Shri Vile Parle Kelavani Mandal's
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

# Department of Information Technology

**COURSE CODE:** DJS22ITL302 **DATE:5/10/2023**

**COURSE NAME:** Data Structure Laboratory **CLASS: I1-Batch1**

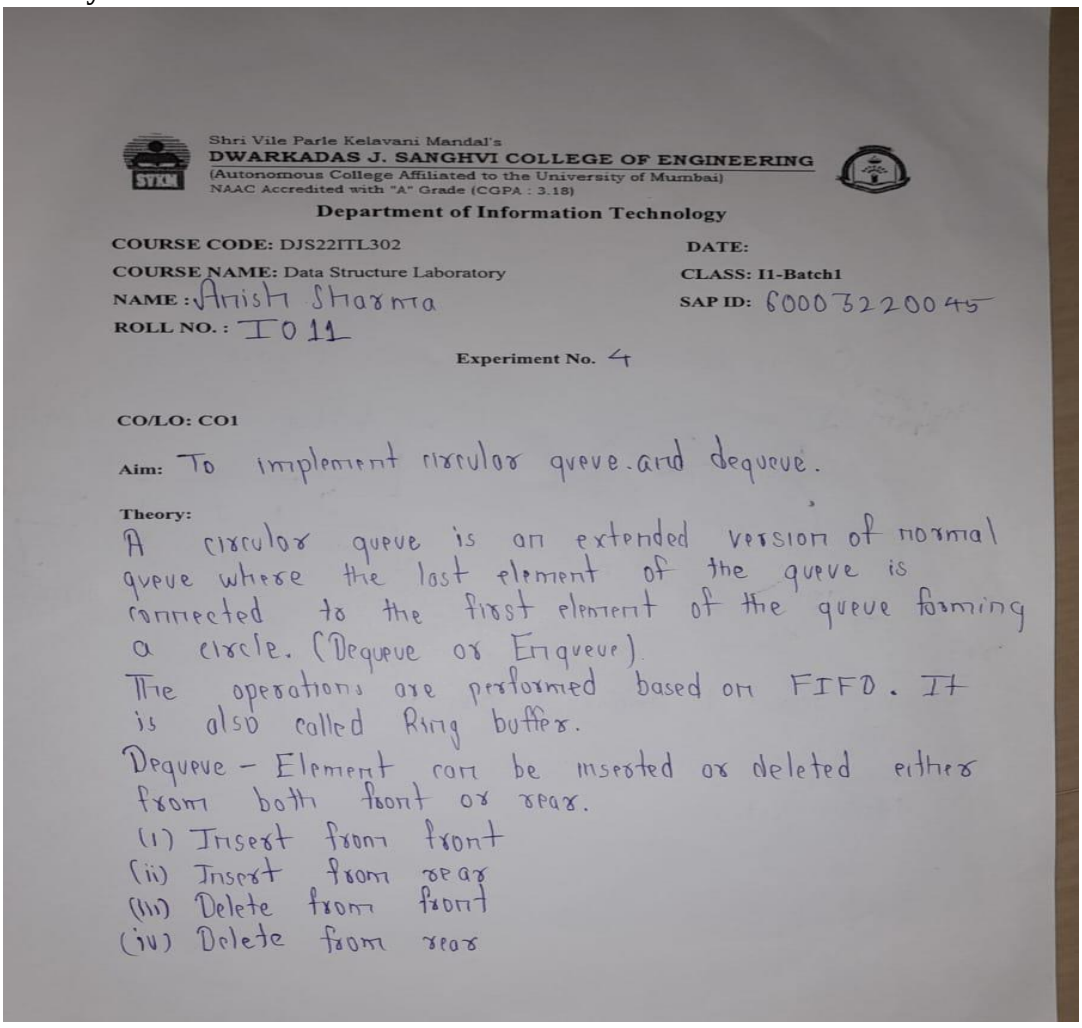
**NAME:** Anish Sharma

## Experiment No. 4

**CO/LO: CO1**

**Aim: Implementing deque and circular queue.**

**Theory:**



Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Information Technology

COURSE CODE: DJS22ITL302 DATE:
COURSE NAME: Data Structure Laboratory CLASS: I1-Batch1
NAME : Anish Sharma SAP ID: 60003220045
ROLL NO. : I011

Experiment No. 4

CO/LO: CO1

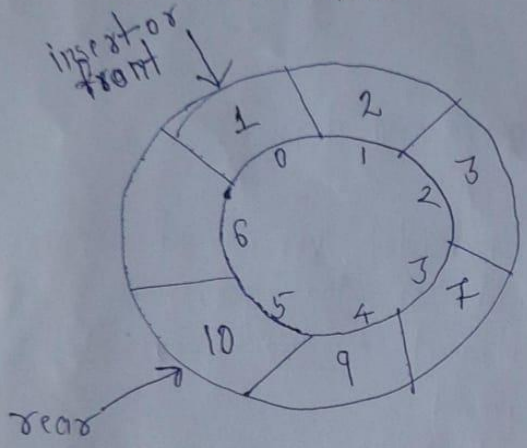
Aim: To implement circular queue and dequeue.

Theory:
A circular queue is an extended version of normal queue where the last element of the queue is connected to the first element of the queue forming a circle. (Dequeue or Enqueue)
The operations are performed based on FIFO. It is also called Ring buffer.
Dequeue - Element can be inserted or deleted either from both front or rear.
(i) Insert from front
(ii) Insert from rear
(iii) Delete from front
(iv) Delete from rear

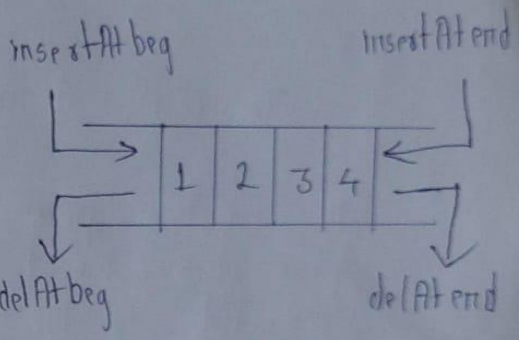
Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)
Department of Information Technology

**Output :**

Circular Queue




Dequeue




**Conclusion :**

We learnt to implement circular, dequeue using arrays.

**References :**

Mam's Notes

# Department of Information Technology

**Circular Queue:**

**A circular queue is similar to a linear queue as it is also based on the FIFO (First In First Out) principle except that the last position is connected to the first position in a circular queue that forms a circle. It is also known as a Ring Buffer.**

**Front: It is used to get the front element from the Queue.**

**Rear: It is used to get the rear element from the Queue.**

**enQueue(value): This function is used to insert the new value in the Queue. The new element is always inserted from the rear end.**

**deQueue(): This function deletes an element from the Queue. The deletion in a Queue always takes place from the front end.**

**Deque:**

**The deque stands for Double Ended Queue. Deque is a linear data structure where the insertion and deletion operations are performed from both ends.**

**Output restricted Queue**

**In output restricted queue, deletion operation can be performed at only one end, while insertion can be performed from both ends.**

**Input restricted Queue**

**In input restricted queue, insertion operation can be performed at only one end, while deletion can be performed from both ends.**

**Program:**

**Circular Queue:**

```
#include <stdio.h>

#include <stdlib.h>


#define size 3

int queue[size];

int front=-1;

int rear=-1;




void insert(int x){

   if(front==-1 && rear==-1){

      front=rear=0;

      queue[rear]=x;

   }

   else if(front==rear+1 || (front==0 && rear==size-1)){

      printf("Overflow\n");

   }

   else{

      rear=(rear+1)%size;

      queue[rear]=x;

   }
```

```
}


void del(){
   if(front==-1 && rear==-1){
      printf("Underflow\n");
   }
   else if(front==rear){
      printf("\nDelete element:%d",queue[front]);
      front=rear=-1;
   }
   else{
      printf("\nDelete element:%d",queue[front]);
      front=(front+1)%size;
   }
}
void display(){
   if(front==-1 && rear==-1){
      printf("There is no element\n");
   }
   else{
      printf("\nDisplaying element:\n");
```

```
    int i=front;
    while(i!=rear){
        printf("%d ",queue[i]);
        i=(i+1)%size;
    }
    printf("%d ",queue[i]);
  }


}
int main()
{
  int choice,x;
  printf("Implementation of Circular queue\n");
  while(1){
    printf("\nInsertion- 1\nDeletion  -2\nDisplay -3\nExit -0\n");
    scanf("%d",&choice);
    if(choice==0)
      break;
    else if(choice==1){
      printf("Enter a element:\n");
```

```
        scanf("%d",&x);
        insert(x);
      }
      else if(choice==2){
        del();
      }
      else if(choice==3){
        display();
      }

  }
  return 0;
}
```

**Deque:**

```
#include <stdio.h>
#define size 5
int deque[size];
int f = -1, r = -1;
void insert_front(int x)
{
```

```
    if((f==0 && r==size-1) || (f==r+1))
    {
       printf("Overflow");
    }
    else if((f==-1) && (r==-1))
    {
       f=r=0;
       deque[f]=x;
    }
    else if(f==0)
    {
       f=size-1;
       deque[f]=x;
    }
    else
    {
       f=f-1;
       deque[f]=x;
    }
}
```

```
void insert_rear(int x)
{
   if((f==0 && r==size-1) || (f==r+1))
   {
      printf("Overflow");
   }
   else if((f==-1) && (r==-1))
   {
      r=0;
      deque[r]=x;
   }
   else if(r==size-1)
   {
      r=0;
      deque[r]=x;
   }
   else
   {
      r++;
      deque[r]=x;
```

```
    }


}




void display()

{

    int i=f;

    printf("\nElements in a deque are: ");


    while(i!=r)

    {

        printf("%d ",deque[i]);

        i=(i+1)%size;

    }

     printf("%d",deque[r]);

}




void getfront()

{
```

```
    if((f==-1) && (r==-1))
    {
       printf("Deque is empty");
    }
    else
    {
       printf("\nThe value of the element at front is: %d", deque[f]);
    }

}


void getrear()
{
    if((f==-1) && (r==-1))
    {
       printf("Deque is empty");
    }
    else
    {
       printf("\nThe value of the element at rear is %d", deque[r]);
```

```
    }


}


void delete_front()
{
    if((f==-1) && (r==-1))
    {
        printf("Deque is empty");
    }
    else if(f==r)
    {
        printf("\nThe deleted element is %d", deque[f]);
        f=-1;
        r=-1;


    }
     else if(f==(size-1))
     {
         printf("\nThe deleted element is %d", deque[f]);
         f=0;
```

```
    }
    else
    {
        printf("\nThe deleted element is %d", deque[f]);
        f=f+1;
    }
}

void delete_rear()
{
   if((f==-1) && (r==-1))
   {
      printf("Deque is empty");
   }
   else if(f==r)
   {
      printf("\nThe deleted element is %d", deque[r]);
      f=-1;
      r=-1;
```

```
    }
     else if(r==0)
     {
        printf("\nThe deleted element is %d", deque[r]);
        r=size-1;
     }
     else
     {
         printf("\nThe deleted element is %d", deque[r]);
         r=r-1;
     }
}

int main()
{
    insert_front(20);
    insert_front(10);
    insert_rear(30);
    insert_rear(50);
    insert_rear(80);
    display();
```

```
    getfront();

    getrear();

    delete_front();

    delete_rear();

    display();

    return 0;

}
```

**Output screenshots:**

**Circular deque:**

## Department of Information Technology

```
Implementation of Circular queue

Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
1

Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
2

Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
3

Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
4
Overflow

Insertion- 1
Deletion  -2
Display -3
Exit -0
2

Delete element:1
Insertion- 1
Deletion  -2
Display -3
Exit -0
3

Displaying element:
2 3
Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
4
```

```
Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
6
Overflow

Insertion- 1
Deletion  -2
Display -3
Exit -0
1
Enter a element:
8
Overflow

Insertion- 1
Deletion  -2
Display -3
Exit -0
3

Displaying element:
2 3 4
Insertion- 1
Deletion  -2
Display -3
Exit -0
2

Delete element:2
Insertion- 1
Deletion  -2
Display -3
Exit -0
3

Displaying element:
3 4
Insertion- 1
Deletion  -2
Display -3
Exit -0

Displaying element:
3 4
Insertion- 1
Deletion  -2
Display -3
Exit -0
0
```

## Department of Information Technology

**Deque:**

```
Elements in a deque are: 10 20 30 50 80
The value of the element at front is: 10
The value of the element at rear is 80
The deleted element is 10
The deleted element is 80
Elements in a deque are: 20 30 50
```

**Conclusion:**

I have understood that the concept of deque and circular queue.

**REFERENCES:**

tutorialpoint