Homework 7 – Ps & Qs

C++ II

Description

For this assignment we will re-implement the min heap we developed in class and create a priority queue class.

Specifications

You will need to complete the following:

1. Write an implementation of the max_heap class with the same public interface we used in class for the min_heap.
2. Modify the implementation of the min_heap class so that the 0 element of the vector is not wasted.
3. An ordinary queue is a first-in, first-out data structure.
    a. Elements are appended to the end of the queue and removed from the beginning.
    b. In a priority queue (PQ), elements are assigned with priorities.
    c. When accessing elements, the element with the highest priority (the max value) is removed first. You could also have a PQ where the element with the least priority (min value) is remove first.
    d. Queues are used in real life in many instances, for example, the emergency room in a hospital assigns priority numbers to patients; the patient with the highest priority is treated first.
4. Using the max_heap you wrote earlier, create a priority_queue class which uses your max_heap and exposes the following public methods –
    a. offer(E value) - inserts an element into the PQ, NOTE you will need a reference and rvalue implementation.
    b. peek() - Retrieves, but DOES NOT remove, the head of the queue, if the queue is empty then null is returned.
    c. poll() - Retrieves AND removes the head of the queue (in other words, the first element of this PQ), or returns null if this queue is empty.
    d. is_empty() – Boolean to determine if the PQ is empty
    e. size() – Returns the number of elements in the PQ
    f. clear() - Removes all of the elements from the PQ.
5. Write a main method which test all the public methods of both the max_heap and PQ class. You should be able to iterate over the PQ using a while (!is_empty()) { poll() }. Be sure your code will work with the loop given.

Documentation

You will create a document (.docx, .rtf, .pdf) which contains the following:

- Your name and assignment.

- A screenshot of your code output.
- Explain the difference between a binary tree, a binary search tree, and a heap, be specific regarding the differences.
- Explain how you could implement the priority_queue with a sorted vector in place of a heap. What might be the benefits / drawbacks of using this implementation over a heap?

What to Submit

You need to submit your C++ code files along with your document. Make sure your document is in the correct format and all your files include your name and assignment. ***ZIP*** your C++ code, but **DO NOT** zip your document file.