

Create a Priority Queue Class

In this challenge you will be creating a Priority Queue. A Priority Queue is a special type of Queue in which items may have additional information which specifies their priority. This could be simply represented with an integer. Item priority will override placement order in determining the sequence items are dequeued. If an item with a higher priority is enqueued after items with lower priority, the higher priority item will be dequeued before all the others.

For instance, let's imagine we have a priority queue with three items:



```
[[ 'kitten', 2], [ 'dog', 2], [ 'rabbit', 2]]
```








Here the second value (an integer) represents item priority. If we enqueue `['human', 1]` with a priority of 1 (assuming lower priorities are given precedence) it would then be the first item to be dequeued. The collection would look like this:

```
[[ 'human', 1], [ 'kitten', 2], [ 'dog', 2], [ 'rabbit', 2]]
```

We've started writing a `PriorityQueue` in the code editor. You will need to add an `enqueue` method for adding items with a priority, a `dequeue` method for removing and returning items, a `size` method to return the number of items in the queue, a `front` method to return the element at the front of the queue, and finally an `isEmpty` method that will return `true` if the queue is empty or `false` if it is not.

The `enqueue` should accept items with the format shown above (`['human', 1]`) where 1 represents the priority. `dequeue` and `front` should return only the item's name, not its priority.

	Your <code>PriorityQueue</code> class should have a <code>enqueue</code> method.
	Your <code>PriorityQueue</code> class should have a <code>dequeue</code> method.

	Your <code>PriorityQueue</code> class should have a <code>size</code> method.
	Your <code>PriorityQueue</code> class should have a <code>front</code> method.
	Your <code>PriorityQueue</code> class should have an <code>isEmpty</code> method.
	Your <code>PriorityQueue</code> class should correctly keep track of the current number of items using the <code>size</code> method as items are enqueued and dequeued.
	The <code>front</code> method should return the correct item at the front of the queue as items are enqueued and dequeued.
	The <code>isEmpty</code> method should return <code>true</code> when the queue is empty.
	The priority queue should return items with a higher priority before items with a lower priority and return items in first-in-first-out order otherwise.