19/10/2020

Date
Page

Lab 5 – Circular Queue Implementation.

pseudocode:

```
Queue[MAX]
front ← -1
rear ← -1

enque(x)
if rear = (rear +1) % MAX = front
    PRINT Queue is full
else
    rear = (rear+1) % MAX
    queue(rear) = x

deque()
if (front = 0) && (rear) = -1
    PRINT Queue is Emphy
else
    front = rear
    front = rear = -1
front = (front+1) % MAX
```

```
#include <stdio.h>
#define MAX 5

int front=-1, rear=-1;
int queue[MAX];
void Enque(int);
int Deque();
void display();
int main(int argc, char
{
   int option;
   int item;
   do{
      printf("\n--------\n");
      printf("\n Circular Queue \n");
      printf("\n 1.Insert to Queue ");
      printf("\n 2. delete from the Queue ");
      printf("\n 3. Display the content");
      printf("\n 4. Exit \n");
      printf("Enter the option :");
      scanf("%d",&option);
      switch(option)
{
 case 1: printf("Enter the element \n");
         scanf("%d",&item);
         Enque(item);
         break;
```

```
case 2 : item = Deque();
         if(item == -1)
           printf("Queue is empty \n");
         else
         printf("Removed element from the queue
                 %d", item);
          break;
case 3 : display();
         break;
case 4 : exit(0);
}
}
while (option != 4);
 return 0;
}
void Enque(int ele)
{
  if ((rear + 1) % MAX == front)
     printf("Queue is full \n");
  else
  {
    rear = (rear + 1) % MAX;
   queue[rear] = ele;
   if (front == -1)
      front = 0;
}
}
int Deque()
{
```

```
int item;
if ((front == -1) && (rear == -1))
    return -1;
else
{
    item = queue[front];
    front = (front + 1) % MAX;
    if(front > rear)
    {
        front = -1;
        rear = -1;
    }
return item;
    }
}
void display()
{
    int i;
    if ((front == 0) && (rear == -1))
        printf ("Queue is empty \n");
    else
    {
    printf("\n Queue contents : ");
    for (i = front ; i <= rear ; i++)
        printf("\t %d", queue[i]);
    }
}
```