Queue and Priority Queue

Lesson 5.2

Learning Objectives

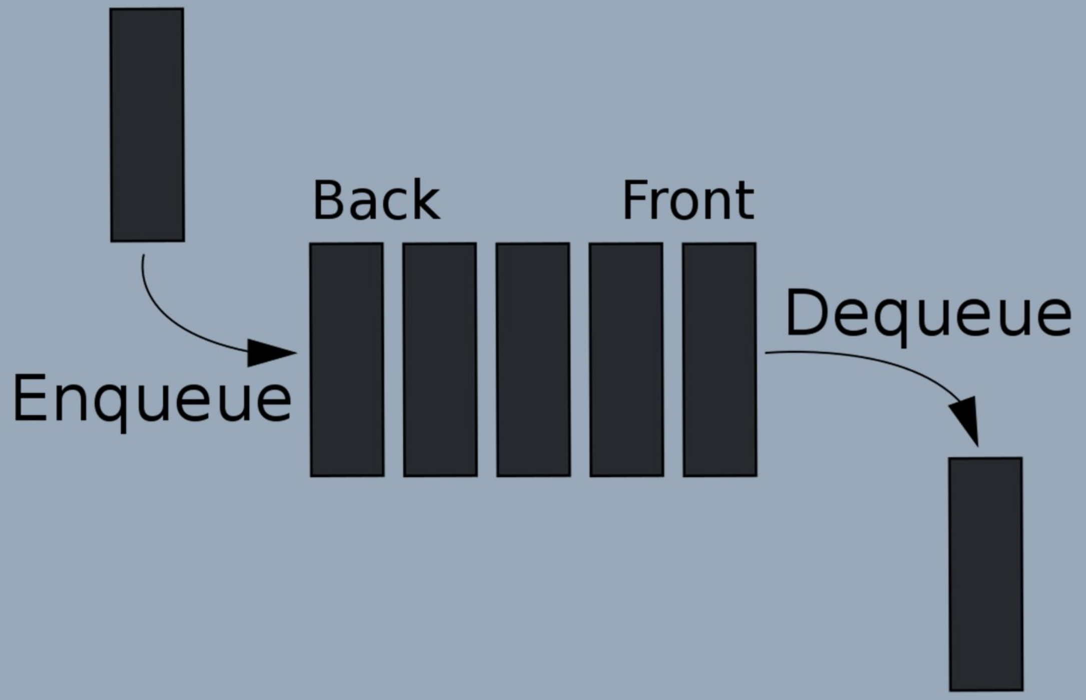
LO 5.2.1	**Enumerate** and **define** the basic operations of a queue ADT
LO 5.2.2	**Compute** the asymptotic complexities of queue operations
LO 5.2.3	**Assert** solutions on computing problems involving queue ADT

Queue

- Queue is an ADT that follows the principle **First In, First Out** (*FIFO*).
- Just like stack, it is *linear* in form thus it can either be implemented using either an array or any type of linked list.

Queue Operations

- The main operations on a queue are *enqueue* and *dequeue*
- **Enqueue** *inserts* the element at the *rear* of the queue
- **Dequeue** *removes/deletes* the element at the *front* of the queue



Queue Operations

- Some of the auxiliary operations of the queue include; but not limited to,
  - **GetFront** return the element at the front of the queue
  - **IsEmpty** returns 1 if the queue is empty and 0 otherwise
  - **IsFull** returns 1 if the queue is full and 0 otherwise

Priority Queue

- Priority queue is a queue that updates its order of elements based on prioritization
- A prioritization formula based on element attributes and/or values should be defined and this will be used during sorting the elements every after enqueue

Applications of Queue

- Managing requests on a single shared resource such as CPU scheduling and disk scheduling
- Handling hardware or real-time systems interrupts
- Handling website traffic
- Routers and switches in networking
- Maintaining the playlist in media players

Strengthening the Learning Objectives

LO 5.2.1 Enumerate and define the basic operations of a queue ADT

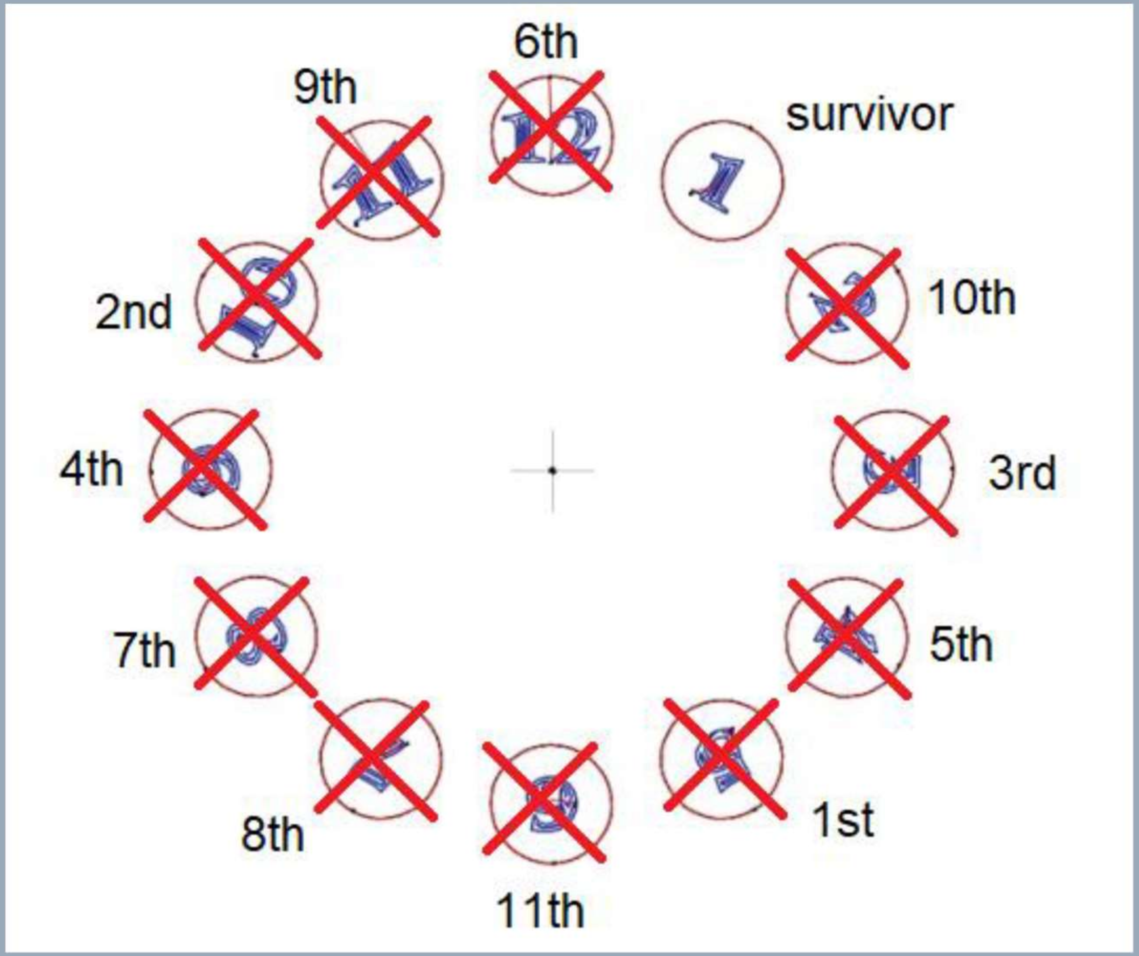
What are the basic operations of a queue ADT and what do they do?

LO 5.2.2 Compute the asymptotic complexities of queue operations

Construct a queue ADT structure in 2 implementations: using arrays and circular linked lists. Compute and compare the asymptotic complexity of all five (5) operations based on their most efficient implementations.

LO 5.2.3 Assert solutions on computing problems involving queue ADT

Implement the function `josephusProblem`. In the Josephus problem, *n* people arranged themselves in a circle, eliminating every *m*th person starting from the 1st person in succession until only 1 person is left. A demonstration on the problem (elimination) is shown on the immediate image on the right with n = 12 and m = 5.


6th
9th
survivor
2nd
10th
4th
3rd
7th
5th
8th
1st
11th