BMI 2123 DATA STRUCTURES

ASSIGNMENT 3/Solution paper

Important statement

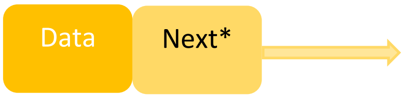
-Especially in the output simulation I used parentheses for some explanations, normally we don't see the text in parentheses as output on the screen

-I implemented front() and size() functions to getMax() function, so i didnt rewrite these functions in the main function.

Import <iostream> library

Define Node structure

- Int data
- Pointer of next node



Define Queue structure

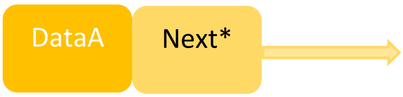
- Head pointer
- Queue constructure

*head
NULL

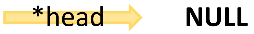
- Functions of Queue:
    - void enqueue(int data);
    - void dequeue();
    - int size();
    - int front();
    - void empty();
    - void getMax(Queue queue);
    - void printQueueItems();

Explanation of enqueue(int data): This function for add a new node at the queue’s end.

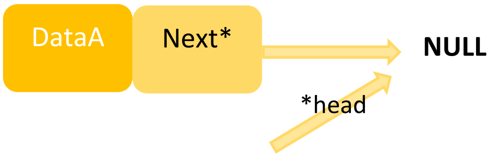
- Create a node
- Insert data

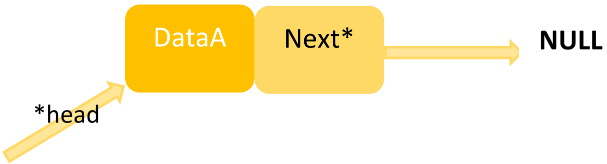


A1)If queue is empty,



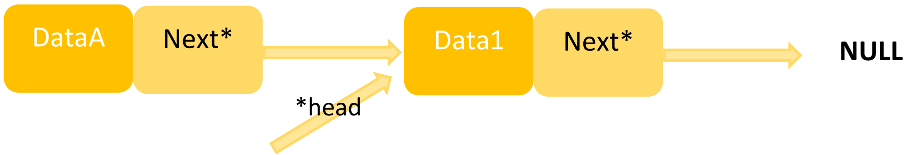
Node’s next is NULL





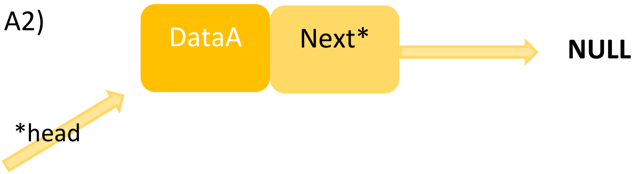
B1)If queue is not empty,(for example queue has an element)

Node’n next is equal to head

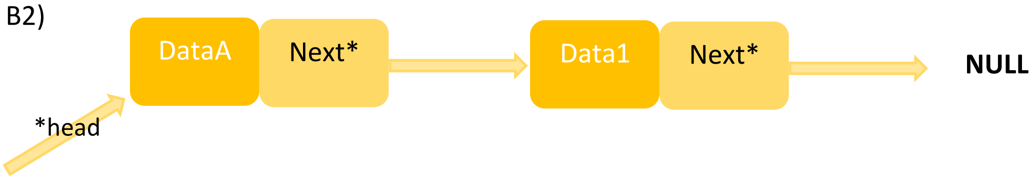


New head is equal to node

A2)

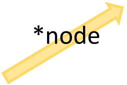


B2)

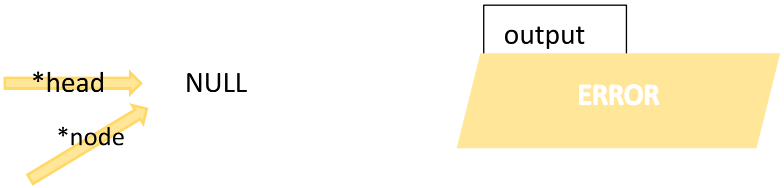


Explanation of dequeue(): This function for remove a node from front of the queue.

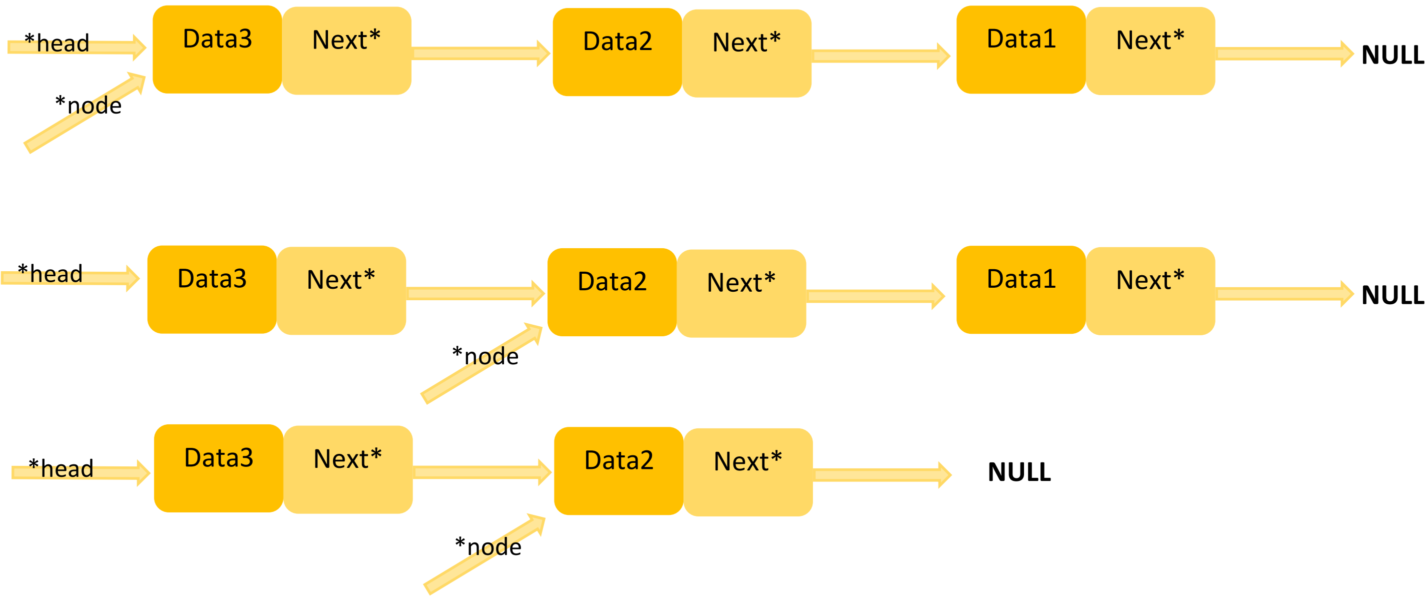
Enter the queue



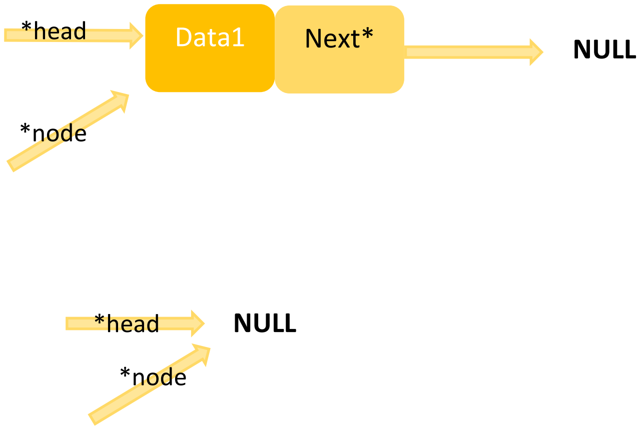
A)If queue is empty



B)If queue has least two elements

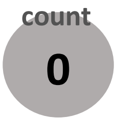


C)If queue has only one element



Explanation of size(): This function returns queue's length/ number of elements**.**

Define an int that name is count and count equal to "0(zero)"

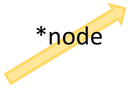


A)If queue is empty

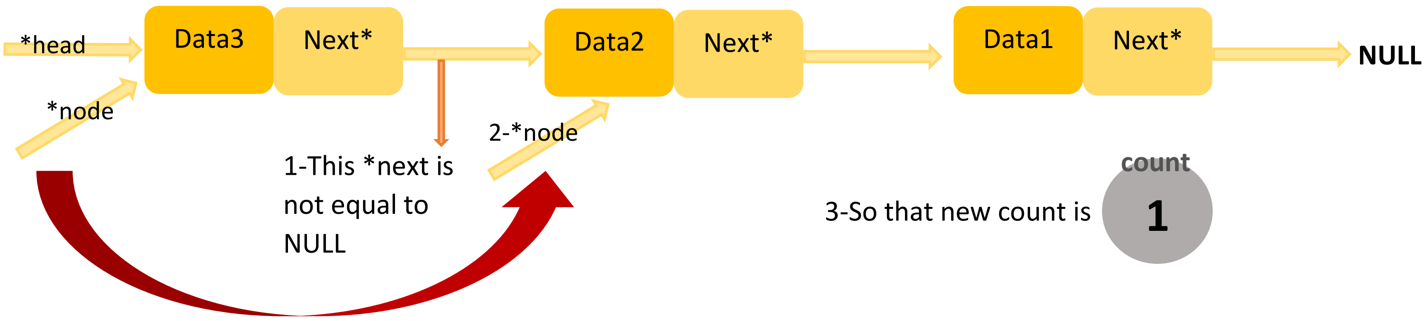
*head
NULL
return
0

B)If queue is not empty (for example queue has three elements)

Enter the queue

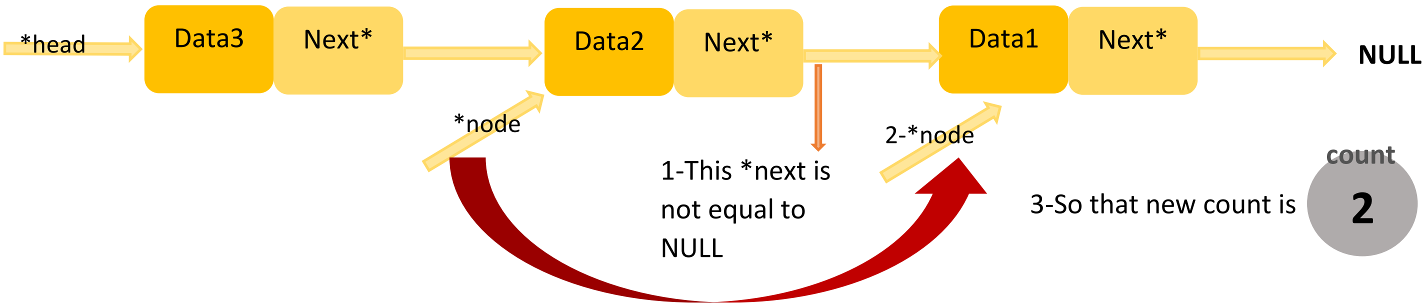


Sample queue



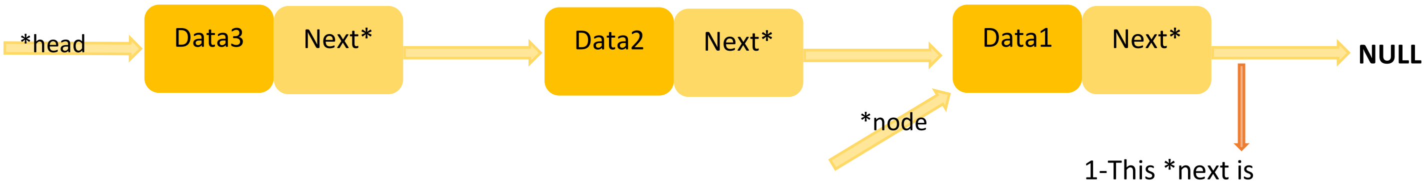
1-This *next is not equal to NULL

3-So that new count is 1



1-This *next is not equal to NULL

3-So that new count is 2



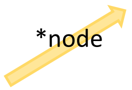
1-This *next is equal to NULL, so that while loop finished.

return

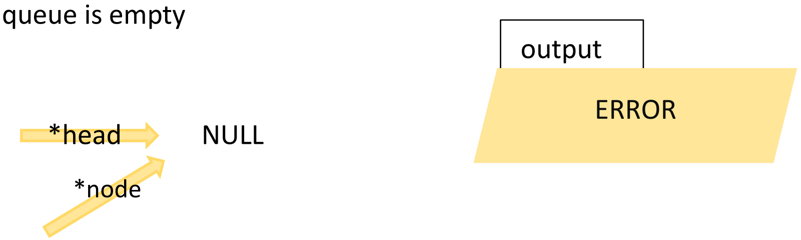
(Count+1 so) 3

Explanation of front(): This function returns queue’s front element

Enter the queue

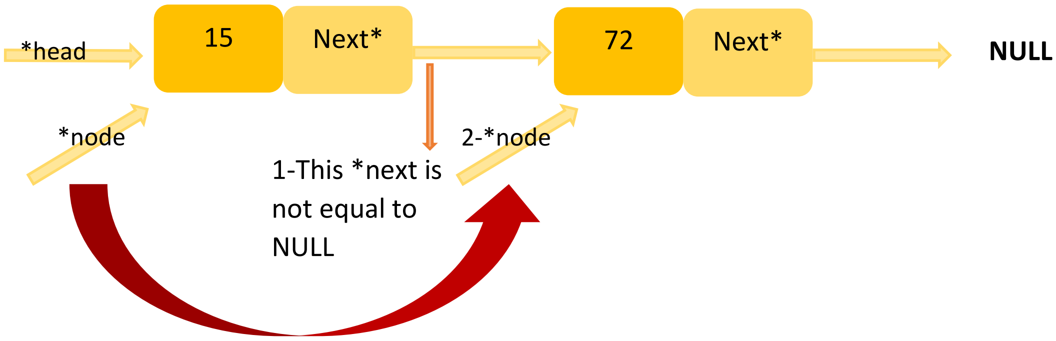


A)If queue is empty

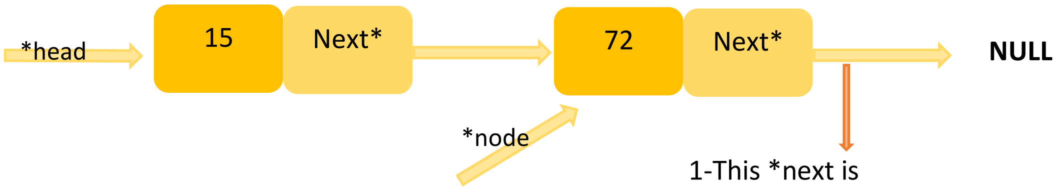


B)If queue is not empty(for example queue has two elements)

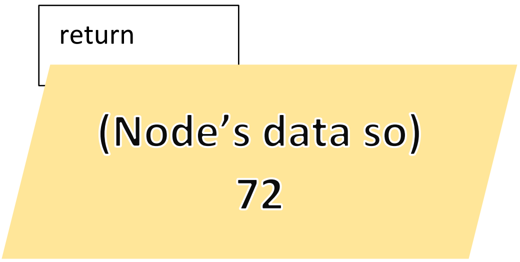
Sample queueu



1-This *next is not equal to NULL



1-This *next is equal to NULL, so that while loop finished



(Node’s data so)

72

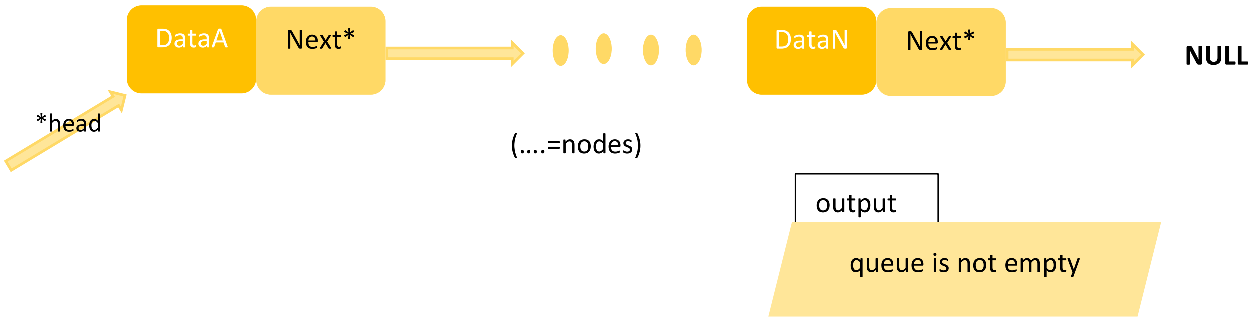
Explanation of empty(): This function displays us wheter queue is empty.

A)If head pointer’s equal to NULL, queue is empty and display on the screen “queue is empty”

*head
NULL
output

queue is empty

B)If head pointer’s not equal to NULL, , queue is not empty and display on the screen “queue is not empty”



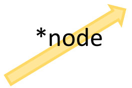
(….=nodes)

output

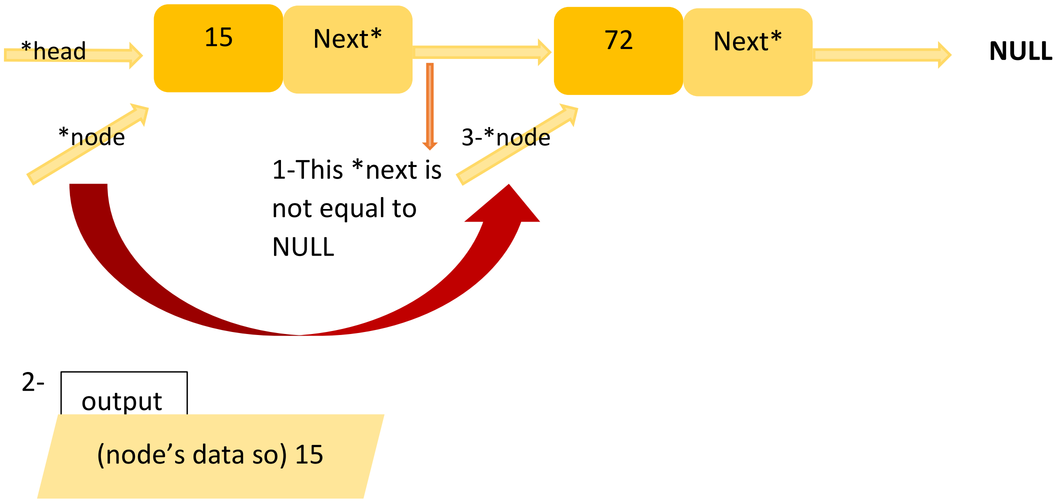
queue is not empty

Explanation of printQueueItems(): This function displays all the elements of the queue, where output is from rightmost to leftmost equal to queue's from first element to last element

Enter the queue



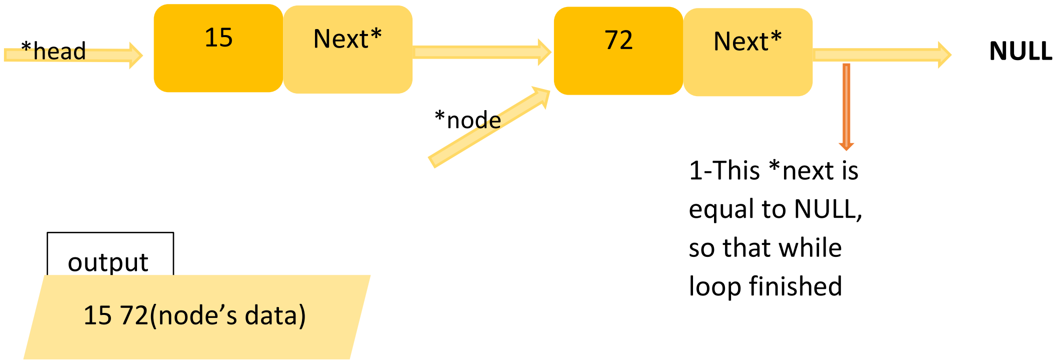
Sample queueu



1-This *next is not equal to NULL

2- output

(node’s data so) 15



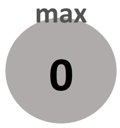
1-This *next is equal to NULL, so that while loop finished

output

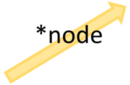
15 72(node’s data)

Explanation of getMax(Queue queue): This function displays the maximum element of a given queue.

Define an int that name is max and max equal to “0(zero)”



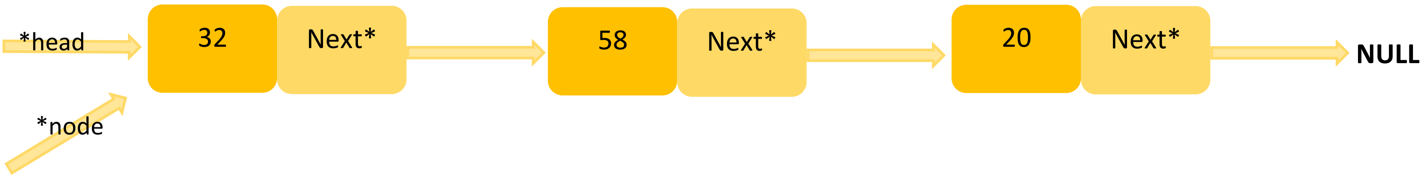
Enter the queue



A)If head pointer’s equal to NULL, queue is empty and display on the screen “ERROR”

*head
NULL
output
ERROR

B) If head pointer’s not equal to NULL, queue is not empty. For example, let our given queue be as follows



If queue’s size not equal to 0, enter the while loop

If queue’s front element is bigger than max ,max equal to queue’s front element,after that loop’s next step remove the front element with dequeue function.

I queue’s front element is not bigger that max, max keeps same. After that loop’s next step remove the front element with dequeue function

So;

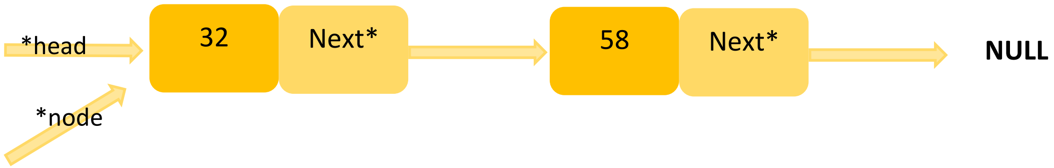
q.size()=3 , enter the loop first time

For now q.front()=20

20 is bigger than max=0

Now max=20

and our queue as follows



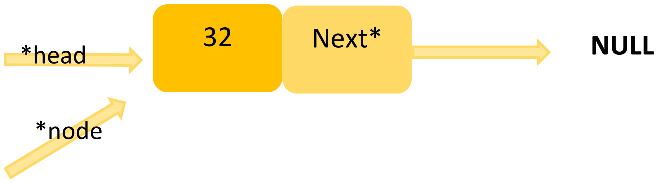
q.size()=2, enter the loop second time

For now q.front()=58

58 is bigger than max=20

Now max=58

and our queue as follows



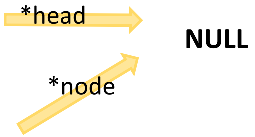
q.size()=1, enter the loop third time

For now q.front()=32

32 is not bigger than max=58

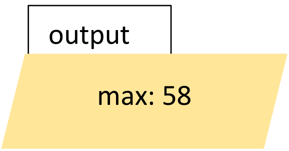
Now max is not change , max=58

and our queue as follows



q.size()=0, don't enter the loop,and go on next line.

Display the max element of queue.



Explanation of main(): Below I will simulate the working stages of the main function

I create an empty queue that thype is queue

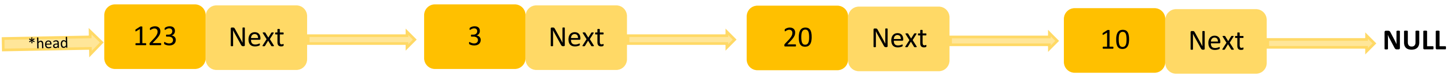
Ask the queue, Is the queue is empty with empty() function.

We expect the output that displays “queue is empty”. Because the queue has no element.

output

```
queue is empty
```

We add elements 10,20,3 and 123 to the queue sequentially used to enqueue(int data) function.



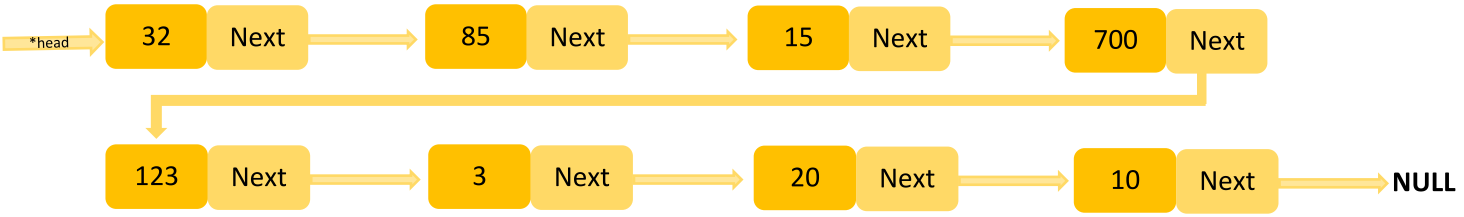
Ask the queue, Is the queue is empty with empty() function.

We expect the output that displays “queue is not empty”. Because the queue has more than 0 element.

output

```
queue is not empty
```

We add elements 700,15,85 and 32 to the queue sequentially used to enqueue(int data) function.



We print all element’s of the queue on the screen used to printQueuesItems() function.

output

```
32 85 15 700 123 3 20 10
```

We remove an element from the queue using the dequeue() function. We expect the function is remove the queue’s front element so “10”.

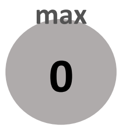
Again we print all element’s of the queue on the screen used to printQueuesItems() function.

output

```
32 85 15 700 123 3 20
```

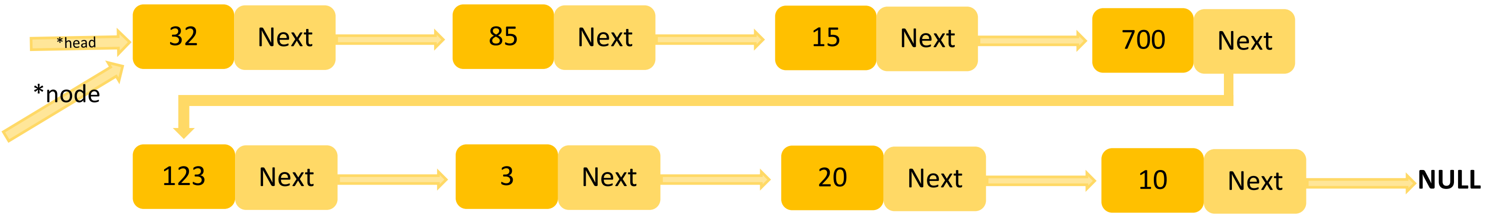
We use the getMax function to find the maximum element of the queue

Define an int that name is max and max equal to “0(zero)”



Enter the queue

Head is not equal to NULL, so that g oto else part,



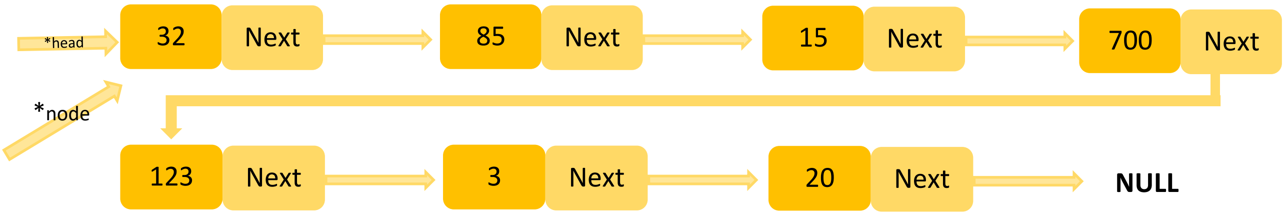
q.size()=8 , enter the loop first time

For now q.front()=10

10 is bigger than max=0

Now max=10

and our queue as follows



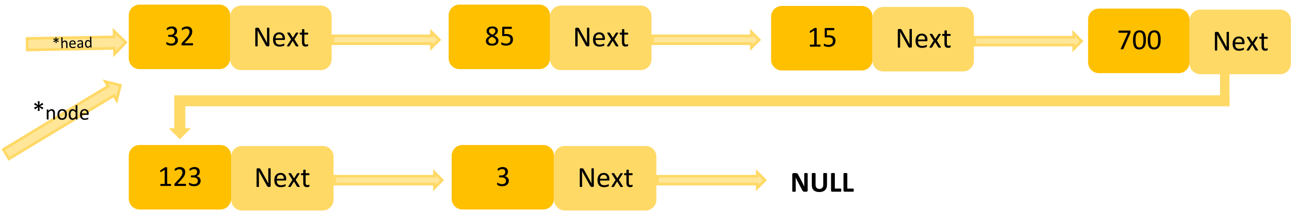
q.size()=7, enter the loop second time

For now q.front()=20

20 is bigger than max=10

Now max=20

and our queue as follows



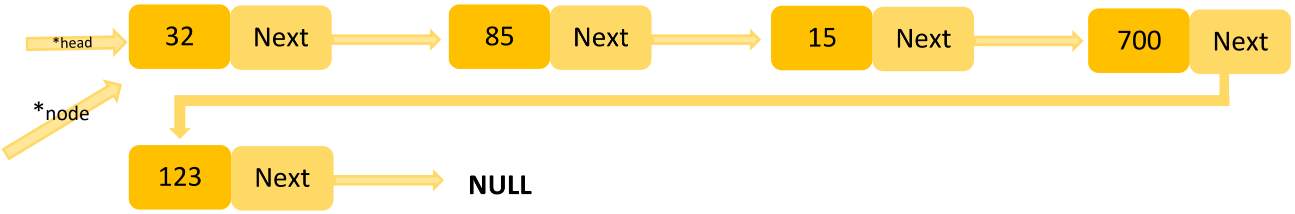
q.size()=6, enter the loop third time

For now q.front()=3

3 is not bigger than max=20

Now max is not change , max=20

and our queue as follows



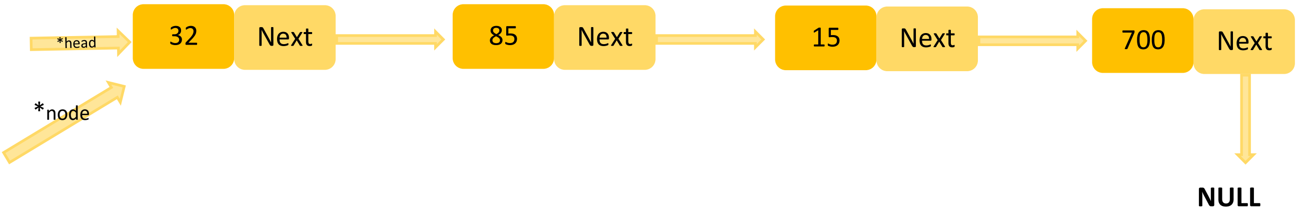
q.size()=5, enter the loop 4th time

For now q.front()=123

123 is bigger than max=20

Now max=123

and our queue as follows



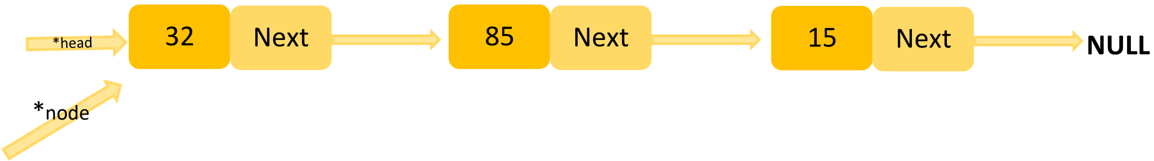
q.size()=4, enter the loop 5th time

For now q.front()=700

700 is bigger than max=123

Now max=700

and our queue as follows



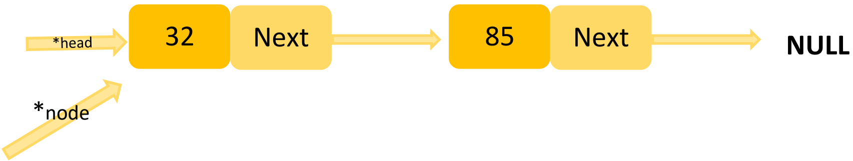
q.size()=3, enter the loop 6th time

For now q.front()=15

15 is not bigger than max=700

Now max is not change , max=700

and our queue as follows



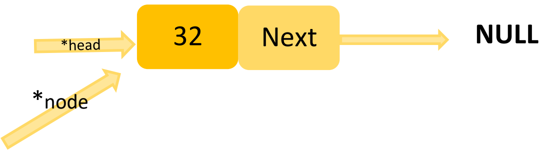
q.size()=2, enter the loop 7th time

For now q.front()=85

85 is not bigger than max=700

Now max is not change , max=700

and our queue as follows



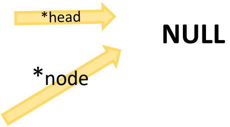
q.size()=1, enter the loop 8th time

For now q.front()=32

32 is not bigger than max=700

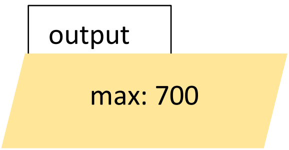
Now max is not change , max=700

and our queue as follows



q.size()=0, don’t enter the loop,and go on next line.

Display the max element of queue.



```
max: 700
```