COP 3502 Section 2 Exam #1 - Part C (Queues) Solutions

Date: 6/8/2020

Start Time: 4:55 pm EST
End Time: 5:25 pm EST

1) (4 pts) In a linked list implementation of a queue, the queue struct contains two pointers instead of one.

(a) Are both pointers ***necessary*** to implement the queue?

No. **(Grading: 1 pt all or nothing)**

(b) What is the purpose of having both pointers?

If only a front pointer stored the list, then the enqueue operation would require looping through the whole list and could potentially take a lot of time. (Specifically, it would take O(n) time, where n represented the number of items in the list.) By including a pointer to the back of the list, the enqueue operation can be executed in O(1) time, with a fixed number of statements, no matter how long the queue is. **(Grading: 1 pt for saying easy access to both sides of the list, 2 pts for explaining the difference in run time in some fashion. O(1), O(n) not necessary since we haven't taught order notation yet.)**

2) (12 pts) In class, a queue was used in a live coding exercise to find the shortest path out of a maze, where a single move was moving up (subtracting 1 from the row), moving left (subtracting 1 from the column), moving right (adding 1 to the column) or moving down (adding 1 to the row). The queue stored integers, which represented row-column coordinates into the maze, which was represented as a two dimensional character array. The characters in the array were as follows:

```
'S' : starting location
'X' : illegal location
'-' : valid location to travel to
'~' : maze border
```

Given the following input maze (6 rows, 8 columns), determine the first 12 integers enqueued into the queue, and ALSO state which 0-based row, column coordinates those integers represent. Note that there are different correct answers to this question, but all correct answers have the same set of integers/ordered pairs, just in slightly different orders. In order to receive full credit, you must provide your list in one of the valid orderings.

```
6 8
~~~~~~~~
~-------~
~---XX--~
~---S-X~
~--XX--~
```

~~~~~~~~

Please present your answer in the following format:

1. Value 18, Location (4, 2)
2. Value 19, Location (4, 3)
3. …
…
12. Value 9, Location (2, 1)

Note that in my description above, both the values and locations listed are dummy values and not the correct answers for this particular problem. They have just been put in to make the format I want clear.

Solution

1. 28, ordered pair (3, 4), row 3 column 4
2. 27, ordered pair (3, 3)
3. 29, ordered pair (3, 5)
4. 19, ordered pair (2, 3)
5. 26, ordered pair (3, 2)
6. 37, ordered pair (4, 5)
7. 11, ordered pair (1, 3)
8. 18, ordered pair (2, 2)
9. 25, ordered pair (3, 1)
10. 34, ordered pair (4, 2)
11. 38, ordered pair (4, 6)
12. 45, ordered pair (5, 5)

Grading: different ordering of answers possible, trace through their BFS to see if it's valid. 1 pt per row. Take off 2 pts if invalid ordering total.

3) (9 pts) Consider an array implementation of a queue where the size of the array is 5 and the following operations have occurred (assume that queue is the variable storing the queue and that the queue has just been initialized to be empty right before this code segment executes):

```
enqueue(&queue, 5);
enqueue(&queue, 3);
enqueue(&queue, 2);
int a = dequeue(&queue);
int b = dequeue(&queue);
enqueue(&queue, 7);
int c = dequeue(&queue);
enqueue(&queue, 4);
enqueue(&queue, 9);
enqueue(&queue, 1);
```
~~~~~~~~

Answer the following questions:

(a) In which array index (of the array in the array in the queue struct) is the value 9 stored?

Answer: index 0, the value 4 is stored in index 4, after which there is a wrap around and 9 is stored in index 0.

Grading: 3 pts all or nothing, no reason needed.

(b) In the array implementation of a queue, the variable front keeps track of the index into the array of the front of the queue. What is the value of front after this code segment is completed?

Answer: front is 3. It starts at 0, then increments three times, once for each dequeuer.

Grading: 3 pts all or nothing, no reason needed.

(c) How many elements are in the queue right after this code segment completes?

Answer: 4, we add 3, remove 2, leaving 1, then add 1, going to 2 items, remove 1, going back down to 1 item, then add 3, going to 4 items.

Grading: 3 pts all or nothing, no reason needed.