CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

QUEUE
(CAT-201)

Design By:
Ms. Gurpreet kaur dhiman
Ms.Mandeep kaur
Chandigarh University-Gharuan

QUEUE

- Queue is an abstract data structure, somewhat similar to Stacks. Unlike stacks, a queue is open at both its ends. One end is always used to insert data (en-queue) and the other is used to remove data (de-queue). Queue follows First-In-First-Out methodology, i.e., the data item stored first will be accessed first.
- A real-world example of queue can be a single-lane one-way road, where the vehicle enters first, exits first. More real-world examples can be seen as queues at the ticket windows and bus-stops.

QUEUE

- As we now understand that in queue, we access both ends for different reasons. The following diagram given below tries to explain queue representation as data structure −
- As in stacks, a queue can also be implemented using Arrays, Linked-lists, Pointers and Structures. For the sake of simplicity, we shall implement queues using one-dimensional array.

Reference:
https://www.tutorialspoint.com/data_structures_algorithms/dsa_queue.htm

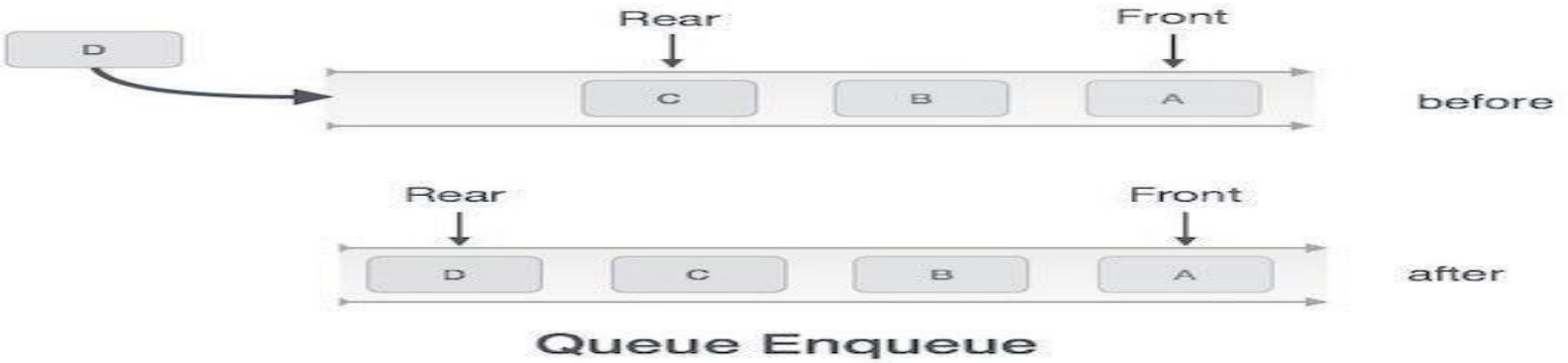
QUEUE OPERATION

Queue operations may involve initializing or defining the queue, utilizing it, and then completely erasing it from the memory. Here we shall try to understand the basic operations associated with queues −

- **En-queue()** − add (store) an item to the queue.
- **De-queue()** − remove (access) an item from the queue.
- Few more functions are required to make the above-mentioned queue operation efficient. These are −
- **peek()** − Gets the element at the front of the queue without removing it.
- **Is-full()** − Checks if the queue is full.
- **Is-empty()** − Checks if the queue is empty.
- In queue, we always de-queue (or access) data, pointed by **front** pointer and while en-queing (or storing) data in the queue we take help of **rear** pointer.

ENQUEUE

- Queues maintain two data pointers, **front** and **rear**. Therefore, its operations are comparatively difficult to implement than that of stacks. The following steps should be taken to en-queue (insert) data into a queue −
- **Step 1** − Check if the queue is full.
- **Step 2** − If the queue is full, produce overflow error and exit.
- **Step 3** − If the queue is not full, increment **rear** pointer to point the next empty space.
- **Step 4** − Add data element to the queue location, where the rear is pointing.
- **Step 5** − return success.



Queue Enqueue

DEQUEUE

- Accessing data from the queue is a process of two tasks − access the data where **front** is pointing and remove the data after access. The following steps are taken to perform **de-queue** operation −
- **Step 1** − Check if the queue is empty.
- **Step 2** − If the queue is empty, produce underflow error and exit.
- **Step 3** − If the queue is not empty, access the data where **front** is pointing.
- **Step 4** − Increment **front** pointer to point to the next available data element.
- **Step 5** − Return success.

Queue Dequeue

ENQUEUE PROCEDURE

CHANDIGARH UNIVERSITY

```
procedure en-queue(data)
if queue is full
return overflow
 end-if
rear ← rear + 1
queue[rear] ← data
 return true
end procedure
```

DEQUEUE PROCEDURE

procedure de-queue(data)

if queue is empty

return underflow

end-if

data ← queue[front]

front← front+1

return true

end procedure

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

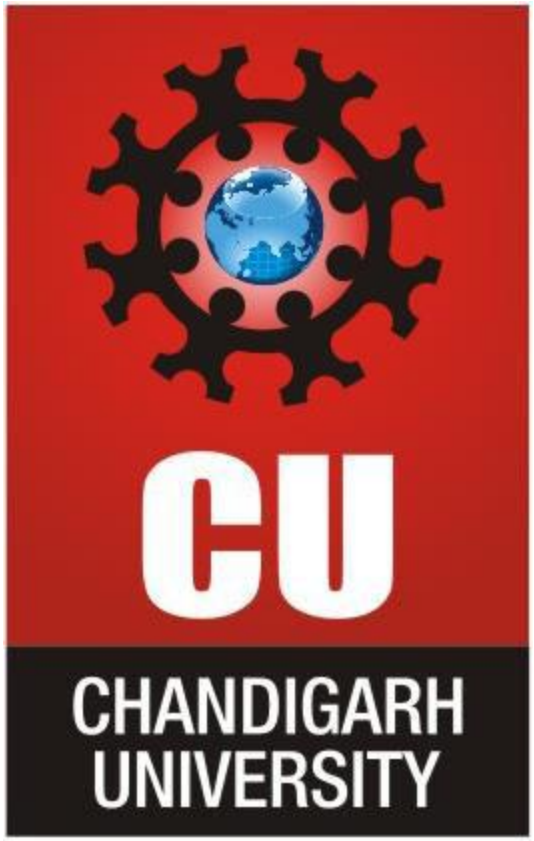
FAQ

- How elements are inserted into queue?
- In which order elements are deleted from queue?
- If queue front=-1and rear =-1 what does it means?

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Bibliography

- Seymour Lipschutz, Schaum's Outlines Series Data structures TMH
- Introduction to Data Structures Applications, Trembley&Soreson, Second Edition, Pearson Education
- www.geeksforgeeks.org/analysis-of-algorithms-set-3asymptotic-notations/
- www.tutorialspoint.com/data_structures_algorithms/asymptotic_analysis.htm



Thank You