Question 2 – The “Big” Coding Question – So, how about those waitlists?

For the next few questions, answer them with regards to the description below.

It’s that time of year… you know what we mean. The season of angst, heartache, and anxiousness about your future. What? Holiday break? Oh, I was talking about class registration.

Every UVa student (and faculty member) has been dealing with increase in enrollment but not necessarily an increase in the number of sections of a course being offered. So all of us have been trying to determine exactly how to best work the waitlist.

One thing that faculty hate about the current waitlist system is that we can’t give a higher priority to certain students – it’s all based on when they entered the list. You may think this is how it should be, but sometimes there are special cases that faculty just have to take care of somehow.

So, what we need here is a wait list that also takes into account people’s priority, which we’ll call a “priority queue.” Basically, what we’ll do is create a list and then within that list there will be an indicator with each item as to its relative priority in the list. When you add a person to the list, that person can have any positive-value priority. When you remove someone, you should *always* get the person from the highest value priority group that entered the queue first.

The implementation of the priority queue in the `PriorityWaitList` needs to have the following:

- An `ArrayList` to hold `WaitListEntry` objects. `WaitListEntry` will be used to hold a student name and priority value. You can find this class in the appendix of the test. *We provide this field for you. You do not need to (and should not) create any additional fields.*
- A default constructor method that will initialize the field.
- A method called `addToWaitList` that will take a name and priority value and add it to the list. It does not return anything. You can assume that all priority values will be positive integers.
- A method called `getNextOnWaitlist` that will remove the next student from the waitlist based first on priority value (a higher number is higher priority) and then by when they entered the list. It will return the name of the student taken from the list and remove that entry from the `ArrayList`.
- A method called `getWaitListSize` that will take a priority value and will return the number remaining in the waitlist with that priority. If the priority value passed in is 0 or less, the method will return the total number of people in the list, regardless of priority value.

In this question you will need to write a Java class called `PriorityWaitList` that implements the priority queue data type described above. The methods listed above will be class-level methods of this class. *You do not need to (and should not) create any additional methods. You must match our API as described above! How you implement the methods is up to you.*
public class PriorityWaitList {
  private ArrayList<WaitListEntry> waitlist;
  // Constructor ------------------------------ 5 pts

  // addToWaitList method --------------------- 5 pts

  // getNextOnWaitlist method ----------------- 5 pts

  // getWaitListSize method -------------------- 5 pts
Question 2b – Using the WaitList [15 points]

Now that you have a `PriorityWaitList` class, it’s time to use it. To the right is an example waitlist made by a professor. **Do not hardcode any values from this file! Do not assume any other methods are in `PriorityWaitList` other than the ones you wrote!**

Fill in the `main` method below so that it does the following: 1) read the file called `list.txt`; 2) add the students to the waitlist using the method(s) you wrote; 3) print the total number of students with priority 5 using the method(s) you wrote; and, 4) print out the students in priority order using the method(s) you wrote.

```java
public static void main(String[] args) throws Exception {

```
Appendix:

```java
public class WaitListEntry {

    private String name;
    private int priority;

    public WaitListEntry(String name, int priority) {
        this.name = name;
        this.priority = priority;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getPriority() {
        return priority;
    }

    public void setPriority(int priority) {
        this.priority = priority;
    }

}
```