PROG8130
Data Structures & Algorithms for Embedded Programming

Lab 2
Circular Queues

Shamseddin Elmasri

selmasri@conestogac.on.ca

Objectives

- Implement a circular queue
- Simulate the idea of priority scheduling

Procedure

- Three queues have already been created.
- An enum has been declared to represent queue priority: HIGH_PRIORITY, MED_PRIORITY and LOW_PRIORITY
- Using the program provided as a starting point, complete the enqueuing and dequeuing functions.

Procedure

- Inside the main() function, there is a while loop that runs for 30 seconds using the functions provided in time.h header file.
- An element is added to the high priority queue every 6 seconds, while the medium and low priority queues are enqueued every 4 and 2 seconds respectively.
- The dequeuing process is done based on the priority level every 1 second.

Submission

- Submit your completed program to eConestoga.
- Program must be in a single (.c) source file that can be complied on its own.
- Any number of submissions will be accepted, but only the latest one before the deadline will be evaluated.

Evaluation

- Your program will be evaluated for the following criteria:
  - Coding style, proper indentation, comments.
  - Correct functionality.
  - Compilation without any warnings or errors.
  - Proper error handling
  - Marks will be deducted for:
    - Extra debug left in program after submission
    - Extra unused elements in data structures

Basic Outcomes (55 to 69%)

- Enqueuing and dequeuing functions work as per procedure.
- Make sure to implement and use queue_IsFull() and queue_IsEmpty() functions.

Intermediate Outcomes (70 – 89%)

- Basic outcomes fully met
- Complete the following functions to display the dequeued items during the duration of the program:

queue_UpdateDequeuedItems()

queue_DisplayDequeuedItems()

Advanced Outcomes (90 – 100%)

- Intermediate outcomes fully met
- Study time.h library and use one or more of its functions to add a creative feature in your code.