Homework – Queues

Brad Peterson – Weber State University

The goals of this assignment are to:

* Help review C++.
* To better understand classes, arrays, dynamic allocation and deallocation, error handling, templates, and basic logic.
* To help you understand C++11 move semantics.

For my unit tests, I made a base class with methods that simply have enough logic to compile. You should not modify the base class. Instead, you should modify the derived class and override the constructor, destructor, and all methods there (I gave you an example of overriding the constructor's declaration).

Note that `arr` is defined in the base class, but you want to use it in the derived class. For most compilers, you must access it using `this->arr` instead of just `arr`.

The `QueueForCS2420` class needs to have the following members:

- Four unsigned integer data members. They should be variable to hold 1) the total capacity of the array, 2) the number of slots used in the queue, 3) the head of the queue, and 4) the tail of the queue. All can be initialized to zero.
- A constructor that accepts a capacity parameter. The parameter needs to be copied into the appropriate data member. The array must be allocated using the new keyword and the address stored in `arr`.
- You need a destructor, because you used the new keyword in the constructor.
- A `numItems()` method. The return type is unsigned int. Returns the number of used slots in the queue.
- A `push_back()` method. This method should have a single parameter, the data type of that parameter should be `const T&`. The `push_back()` method should have a void return value. This method should see if the queue is full. If so, simply state an error message and return. Otherwise, check if your tail index is too large (the same as the number of elements in the queue). If so, set the tail index back to zero. Put the data in the array at the tail index. Increment the tail and the number of elements.
- A `pop_front()` method. This method should not have any parameter. The return type should be void. The method first checks if the queue is empty. If so, simply state an error message and return. Otherwise, increment the head, see if head is out of bounds and needs to be wrapped back to zero, then decrement the size.
- A `front()` method and a `back()`. These check if the queue is empty. If so, state an error message and throw an integer. Otherwise, return the data at the head or the tail of the queue, respectively (not that the tail index is always one past the actual item).
- A move constructor method. The constructor copies over all five data members (including arr). The original object's data member are all set to zero or nullptr.
- A move assignment method. This first checks if the "this" object has a valid array, and if so, reclaims it. The method copies over all five data members (including arr). The original object's data member are all set to zero or nullptr. The method finishes with returning itself (return *this;).

Use the .cpp file given. Your assignment should pass all tests.

The lecture videos also carefully walk you through this process. If you haven't yet learned how to use your debugger, please do so!