CX 4010 / CSE 6010
Assignment 2
Software Modules

Due Dates:

- Due: 11 AM, Friday, September 6, 2019
- No late submissions will be accepted

The objectives of this assignment are to write a program to implement a program module implementing a priority queue using a sorted, linear linked list data structure and to write a unit test to verify the program works correctly. The priority queue code should be structured as if it were part of a software library to be used with other programs written by someone else. Small keys indicate higher priority. The software should implement the following interface:

1. Use a `typedef` to define a data structure called `PrioQ`, defined as a C `struct` representing the priority queue. The implementation of the data structure is not visible outside the module you are creating.

2. Write a function with the prototype

   ```
   PrioQ *PQ_create();
   ```

   Create a new priority queue and return a pointer to it. The priority queue is initially empty, i.e., contains no elements. Return `NULL` if the operation failed, e.g., due to a lack of memory.

3. Write a function with the prototype

   ```
   int PQ_insert(PrioQ *PQ, double key, void *data);
   ```

   Insert the item pointed to by `data` into the priority queue `PQ`. The priority of this data item is `key`. Return `0` if the operation failed, or an arbitrary value not equal to `0` if it succeeded. The data type of the inserted item pointed to by `data` is defined by the caller, and is not visible to the priority queue module.

4. Write a function with the prototype

   ```
   void *PQ_delete(PrioQ *PQ, double *key);
   ```

   Remove the data item from the priority queue `PQ` with the smallest key (highest priority). The function returns a pointer to the data item that was removed, or `NULL` if the queue was empty. The priority of the deleted item is returned in `key`.

5. Write a function with the prototype

   ```
   unsigned int PQ_count(PrioQ *PQ);
   ```

   Returns the number of items currently residing in the priority queue.

6. Write a function with the prototype

```
void PQ_print(PrioQ *PQ);
```

Prints the priority of all of the items currently in the priority queue from the highest priority (smallest key) to lowest priority (largest key) without changing the contents of the priority queue.

7. Write a function with the prototype

```
void PQ_free(struct PrioQ *PQ);
```

Delete the priority queue `PQ` by releasing all memory used by the contents of the data structure.

Write a "unit test" program in a separate file from the priority queue module to test your priority queue for a variety of test cases. The program should exercise the priority queue using a sequence of test cases, including (i) creating the priority queue, (ii) inserting an item into an empty queue, (iii) inserting an item at the beginning, middle, and end of the linked list for a queue containing at least five elements, (iv) removing all of the items in a queue containing at least five elements, one after the other, and (v) deleting an item from an empty queue (error case). For each test case use the `PQ_count()` and `PQ_print()` functions to verify the operations work correctly.

Your code should include a .h file that defines the interface to the software that includes only the information necessary to use the module. It must *not* specify the internal implementation of the code.

Turn in your software in a single zip file. Your software must be well documented and include comments so the code is easy to understand. Also turn in a brief report describing the results of your tests, including the output produced by the test program.

Grading

Your grade will be based on the structuring and documentation of your code, and the brief report verifying the code works correctly.

Collaboration, Citing, and Honor Code

As a reminder, please refer to the course syllabus regarding rules and expectations regarding collaborating with other students and use of other resources, including materials available on the web.