     public UnorderedListPriorityQueue() {
28         list = new UnorderedList<E>();
29     } //End constructor
30
31     //Inserts a new object. Returns true if insertion is succesful
32     public boolean insert(E object) {
33         list.addFirst(object);
34         return true;
35     } //End insert()
36
37     //Removes the object with highest priority
38     //Returns null if the list is empty
39     public E remove() {
40         return list.remove(peek());
41     } //End remove()
42
43     //Gets the object with the highest priority
44     //the longest but does NOT remove it.
45     //returns null if empty
46     public E peek() {
47         if(list.isEmpty()) return null;
48         int comp;
49         E lowPri = null;
50         for(E current : list) {
51             if(lowPri==null) {
52                 lowPri = current;
53                 continue;
54             }
55             comp = ((Comparable<E>)lowPri).compareTo(current);
56             if(comp>0) {
57                 lowPri = current;
58             }
59         }
60         return lowPri;
61     } //End peek()
62
63     //Returns true if the queue contains the specified element
64     public boolean contains(E obj) {
```

```
65     return list.find(obj) != null;
66 } //End contains()
67
68 //Returns the number of objects in the queue.
69 public int size() {
70     return list.size();
71 }
72
73 //returns to an empty list.
74 public void clear() {
75     list = new UnorderedList<E>();
76 } //End clear()
77
78 //Returns true if queue is empty
79 public boolean isEmpty() {
80     return list.isEmpty();
81 } //End isEmpty()
82
83 //Returns true if full. otherwise false.
84 //Linked lists are never full.
85 public boolean isFull() {
86     return false;
87 } //End isFull()
88
89 //Returns an iterator of the objects in the PQ,
90 public Iterator<E> iterator() {
91     return list.iterator();
92 } //End iterator()
93 } //End UnorderedListPriorityQueue
```