Assignment-3

Problem:

1.Impelement the queue.

2.Find the maximum element in the queue.

Solution:

I use doubly linked list for implement the queue.

I need a Node class, Queue class, some functions that queue has and main class.

Functions are: enqueue(int), dequeue(), size() , getFront(),getBack(),print() and findMax().

1.NODE CLASS

```
class Node {

    public:

    int data;
    Node* next;
    Node* prev;

};
```

2.QUEUE CLASS

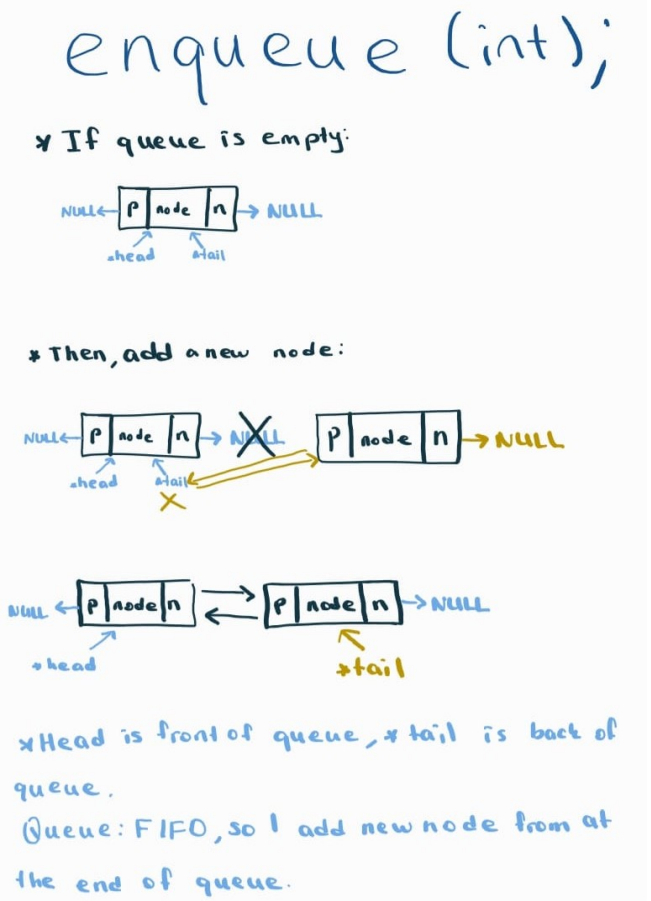
```
class Queue {

public:

    Queue();

    void enqueue(int);
    void dequeue();
    void print();
    bool empty();
    int size();
    int getFront();
    int getBack();
    int findMax(Queue Qa);


private:

    Node* head;
    Node* tail;

};

Queue::Queue() {

    head = NULL;
    tail = NULL;
}
```

I define the head pointer and the tail pointer for my queue. Firstly they are NULL. When we add new element they will not be a NULL.

3.FUNCTIONS (! p : prev, n : next)

-> enqueue(int)

```
void Queue::enqueue(int value){

    Node* node = new Node();
    node->data = value;

    if (empty() == false) {

        node->prev = tail;
        tail->next = node;
        node->next = NULL;
        tail = node;
    }
    else {

        node->next = NULL;
        node->prev = NULL;
        head = node;
        tail = node;
```



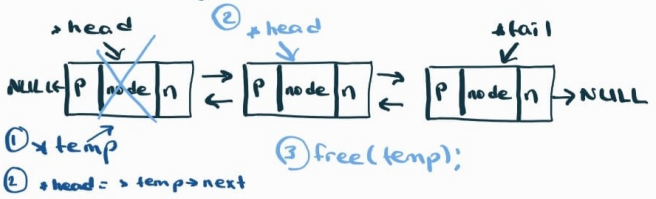
-> dequeue();

```
void Queue::dequeue() {

    if (empty() == false) {

        Node* temp = head;
        head = temp->next;
        head->prev = NULL;
        free(temp);

    }
    else {

        cout << "QUEUE IS EMPTY!";
    }
}
```

dequeue();

* If my queue is empty program will say "heey it is empty !!!"

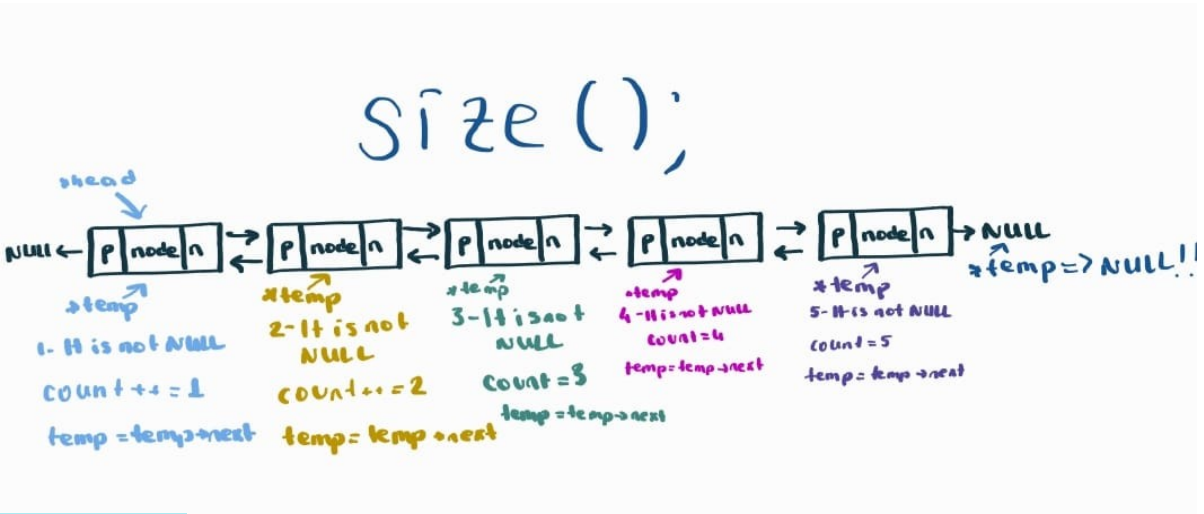
* If my queue is not empty



* In the queue nodes will be deleted at the beginning of queue.

-> size();

```
int Queue::size(){

    int count = 0;
    Node* temp = head;

    while (temp != NULL) {
        count++;
        temp = temp->next;
    }

    cout << count;
    return count;
}
```



-> getFront();

*I want to head of queue(list).

```
int Queue::getFront() {
    int a = head->data;

    return a;
}
```

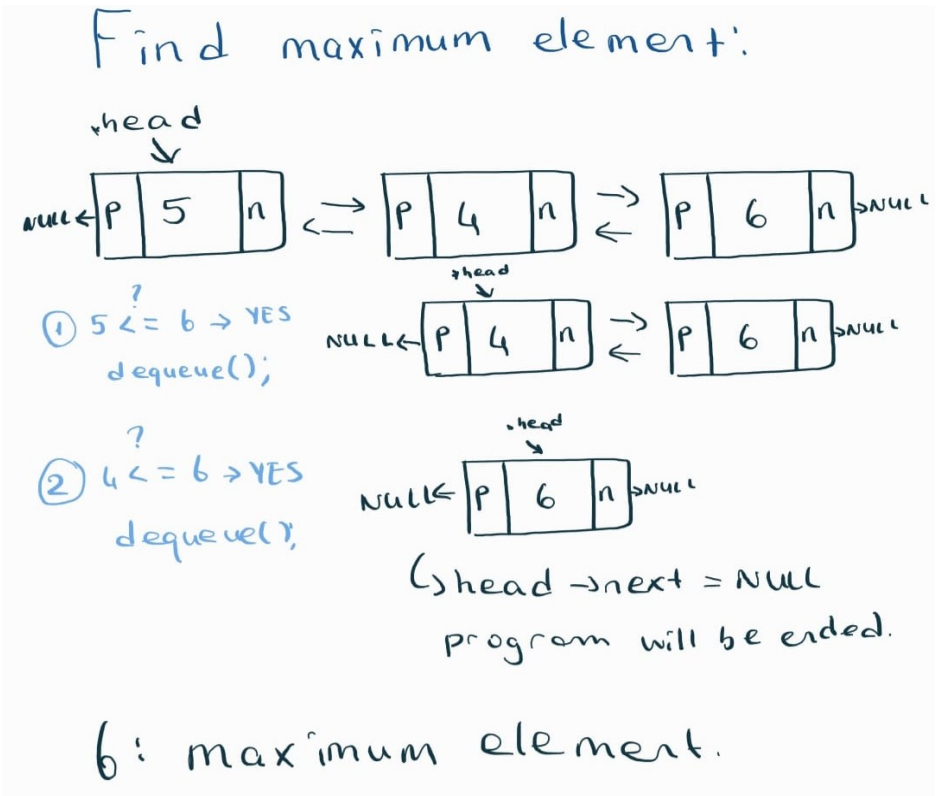
->getBack();

*I want to tail of queue(list).

```
int Queue::getBack() {

    int a = tail->data;

    return a;

    }
```

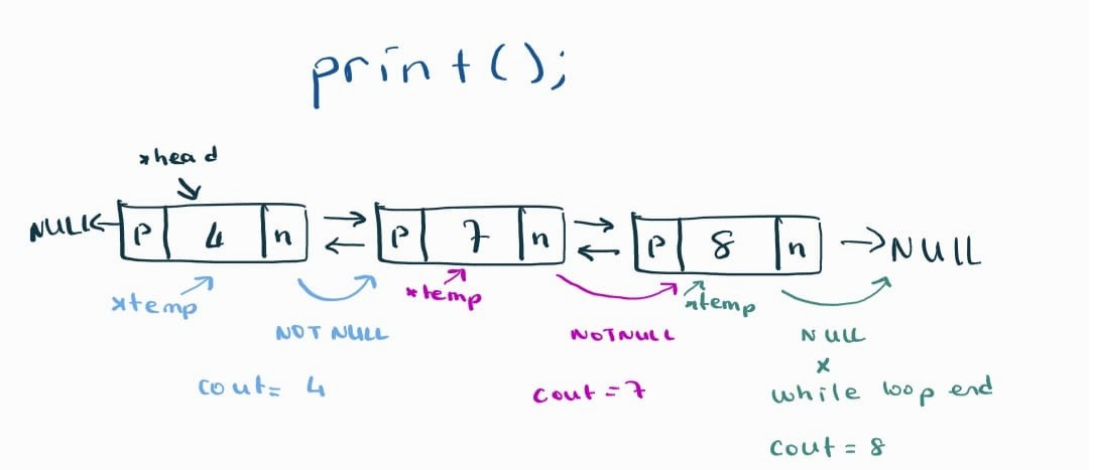
-> findMax();

```cpp
int Queue::findMax(Queue Qa){

    Queue Q = Qa;


    if (empty() == false) {

        while (head->next != NULL) {

            if (head->data <= tail->data) {

                dequeue();

            }
            else {

                enqueue(head->data);
                dequeue();


            }
        }

        return head->data;


    }
    else {

        cout << "QUEUE IS EMPTY!";
    }


}
```



->print();

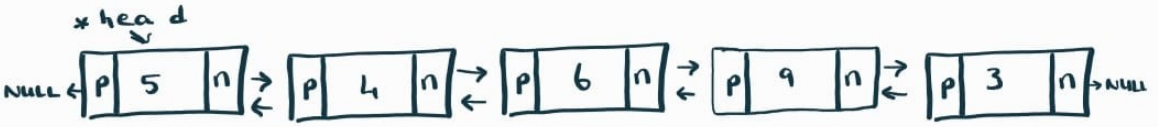
```
void Queue::print() {

    Node* temp = head;

    while(temp->next != NULL) {

        cout << temp->data << " ";
        temp = temp->next;

    }

    cout << temp->data << " ";

}
```



main();

```
int main () {

    Queue Q1;

    cout << "QUEUE :";
    Q1.enqueue(5);
    Q1.enqueue(4);
    Q1.enqueue(6);
    Q1.enqueue(9);
    Q1.enqueue(3);
    cout << Q1.print(); << "\n";
    cout << "size:" Q1.size(); << "\n";
    cout << "Rair:" Q1.getBack();<< "\n";
    cout <<"Front: "Q1.getFront(); <<"\n";
    cout << "Max of queue is :";
    cout << Q1.findMax(Q1) << endl;



    return 0;
}
```

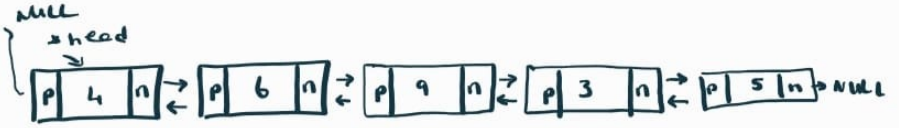
findMax(01) in Main



1- 5 <= 3 ? → NO

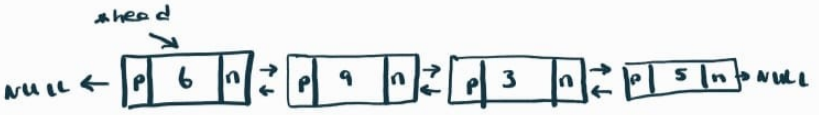
→ enqueue(5);

→ dequeue();



2- 4 <= 5 ? → YES

→ dequeue();



3 - 6 <= 5 ? → NO

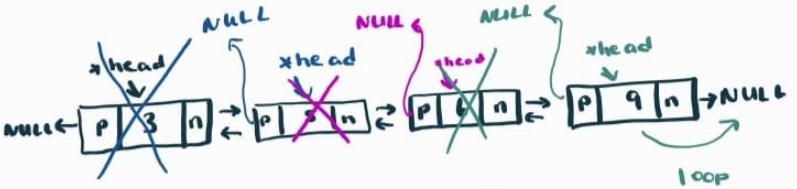
→ enqueue(6);

→ dequeue();

4 - 9 <= 6 ? → NO

→ enqueue(9);

→ dequeue();



loop will be ended

5 - 3 <= 9 ? → YES

→ dequeue();

6 - 5 <= 9 ? → YES

→ dequeue();

7 - 6 <= 9 ? → YES

→ dequeue();

MAXIMUM ELEMENT IS: 9