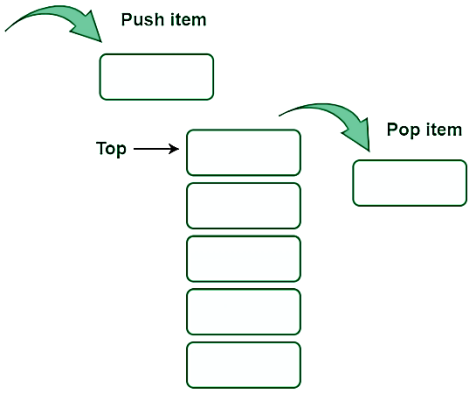
DATA STRUCTURE

SEMSTER - 2ND **ASSIGNMENT - 4** **COURSE - BSC – IT**

1. Define stack. What are the basic operations on stack

Ans - A stack is a linear data structure that follows the Last In, First Out (LIFO) principle. This means that the last element added to the stack will be the first one to be removed**.**



- **Basic operations:**

- **Push:** Add an element to the top of the stack.

- **Pop:** Remove the element from the top of the stack.

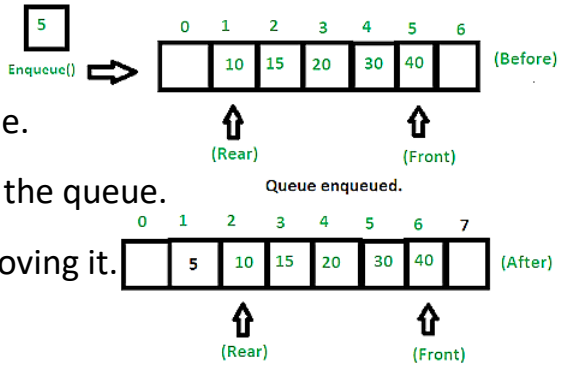
- **Peek/Top:** View the top element without removing it.

- **IsEmpty:** Check if the stack is empty.

- **IsFull:** Check if the stack is full (mainly in the case of a stack with a fixed size).

2. Define queue. What are the basic operations on queue?

Ans - A queue is a linear data structure that follows the First In, First Out (FIFO) principle. This means that the first element added to the queue will be the first one to be removed.



- **Basic operations:**

- **Enqueue:** Add an element to the end of the queue.

- **Dequeue**: Remove the element from the front of the queue.

- **Front/Peek:** View the front element without removing it.

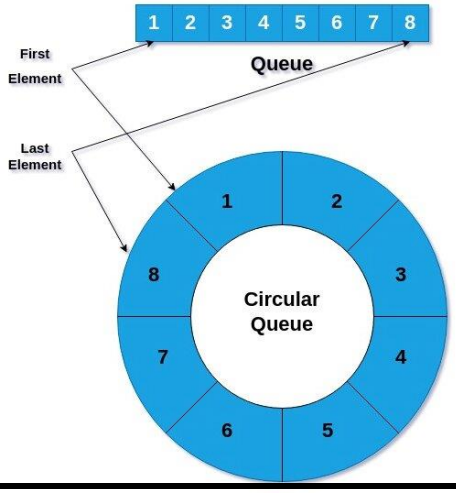
- **IsEmpty:** Check if the queue is empty.

- **IsFull:** Check if the queue is full (mainly in the case of a queue with a fixed size).

3. What is circular queue?

Ans –

A circular queue is a type of queue where the end of the line connects back to the front. Imagine a line of people that forms a circle.



4. **What are the applications of stack?**

Ans - - Function call management (call stack)

- Expression evaluation and syntax parsing (e.g., parsing expressions in compilers)

- Undo mechanisms in text editors

- Backtracking algorithms (e.g., solving mazes, puzzles)

- Depth-first search in graph algorithms

5. **Write the algorithm for following operations on queue:**
   i. **Enqueue ii. Dequeue**

Ans –

- **Insertion: enqueue()**

Step 1: START

Step 2: Check if the queue is full.

Step 3: If the queue is full, produce an overflow error and exit.

Step 4: If the queue is not full, increment the rear pointer to point to the next space.

Step 5: Add a data element to the queue location, where the rear is pointing.

Step 6: End the process and exit.

- **Deletion: dequeue()**

Step 1: START

Step 2: Check if the queue is empty.

Step 3: If the queue is empty, print underflow and exit.

Step 4: If the queue is not empty, access the data where the front is pointing.

Step 5: Increment the front pointer to point to the next available data element.

Step 6: Set the front and rear as -1 for the last element.

Step 7: End the process and exit.

CODE LINK : https://www.scholarhat.com/tutorial/datastructures/queue-data-structure-implementation

Disclaimer: **Answers are based on available data and calculations. We strive for accuracy but cannot guarantee it. Users should verify information independently. We are not responsible for any errors or outcomes.**

THANK YOU, CREATED BY SAURABH

All THE BEST