Homework #14
due Saturday, May 22, midnight

In this assignment you will implement a `PriorityQueue` of integers.

1 Concerning the `PriorityQueue`

The ADT for the `PriorityQueue` is quite simple. Its `add` method adds an element to the `PriorityQueue` with some priority. The `remove` method removes and returns the element with the highest priority `cost`, or throws a `NoSuchElementException` if the queue is empty.

To implement this ADT, we will use a minheap data structure. It is an array-based ternary minheap. Ternary means that each implied node can have up to three children, which is different from the binary heaps we have talked about. The `n` elements will be stored in the first `n` indices of the array, such that each element has higher priority than its implied parent. This means that the elements stored at indices `1`, `2`, and `3` must have a higher priority than the one stored at index `0`, and so on. Adding involves placing an element at the end, and then "bubbling up." Removing involves moving the last element to the first position and then "bubbling down." See the textbook pages 521-527, as well as 649-659 (3rd ed: pp. 511-517, 633-643) and lecture notes for more information. You will have to figure out how to modify the index math to make it ternary instead of binary.

Note that, because we are using `Integer` objects, we use a very simple means of comparison. You will write a `Comparator` that uses increasing numeric order on integers.

2 What you need to do

You need to finish the implementation of `IntegerPriorityQueue.java`, including the invariant and all public methods.

3 Files

The repository includes the following files:

src/TestInvariant.java Tests your invariant.

src/TestPriorityQueue.java Tests your ADT.

src/TestEfficiency.java Tests your ADT's efficiency.

src/edu/uwm/cs351/IntegerPriorityQueue.java The main ADT to work on.