F.G.H.
5/30/85

LispCourse #32: Homework on circular queues

Exercise

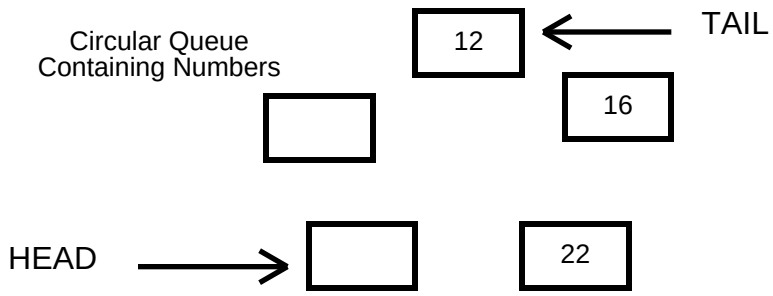
Overall problem

Write the functions to manage a circular queue. Randomly add and delete random numbers to and from this queue. Print out the contents of the queue after each addition or deletion.

Defintions

A ***queue*** is a data structure that can hold from 1 to N other data objects. The queue has a beginning (called its ***head***) and an end (called its ***tail***). You can add new data data items to the queue (as long as you don't exceed N items) and you can remove data items from the queue. New data objects are always added to the tail of the queue and old data items are always removed from the head of the queue, just like in the queues at banks, airline counters, etc.

A ***circular queue*** is one where the N places for data objects are arranged in a circle. There is a pointer for the head and a pointer for the tail. When you add a new data object, you put it into the place indicated by the head pointer, and then move the head pointer clockwise by one. When you remove a data object, you take the one indicated by the tail pointer, and then move the tail pointer clockwise by one. You can keep adding and deleteing elements until the head pointer equals the tail pointer, in which case the queue is either empty or full, depending on whether you just removed or added an item.



Subproblems

1) Write a function that produces a list of N NILs.

2) Write a function that creates a circular list out of a non-circular list by making the CDR of the last CONS cell in the list point to the CAR of the first CONS cell in the list. (Caution: You will have to ^D when Lisp starts printing the result of this function otherwise it will go on forever.)

3) Use the functions from 1 and 2 to create a circular queue (i.e., list) containing all NILs. (See caution from 2). SETQ LC.Queue to this circular list.

4) Set up the pointers LC.Head and LC.Tail. Originally both should point to the same CONS cell that LC.Queue does, indicating that the queue is empty.

5) Diagram the CONS cells in the queue and the LC.Queue, LC.Head, and LC.Tail pointers. Use this diagram to help you in the following problems.

6) Write a function that adds a number to the tail of the queue. Be sure to check first that the queue is not full. If it is return NIL, otherwise return T. (Hint: RPLACA and CDR are critical here).

7) Write a function that removes a number from the head of the queue. Be sure to check first that the queue is not empty. If it is, return NIL. (Hint: CAR and CDR are critical here).

8) Write a function the prints all the numbers in the queue. (Hint: Produce a list starting at the tail and going to the head, then reverse this list and print it.) Note: Use the function **(PRIN1 *X*)** to print a *X* in the Exec window. Use **(TERPRI)** to print end a line.

9) Write a function that randomly excercises the queue as follows:

    Repeat the following 100 times.

        Print the queue.

        Generate a random integer between 0 and 1, inclusive.

            If this random integer is 0, then remove an item from the queue and print it preceded by an appropriate message. If the delete

fails because the queue is empty, just print an appropriate message.

If this random integer is 1, then generate a second random integer between 1 and 99. Add this second random integer to the queue and print the number preceded by an appropriate message. If the add fails because the queue is full, just print an appropriate message.

Note: The function call **(RAND *N M*)** generates a random integer between *N* and *M*.