**CS:2820 Object Oriented Design Spring 2020**

**HW # 4: Interfaces and Abstractions**

**Deadline: April 14 at 11:59 pm**

**Priority Queues Abstract Data Types (ADT)**

A priority queue is a data structures where elements are stored and retrieved based on ***priority***. For example, in a stack data structure, data are retrieved in last- in/first-out fashion. Which means, the last element inserted is the first element retrieved; i.e. the element with highest priority is the last one inserted. Another example is the queue data structure, where data are retrieved in a first-in/first out fashion. (Do not confuse the term “priority queue” with the “queue data structure”; priority queue is an umbrella for several data structures, where queue data structure is one type of a priority queue.)

Priority in a priority queue can be defined in several ways: last-in/first-out (stack), or first-in/first-out (queue), maximum element (max-heap), and so.

From the definition of priority queue ADT, we can see that priority queue is a general term, an abstraction for several data structures. In other words, stack, queue, max-heap, and so, can be arranged in a class hierarchy under priority queue.

**Task**

Your task in this hw is to make the following class hierarchy:

1. PrioirtyQueue (interface/abstract class: which means it contains pure virtual methods)
2. Stack (inherits PriorityQueue)
3. Queue (inherits PriorityQueue)

The priority queue interface will have the following methods:

1. void enqueue(T data): add the data to the priority queue.
2. T dequeue(): remove the highest priority element from the data structure and return it.
3. T peek(): Return the highest priority element in the queue without removing it.
4. bool isEmpty(): return true if the queue is empty.

Your implementation needs to be composed of multiple files, where each class is written in its own class file.

**NOTES**:

1. The priority queue might hold variable capacity—no capacity constraints. Thus, it is recommended to implement it with DynamicArray. Use your DynamicArray implementation with some modifications to allow it to perform according to specification.
2. The priority queue needs to use templates, so we can store any data types we want. Since we are implementing stack and queue, we can store any data type, even if it does not override “==” operator, or so. (If we are to implement a max heap, the stored data needs to override “>” and “==” operators for things to work correctly.)
3. Just like in HW3, you need to test your implementation with different data types (classes), to ensure it works with different data without problems.