Data Structures

Circular Queue – Linked list

Team Emertxe

Algorithm -dequeue

Data Structure – Circular Queue

dequeue

Algorithm :dequeue(front,rear)

Input Specification :

- front : Pointer contains first node address
- rear : Pointer contains last node address

Output Specification :

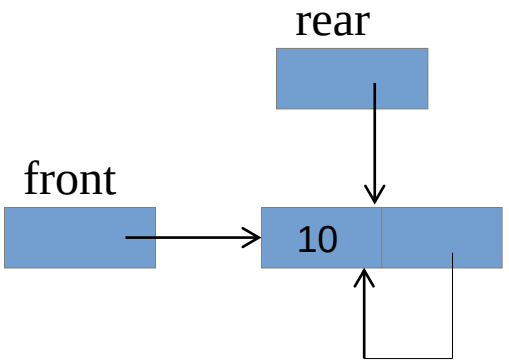
- Status: e_true/e_false

Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶ link
   free( rear ⟶ link)
   rear ⟶ link = front
return e_true
```

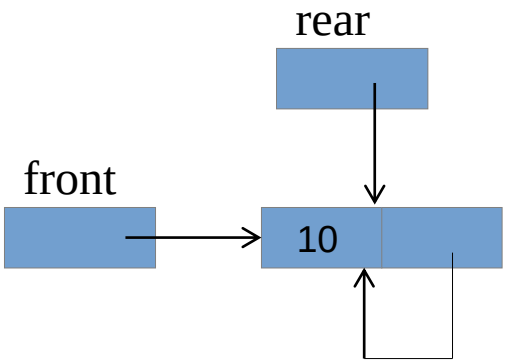
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



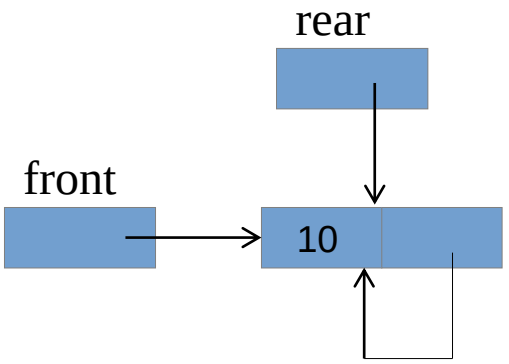
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



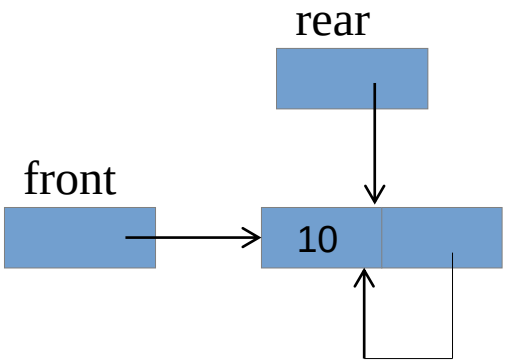
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



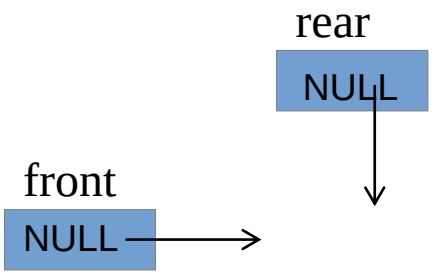
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```

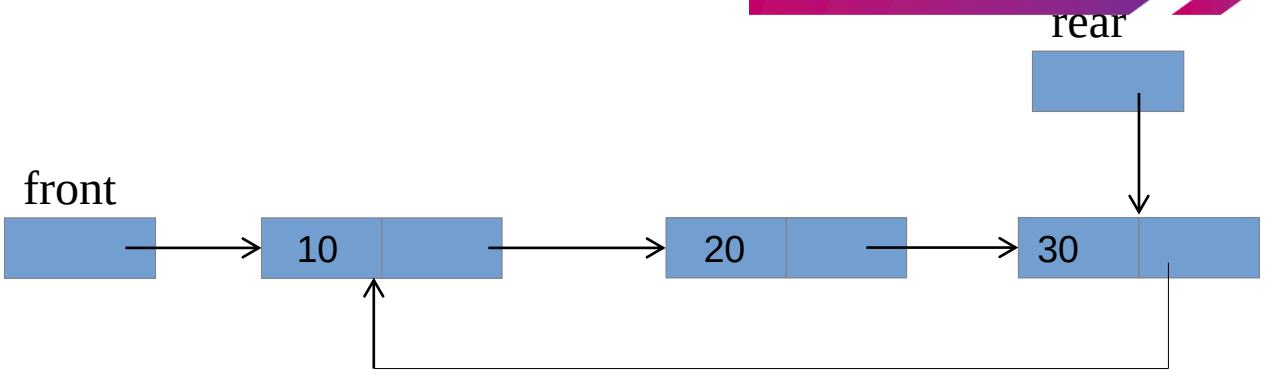


Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```

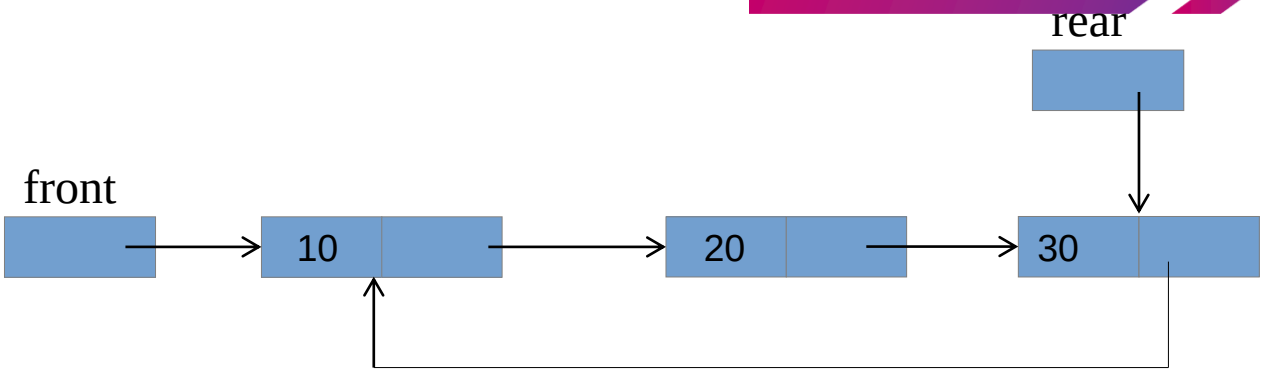
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



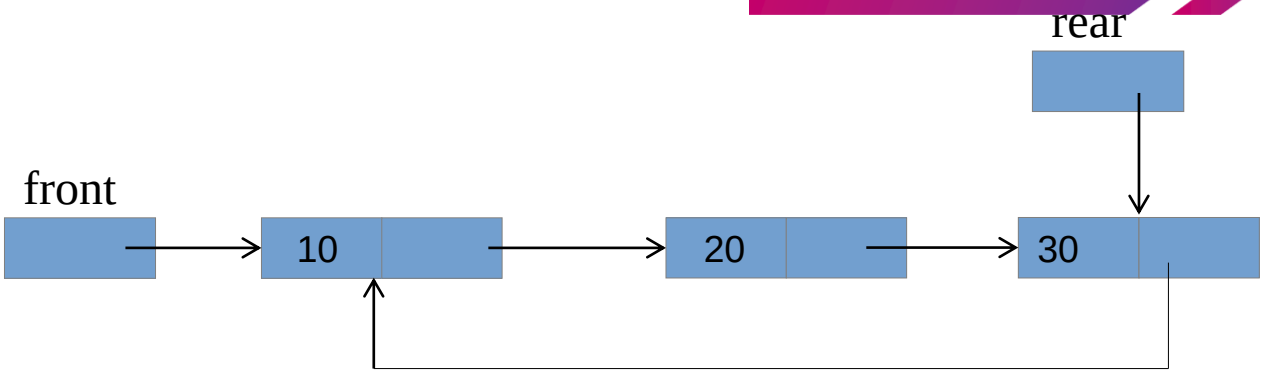
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



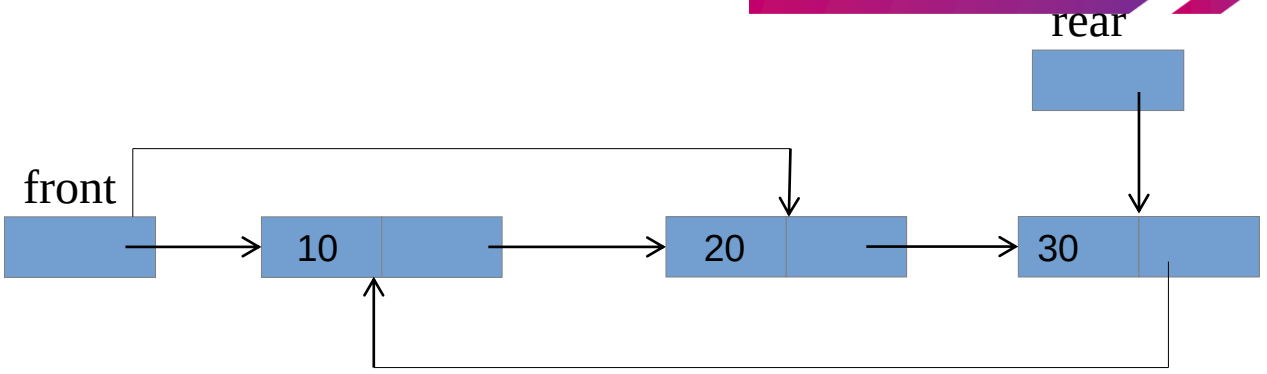
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
    front = front ⟶link
    free( rear⟶link)
    rear⟶link = front
return e_true
```



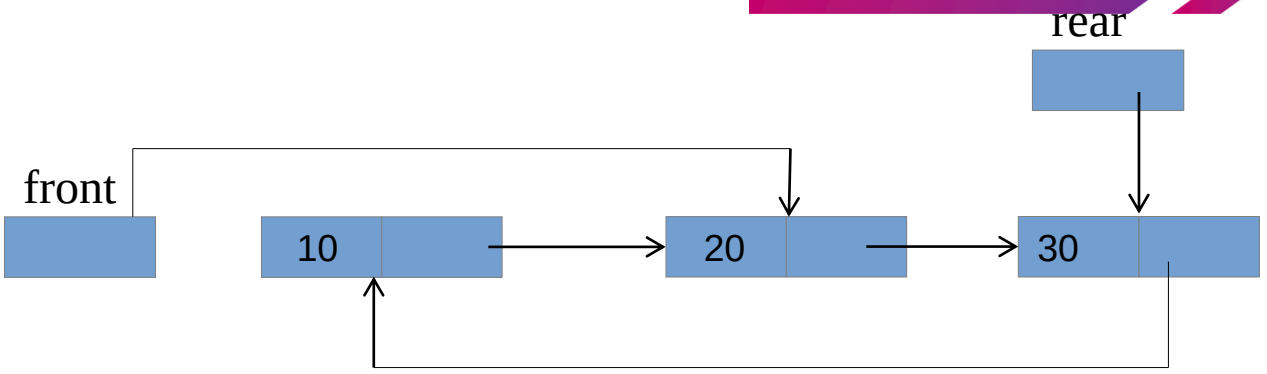
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



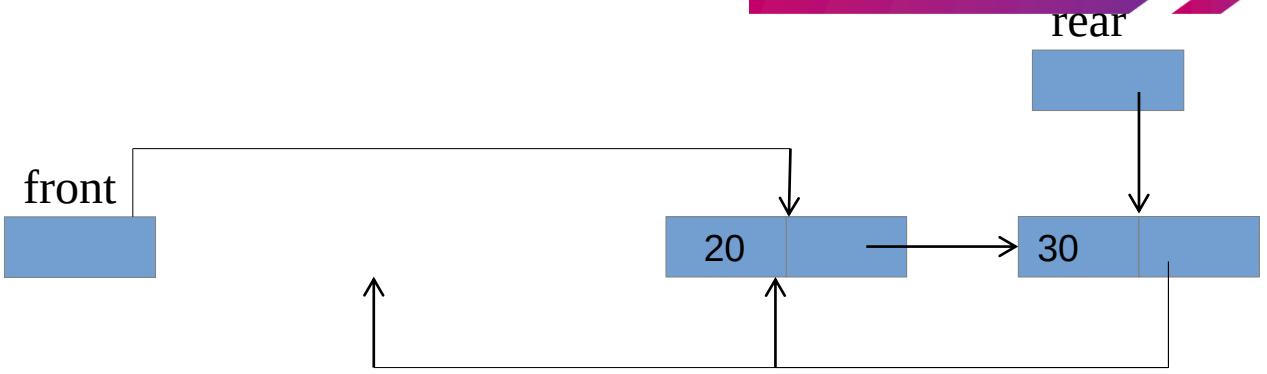
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



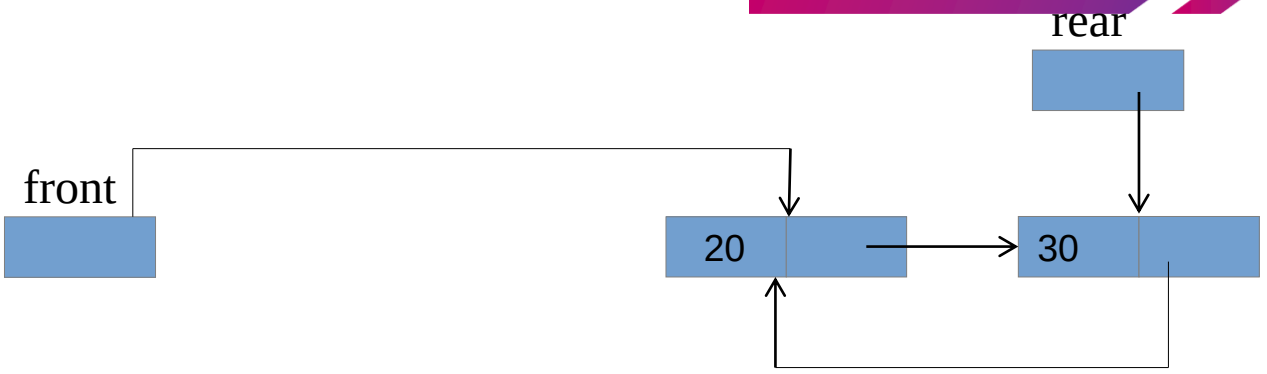
Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



Algorithm

```
if ( front = NULL)
    return e_false
if( front = rear )
    free( front )
    front = rear = NULL
else
   front = front ⟶link
   free( rear⟶link)
   rear⟶link = front
return e_true
```



Code -Circular Queue –Linked List