3. Circular Queue

TITLE:

Implement Circular Queue using Array. Perform following operations on it.

a) Insertion (Enqueue)

b) Deletion (Dequeue)

c) Display

(Note: Handle queue full condition by considering a fixed size of a queue.)

OBJECTIVE:

1) To understand the concept of Queue.

2) To understand the concept of Circular Queue.

3) How data structures Queue is represented as an ADT.

Theory:

1) What is Queue? Explain Queue operations with neat diagrams.

A queue is a particular kind of collection in which the entities in the collection are kept in order and the principal (or only) operations on the collection are the addition of entities to the rear terminal position and removal of entities from the front terminal position. This makes the queue a First-In-First Out (FIFO) data structure. In a FIFO data structure, the first element added to the queue will be the first one to be removed. This is equivalent to the requirement that once an element is added, all elements that were added before have to be removed before the new element can be invoked. A queue is an example of a linear data structure. Queues provide services in computer science, transport, and operations research where various entities such as data, objects, persons, or events are stored and held to be processed later. In these contexts, the queue performs the function of a buffer. Queues are common in computer programs, where they are implemented as data structures coupled with access routines, as an abstract data structure or in object-oriented languages as classes. Common implementations are circular buffers and linked lists. Queue is a data structure that maintains "First In First Out" (FIFO) order. And can be viewed as people queueing up to buy a ticket. In programming, queue is usually used as a data structure for BFS (Breadth First Search)

Operations on queue:

1. enqueue - insert item at the back of queue Q
2. dequeue - return (and virtually remove) the front item from queue Q
3. init - intialize queue Q, reset all variables.

2) Explain how Queue can be implemented as an ADT?

Theoretically, one characteristic of a queue is that it does not have a specific capacity. Regardless of how many elements are already contained, a new element can always be

added. It can also be empty, at which point removing an element will be impossible until a new element has been added again. A practical implementation of a queue, e.g. with pointers, of course does have some capacity limit, that depends on the concrete situation it is used in. For a data structure.the executing computer will eventually run out of memory, thus limiting the queue size. Queue

overflow results from trying to add an element onto a full queue and queue underflow happens when trying to remove an element from an empty queue. A bounded queue is a queue limited to a fixed number of items.

3) What is Circular Queue ? Explain with example.

In a standard queue data structure re-buffering problem occurs for each dequeue operation. To solve this problem by joining the front and rear ends of a queue to make the queue as a circular queue.

Circular queue is a linear data structure. It follows FIFO principle.

- In circular queue the last node is connected back to the first node to make a circle.
- Circular linked list fallow the First In First Out principle
- Elements are added at the rear end and the elements are deleted at front end of the queue
- Both the front and the rear pointers points to the beginning of the array. · It is also called as ―Ring buffer‖.
- Items can inserted and deleted from a queue in O(1) time.

Circular Queue can be created in three ways they are-

· Using single linked list

· Using double linked list

· Using arrays

Using single linked list:

It is an extension for the basic single linked list. In circular linked list Instead of storing a Null value in the last node of a single linked list, store the address of the 1st node (root) forms a circular linked list.

Using circular linked list it is possible to directly traverse to the first node after reaching the last node.

Using double linked list

In double linked list the right side pointer points to the next node address or the address of first node and left side pointer points to the previous node address or the address of last node of a list. Hence the above list is known as circular double linked list.

Using array

In arrays the range of a subscript is 0 to n-1 where n is the maximum size. To make the array as a circular array by making the subscript 0 as the next address of the subscript n-1 by using the formula subscript = (subscript +1) % maximum size. In circular queue the front and rear pointer are updated by using the above formula.

ALGORITHM:

Enqueue operation using array

Step 1. start

Step 2. if (front == (rear+1)%max)

Print error —circular queue

overflow — Step 3. else

{ rear =

(rear+1)%max

Q[rear] =

element;

If (front == -1) f = 0;

}

Step 4. stop

Dequeue operation using array

Step 1. start

Step 2. if ((front == rear) && (rear

== -1)) Print error ―circular queue

underflow ― Step 3. else

{ element = Q[front]

If (front == rear)

front=rear = -1 Else

Front = (front + 1) % max

}

Step 4. stop

INPUT:

1. Queue Empty Display message-Queue EMPTY
2. Queue Full Display message-Queue FULL

OUTPUT:

Display elements of Circular queue.

CONCLUSION: In this way, we have studied implementation of circular queue using array.

FAQ:

1. What are the types of data structure?
2. What are the operations can implement on queue?
3. What is circular queue?