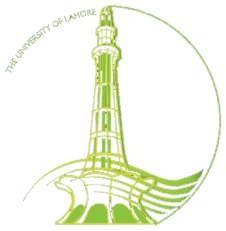

THE UNIVERSITY OF LAHORE
ISLAMABAD CAMPUS

DATA STRUCTURE

Lab Report

Name:	SAMI ULLAH
Registration #:	SEU-S17-030
Lab Report #:	01
Dated:	16-04-2018
Submitted To:	Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 2
Queue with Array implementation

Software Tools:-

1. **DEV C++**

Theory:-

Queue using Array: -

This manual discusses an important data structure, called a queue. The idea of a queue in computer science is the same as the idea of the queues to which you are accustomed in everyday life. There are queues of customers in a bank or in a grocery store and queues of cars waiting to pass through a tollbooth. Similarly, because a computer can send a print request faster than a printer can print, a queue of documents is often waiting to be printed at a printer. The general rule to process elements in a queue is that the customer at the front of the queue is served next and that when a new customer arrives, he or she stands at the end of the queue. That is, a queue is a First In First Out data structure.

A queue is a set of elements of the same type in which the elements are added at one end, called the back or rear, and deleted from the other end, called the front. For example, consider a line of customers in a bank, wherein the customers are waiting to withdraw/deposit money or to conduct some other business. Each new customer gets in the line at the rear. Whenever a teller is ready for a new customer, the customer at the front of the line is served.

The rear of the queue is accessed whenever a new element is added to the queue, and the front of the queue is accessed whenever an element is deleted from the queue. As in a stack, the middle elements of the queue are inaccessible, even if the queue elements are stored in an array.

Queue: A data structure in which the elements are added at one end, called the rear, and deleted from the other end, called the front; a First-In-First-Out (FIFO) data structure.

Queues may be represented in the computer in various ways, usually by means at one-way list or linear arrays. Unless otherwise stated or implied each of our queues will be maintained by a linear array QUEUE and two pointer variable FRONT containing the location of the front element of the queue and REAR containing the location of the rear element of the queue. The condition FRONT = NULL will indicate that the queue is empty.

CODE:

```
#include<iostream>
#include<conio.h>
#include<stdlib.h>
#define SIZE 5
using namespace std;
int q[SIZE],front=0,rear=0;
void enqueue()
{
int no;
if (rear==SIZE && front==0)
cout<<"queue is full";
else
{
cout<<"enter the num:";
cin>>no;
q[rear]=no;
}
rear++;
}
void dequeue()
{
int no,i;
if (front==rear)
cout<<"queue is empty";
else
{
no=q[front];
front++;
cout<<"\n"<<no<<" -removed from the queue\n";
```

```
}
}
void display()
{
int i,temp=front;
if (front==rear)
cout<<"the queue is empty";
else
{
cout<<"\n element in the queue:";
for(i=temp;i<rear;i++)
{
cout<<q[i]<<" ";
}
}
}
int main()
{
int ch;
//clrscr();
//void enqueue();
//void dequeue();
//void display();
while(1)
{
cout<<"\n 1. add element";
cout<<"\n 2. remove element";
cout<<"\n 3.display";
cout<<"\n 4.exit";
cout<<"\n enter your choice:";
cin>>ch;
//clrscr();
switch(ch)
{
case 1:
enqueue();
break;
case 2:
```

```
dequeue();
break;
case 3:
display();
break;
case 4:
exit(0);
default:
cout<<"\n invalid choice";
}
}
}
```

CONCLUSION:

In this lab we understand the basic concept of queue we also implement the queue with their basic function as above mention.