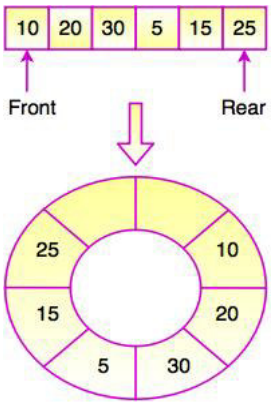
UNIVERSITY OF DIYALA

Data Structure

Lecture 4:
Linear Data Structure: The Queue

ASST. INSTRUCTOR
ALI A. AL-ANI



Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

LDS: The Queue

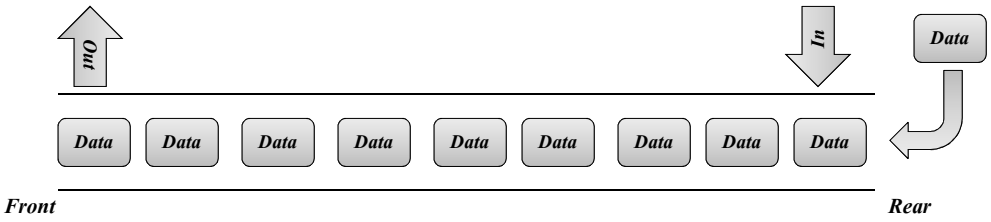
- *Queue is a linear list of elements , somewhat similar to Stacks. Unlike stacks, a queue is open at both its ends. One end is always used to insert data (**Enqueue**) and the other is used to remove data (**Dequeue**). Queue follows **First-In-First-Out** methodology, i.e., the data item stored first will be accessed first.*
- *A real-world example of queue can be a single-lane one-way road, where the vehicle enters first, exits first. More real-world examples can be seen as queues at the ticket windows and bus-stops.*

Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

LDS: The Queue

- *A queue is a linear list of elements in which deletions can place only at one end, called the **front**, and insertions can take place only at the other end, called the **rear**. The terms "**front**" and "**rear**" are used in describing a linear list only when it is implemented as a queue. The figure illustrates a queue containing Data elements.*



UNIVERSITY OF DIYALA

LDS: Queue Representation

- *Since a queue is a linear data structure, any linear data structure implementation will do. As in stacks, a queue can also be implemented using **Arrays**, **Linked-lists**.*

1. ***Non-linked- structures (Array):** The simplest method to represent a queue is to use one-dimensional array. The queue may therefore be declared and containing **three objects**: an array with suitable size and with suitable data type (Int, Float,..etc) to hold the elements of the queue, and two integers to indicate the position of the **front** and the **rear** within the array. Ex: The below declaration example in the C++ language*

const SIZE= 10; Int queue[SIZE]; Int rear = -1, front = -1; // -1 mean the queue is empty.

UNIVERSITY OF DIYALA

LDS: Queue Operations

- *Queue operations may involve initializing or defining the queue, utilizing it, and then completely erasing it from the memory. The basic operations associated with queues are:*

1. ***Enqueue() − add (store) an item to the queue.***
2. ***Dequeue() − remove (access) an item from the queue****.*

- *Few more functions are required to make the above-mentioned queue operation efficient. These are:*

1. ***isfull() − Checks if the queue is full.***
2. ***isempty() − Checks if the queue is empty.***

- *In queue, we always dequeue (or access) data, pointed by **front** pointer and while enqueing (or storing) data in the queue we take help of **rear** pointer.*

UNIVERSITY OF DIYALA

LDS: Queue (Enqueue)

- ***Steps for inserting an Element in a Queue (Enqueue):***

1. *Initialize both the front and rear as **-1**, which means that the queue is empty.*
2. *When the first element will be inserted then the **rear** and **front** both will be incremented by **0**. But for the second element onwards only the **rear** will be incremented.*
3. *The value of **rear** can be maximum up to **Size-1**, where **Size** is the maximum number of elements a queue can hold.*
4. *If the **rear** reaches **Size-1**, then display a message that **"The queue is full or Queue Overflow"**.*

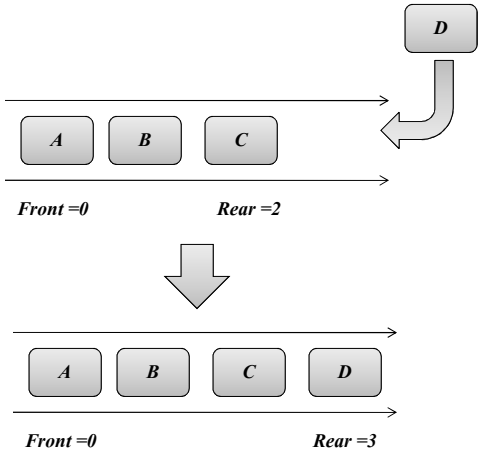
Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

LDS: Queue (Enqueue)

- ***Sub program to add new element to the queue***

```
void Enqueue(int item)
{
 if (fullqueue())
{ cout<<"error...the queue is full"<<endl;
  cout<<"press any key to exit"<<endl;
getch();   exit(0); }
Else  {
rear=rear+1;
queue[rear]=item;  }  }
```



UNIVERSITY OF DIYALA

LDS: Queue (Dequeue)

- ***Steps for deleting an Element from the Queue (Dequeue):***
- *In a queue, delete operation takes place at **front** end. The "Dequeue" operation removes the item at the "**front**" of the queue and returns it.*

1. *Check if the queue is empty **(Front and Rear = -1**). If the queue is empty, produce underflow error and exit.*
2. *When an element will be deleted from the queue the value of **front** will be Incremented by **1**.*
3. *The value of **front** can be Maximize up to **Size-1**.*
4. *If the **front** reaches **-1**, then display a message that "**The queue is empty or Queue Underflow**".*

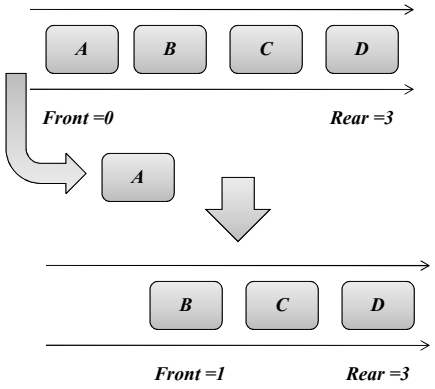
Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

LDS: Queue (Dequeue)

- ***Sub program to delete an element from the queue***

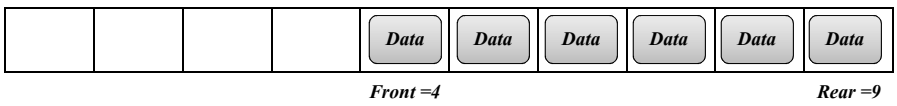
```
void Dequeue()
{
if(emptyqueue())
{  cout<<"error…the queue is empty"<<endl;
cout<<"press any key to exit"<<endl;
getch();  exit(0);  }
Else  {
item=queue[front];
front=front+1;  }  }
```



UNIVERSITY OF DIYALA

LDS: The Queue

- ***Linear Queue Problem:*** *Arrays have fixed sizes. Therefore, after various insertion and deletion operations Queue (**Rear**) will point to the last array position as state in the following figure giving the impression that the queue is full but it is not (size of array =10).*

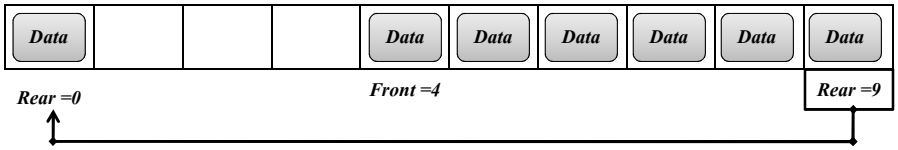


- *In the above situation queue overflow will occur though the free spaces are also available since the rear reaches the end. So, queue is full, but still there are two free cells are available in queue.*

UNIVERSITY OF DIYALA

LDS: Circular Queue

- *To avoid this problem, the queue can be arranged in a circular way called **circular queue**. In a circular queue as soon as the **rear** reaches the Max value (**Rear equal to Size-1**), it should be reset to **0**. So, if we have to insert **new data Item** in the above queue we can insert it into the first place.*



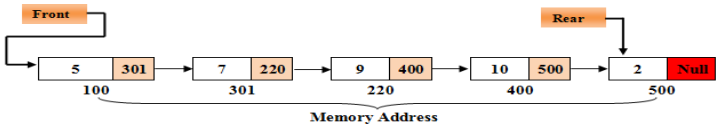
- *In the circular queue the Front and Rear arrows wrap around to the beginning of the array, the elements are arranged in a sequential manner but can logically be regarded as circularly arranged. The use of such a type of queue is to reuse memory locations.*

UNIVERSITY OF DIYALA

LDS: Queue Representation

2. ***Linked- structures (Linked List):** As we know if we implement the queue in array, sometimes number of memory location remains unused also the size of the array is fixed. To overcome these drawbacks, it is better to implement queue using linked list representation.*

- *A linked queue, performs insertions on one side and removals from the other. Therefore it is necessary to maintain a links to both the front and the back of the list. The figure shows the Linked List Implementation of queue. The **front** and **rear** are pointers to the front and rear queue elements, respectively.*



Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

LDS: Queue Representation

- *The below declaration example in the C++ language represent the linked queue by using a structure that contains a **value** and a **link** to its neighbor.*

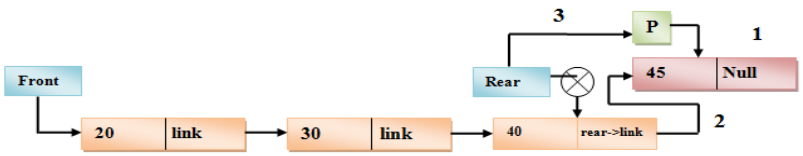
```
struct node{
int data;
struct node*link;
}*rear,*front,*p,*q;
```


UNIVERSITY OF DIYALA

LDS: Queue (Enqueue)

- ***Sub program to add new element to the linked queue***

```
void Enqueue ()
{ p = new node;
cout<<"input new element"<<endl;
cin>>p->data;
p->link=NULL;    //  1
if(rear==NULL)
front=p;
else rear->link=p;    //  2
rear=p;          //  3
}
```



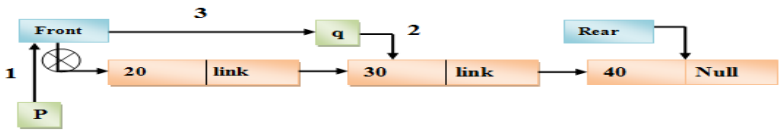
Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

LDS: Queue (Dequeue)

- *Sub program to delete an element from the linked queue*

```
void Dequeue ()
{  int item;
p = front;    //    1
if( p == NULL)  {  cout<<"error...the linked queue is empty"<<endl;
cout<<"press any key to exit"<<endl;   getch();   exit(0);   }
else  {  q=p->link;   //   2
item = p->data;
delete (p);
front = q;        //     3
if (front == NULL)
rear = NULL;   } }
```



UNIVERSITY OF DIYALA

LDS: Queue Applications

- *There are various queues quietly doing their job in your computer's (or the network's) operating system such as:*

I. *The printer queue where print jobs wait for the printer to be available Job scheduling (FIFO Scheduling)*

II. *Key board buffer, A queue also stores keystroke data as you type at the keyboard. This way, if you're using a word processor but the computer is briefly doing something else when you hit a key, the keystroke won't be lost; it waits in the queue until the word processor has to read it.*

Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

The End

Department of Computer Science
College of Science