Lab3- CS1: **Implement a queue using linked list**

It is highly recommended that you complete it during the lab time. You need to submit it.

The main concept:

- Anything added to the linked list will be inserted at the end and anything deleted should be deleted from the front of the linked list. But, to add anyting at the end, traversing to the end every time is expensive/time consuming. So, we would like to avoid such traversal everytime we insert a new item. To solve the problem we will maintain two pointers:

  - One for front of the list
  - One for the back

Our struct to store the queue, would actually store two pointers to linked list structs:

- The first one would point to the head of the list.
- The second one would always point to the last node in that list

See the node structure and queue structure bellow:

```
// Stores one node of the linked list.
struct node {
    int data;
    struct node* next;
};
```

```
// Stores our queue.
struct queue {
    struct node* front;
    struct node* back;
};
```

Functions needed:

I. **init**: This function should make front and rear of the queue as NULL.

II. **enqueue**:

a) Create a new node and store the inserted value into it.
b) Link the back node's next pointer to this new node.
c) Move the back node to point to the newly added node.

III. **dequeue**:

a) Store a temporary pointer to the beginning of the list.
b) Move the front pointer to the next node in the list.
c) Free the memory pointed to by the temporary pointer.

IV. **front**:

Directly access the data stored in the first node through the front pointer to the list.

V. **empty**:

1. Check if both pointers (front, back) are null.