CS110: Computer Programming Lab
Department of CSE
IIT, Guwahati

Module 04 Stage 02 Exercise 02

Assessment exercises are designed to help us check if the student has learned the basics of the topics included in the drill instructions. However, a drill assessment is not a comprehensive assessment. A fuller and complete assessment aimed at determining the course grades will be done through CS110 examinations.

It is expected that the student will attempt and solve many more exercises from the drill assessment sets to improve their programming skills and for an excellent performance at the examinations.

Exercise

Add code in file `queue.c` to implement method/function `forceRmQueue()`. The code must free all memories referenced in the queue. For this exercise assume that all application objects whose references are stored in the queue were assigned memory through call a to function `malloc()`.

Your attention is drawn to the fact that reference to a single application object may be added to a queue more than once. You must ensure that your code free such objects only once; not for every reference found there.

A possible way to attend to this issue is by scanning all `queue_elem` as memory for an application object is freed by `stdlib` function `free()`, and replace all usesof the same pointer value by NULL. This will prevent multiple calls to `free()` for the same `malloc()` memory.

Verify that your code above works correctly. The interface specification has suggested that the method returns the count of application objects be returned. This value can be checked to match the expected value in the tests-first suite.

Suggested/Referred Interface

```
#ifndef QUEUE_H_INCLUDED
#define QUEUE_H_INCLUDED

/* Returns a reference to a new queue */
void * mkQueue(void);

/* Removes a queue specified by a valid reference */
int rmQueue (void *);

/* Returns number of entries in a validly referenced queue */
```

```
int sizeQ (void *);

/* Place a new entry and returns new count of entries */
int joinQ (void * queueP, void * objP);

/* Returns reference to the most ancient arrival in queue */
void * leaveQ (void * queueRef);

/* Force delete queue and its malloc() memory */
int forceRmQueue (void *);

#endif // QUEUE_H_INCLUDED
```