Harsh Kashiwal
2003085

DATE 02 10 21

Experiment 5

Aim: Implement Circular Queue ADT using array.

Theory:

Limitation of Linear Queue

In linear queue, once an element is deleted, we cannot insert another element in its position. This disadvantage of a linear queue is overcome by a circular queue, thus saving memory.
The circular queue connects the last node of a queue to its first by forming a circular link & because of this it resolves the memory wastage problem.

```
1. enqueue(val) {
    if( front==0 and rear=n-1) || (front==rear+1)
        print Queue is full.
    else
        if(front==-1)
            then front=0
        if (front!=0 and rear=n-1
            rear=-1
        rear++
        queue[rear]=val
    }
```

HK
085

2. dequeue(){
if (front==-1 && rear==-1)
print Empty.
else
if(front==rear)
queue[rear]
front=rear=-1
else if(front=n-1)
queue[front]
front 0
else
queue[front]
front++
}

3. display(){
if(front==-1 and rear==-1) ⇒ Empty.
else if(front<=rear)
for(i=front, i<=rear)
queue[i]
else{
for(i=front- i<=n-1)
queue[i]
for(i=0; i<=rear)
queue[i]}
}

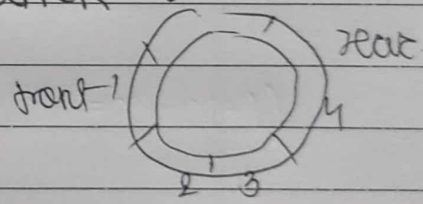
Harsh Hasijwal
2003085

4. get_front(){
        queue[front]
    }

5. ~~get_rear(){}~~
5. isEmpty(){
        if(front==-1 and rear==-1)
            return True

6. isfull(){
        if(front==0 and rear==n-1) || (front==rear+1)
            return True
    }.

Conclusion: Circular queue resolves the memory wastage problem by forming a circular link.



PROGRAM:

Write a menu driven code to implement CIRCULAR QUEUEADT using arrays.

Code:

```
#include <iostream>
using namespace std;
#define n 3
int queue[n];
int size = 0;
int rear = - 1;
int front = -1;

void enqueue(){
   int val;
    if(((front==0)&&(rear==n-1)) || (front==rear+1))
    cout<<"Queue iS FULL";
   else{
   if(front == - 1 )
      front = 0;
   if(front!=0 && (rear==n-1))
       rear=-1;
      cout<<"Insert the element in queue : "<<endl;
      cin>>val;
      rear++;
      queue[rear] = val;
      size++;
   }
}

void dequeue(){
   if (front == - 1 && rear == -1) {
      cout<<"Queue Underflow ";
   }
   else {
   if(front==rear){
       cout<<"1.Element deleted from queue is : "<< queue[front] <<endl;
       front=rear=-1;
       size--;
   }
   else if(front==n-1){
    cout<<"2.Element deleted from queue is : "<< queue[front] <<endl;
    front=0;
```

```
    size--;
    }
    else{
       cout<<"3.Element deleted from queue is : "<< queue[front] <<endl;
       front++;
       size--;
   }
}
}
void display(){
   if (front == - 1 && rear == -1)
   cout<<"Queue is Empty"<<endl;
   else if (front <=rear){
       for(int i=front; i<=rear; i++)
        cout<<queue[i]<<" ";
   }
   else{
       for(int i=front; i<=n-1; i++)
       cout<<queue[i]<<" ";
       for(int i=0; i<=rear; i++)
       cout<<queue[i]<<" ";
   }
   cout<<endl;
}
void get_front(){
    if (front == - 1 && rear == -1)
    cout<<"Queue is Empty"<<endl;
    else{
        cout<<"Your front element is: "<<queue[front]<<endl;
    }
}
void get_rear(){
    if (front == - 1 && rear == -1)
    cout<<"Queue is Empty"<<endl;
    else{
        cout<<"Your rear element is: "<<queue[rear]<<endl;
    }
}
int main() {
   int ch;
   cout<<"1) ENQUEUE "<<endl;
   cout<<"2) DEQUEUE"<<endl;
   cout<<"3) GET FRONT"<<endl;
   cout<<"4) GET REAR"<<endl;
   cout<<"5) SIZE"<<endl;
   cout<<"6) DISPLAY"<<endl;
   cout<<"7) EXIT"<<endl;
   do {
```

```
      cout<<"Enter your choice : "<<endl;
      cin>>ch;
      switch (ch) {
         case 1: enqueue();
         break;
         case 2: dequeue();
         break;
         case 3: get_front();
         break;
         case 4: get_rear();
         break;
         case 5:  cout<<"No. of elements in queue = "<<size;
                  cout<<endl;
         break;
         case 6:  display();
         break;
         case 7: cout<<"Exit"<<endl;
         break;
         default: cout<<"Invalid choice"<<endl;
      }
   } while(ch!=7);
   return 0;
}
```

OUTPUT:

```
PS D:\Harsh\SEM 3\DS\CODES> cd "d:\Harsh\SEM 3\DS\CODES\" ; if (
1) ENQUEUE
2) DEQUEUE
3) GET FRONT
4) GET REAR
5) SIZE
6) DISPLAY
7) EXIT
Enter your choice :
1
Insert the element in queue :
12
Enter your choice :
1
Insert the element in queue :
13
Enter your choice :
1
Insert the element in queue :
14
Enter your choice :
1
Queue iS FULL
Enter your choice :
6
12 13 14
Enter your choice :
2
3.Element deleted from queue is : 12
Enter your choice :
6
13 14
Enter your choice :
1
Insert the element in queue :
15
Enter your choice :
6
13 14 15
Enter your choice :
5
No. of elements in queue = 3
Enter your choice :
7
Exit
PS D:\Harsh\SEM 3\DS\CODES>
```