S.No: 16	Exp. Name: ***Implementation of Circular Queue using Dynamic Array***	Date:2023-05-20

Aim:

Write a program to implement `circular queue` using **dynamic array**.

ID: 224G1A0591

2022-2026-CSE-B

Srinivasa Ramanujan Institute of Technology

```
Sample Input and Output:
    Enter the maximum size of the circular queue : 3
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 2
    Circular queue is underflow.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 3
    Circular queue is empty.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 1
    Enter element : 111
    Successfully inserted.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 1
    Enter element : 222
    Successfully inserted.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 1
    Enter element : 333
    Successfully inserted.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 1
    Enter element : 444
    Circular queue is overflow.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 3
    Elements in the circular queue : 111 222 333
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 2
    Deleted element = 111
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 1
    Enter element : 444
    Successfully inserted.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 3
    Elements in the circular queue : 222 333 444
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 2
    Deleted element = 222
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 2
    Deleted element = 333
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 2
    Deleted element = 444
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 3
    Circular queue is empty.
    1.Enqueue 2.Dequeue 3.Display 4.Exit
    Enter your option : 4
```

Source Code:

CQueueUsingDynamicArray.c

```
#include<stdio.h>
#include<stdlib.h>
int *cqueue;
int front,rear;
int maxSize;
void initCircularQueue() {
   cqueue=(int*)malloc(maxSize*sizeof(int));
   front=-1;
   rear=-1;
}
void dequeue(){
   if(front==-1){
      printf("Circular queue is underflow.\n");
   }
   else{
      printf("Deleted element = %d\n",*(cqueue+front));
      if(rear==front){
         rear=front=-1;
      }
      else if(front==maxSize-1){
         front=0;
      }
      else{
         front++;
      }
   }
}
void enqueue(int x){
   if(((rear==maxSize-1)&&(front==0))||(rear+1==front)){
      printf("Circular queue is overflow.\n");
   }
   else{
      if(rear==maxSize-1){
         rear=-1;
      }
      else if(front==-1){
         front=0;
      }
      rear++;
      cqueue[rear]=x;
      printf("Successfully inserted.\n");
   }
}
void display(){
   int i;
   if(front==-1 && rear==-1){
      printf("Circular queue is empty.\n");
   }
   else{
      printf("Elements in the circular queue : ");
      if(front<=rear){
         for(i=front;i<=rear;i++){
            printf("%d ",*(cqueue+i));
         }
```

```
      }
      else{
         for(i=front;i<=maxSize-1;i++){
            printf("%d ",*(cqueue+i));
         }
         for(i=0;i<=rear;i++){
            printf("%d ",*(cqueue+i));
         }
      }
      printf("\n");
   }
}
int main(){
   int op,x;
   printf("Enter the maximum size of the circular queue : ");
   scanf("%d",&maxSize);
   initCircularQueue();
   while(1){
      printf("1.Enqueue 2.Dequeue 3.Display 4.Exit\n");
      printf("Enter your option : ");
      scanf("%d",&op);
      switch(op){
         case 1:  printf("Enter element : ");
               scanf("%d",&x);
               enqueue(x);
               break;
         case 2: dequeue();
               break;
         case 3: display();
               break;
         case 4: exit(0);
      }
   }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the maximum size of the circular queue : 3
1.Enqueue 2.Dequeue 3.Display 4.Exit 2
Enter your option : 2
Circular queue is underflow. 3
1.Enqueue 2.Dequeue 3.Display 4.Exit 3
Enter your option : 3
Circular queue is empty. 1
1.Enqueue 2.Dequeue 3.Display 4.Exit 1
Enter your option : 1
Enter element : 111
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.Exit 1
Enter your option : 1
Enter element : 222

ID: 224G1A0591

```
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.Exit 1
Enter your option :  1
Enter element :  333
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.Exit 1
Enter your option :  1
Enter element :  444
Circular queue is overflow. 3
1.Enqueue 2.Dequeue 3.Display 4.Exit 3
Enter your option :  3
Elements in the circular queue : 111 222 333  2
1.Enqueue 2.Dequeue 3.Display 4.Exit 2
Enter your option :  2
Deleted element = 111 1
1.Enqueue 2.Dequeue 3.Display 4.Exit 1
Enter your option :  1
Enter element :  444
Successfully inserted. 3
1.Enqueue 2.Dequeue 3.Display 4.Exit 3
Enter your option :  3
Elements in the circular queue : 222 333 444  2
1.Enqueue 2.Dequeue 3.Display 4.Exit 2
Enter your option :  2
Deleted element = 222 2
1.Enqueue 2.Dequeue 3.Display 4.Exit 2
Enter your option :  2
Deleted element = 333 2
1.Enqueue 2.Dequeue 3.Display 4.Exit 2
Enter your option :  2
Deleted element = 444 3
1.Enqueue 2.Dequeue 3.Display 4.Exit 3
Enter your option :  3
Circular queue is empty. 4
1.Enqueue 2.Dequeue 3.Display 4.Exit 4
Enter your option :  4
```