Experiment 9
Circular Queue Implementation Using Array

Date : 05-10-2020

Aim: To implement a circular queue using array

Data Structure used : Queue, Array

Algorithms

1. Algorithm for enqueue

Input: An Array implementation of Circular Queue (C_Q[SIZE]), with front pointing to the first element and rear pointing to the last element in and an element E to be inserted into the queue.
Output: The Circular Queue with the element E inserted at the front
Data Structure: Circular Queue

Steps:

```
Step 1: if((rear+1)%SIZE == front) then
        Step 1: print("The queue is full insertion not possible")
        Step 2: exit(1)
Step 2: else
        Step 1: if(rear == -1) then
                Step 1: front ++
        Step 2: EndIf
        Step 3: rear = (rear+1)%SIZE
        Step 4: C_Q[rear] = E
Step 3: EndIf
```

2. Algorithm for dequeue

Input: An Array implementation of Circular Queue (C_Q[SIZE]), with front pointing to the first element and rear pointing to the last element in the queue.
Output: The element E which is removed form the circular queue

Steps:

```
Step 1: if(front == -1) then
        Step 1: print("The Queue is empty")
        Step 2: exit(1)
Step 2: else
        Step 1: E = Q[front]
        Step 2: if(front == rear) then
                Step 1: front =-1
                Step 2: rear =-1
        Step 3: else
                Step 1: front = (front+1)%SIZE
        Step 4: endif
Step 3: endif
```

Program code:

```
#include<stdio.h>
#include<stdlib.h>

//Create a struct for our queue
typedef struct CQueue{
    int* Q;
    int front;
    int rear;
    int size;
} CQueue;


CQueue* initializeQueue(){
    int size = 2;

    //Create a pointer to stack
    CQueue *a = (CQueue*) malloc (sizeof(CQueue));
    if(a == NULL){
        printf("An Overflow error has ocurred while creating the CircularQueue\n");
        exit(1);
    }

    //create the array that will contain our stack
    a->Q = (int*)malloc(size*sizeof(int));
    if(a->Q == NULL){
        printf("An Overflow error has ocurred while creating the Circular Queue
array\n");
        exit(1);
    }

    a->front = -1;
    a->rear = -1;
    a->size = size;
    return a;
}

void deleteQueue(CQueue *a){
    free(a->Q);
    free(a);
}

void enQueue(CQueue *a,int item){
    if((a->rear+1)%(a->size) == a->front){

        a->size = a->size*2;
        a->Q = realloc(a->Q,a->size);
        //printf("CircularQueue is Full \n");
        //fflush(stdout);
        if(a->Q == NULL){
            printf("An Overflow Error has occured while reallocating the array\
nEXITING!!!!!!!\n");
            exit(1);
```

```
        }
    }

    if(a->front == -1){
        a->front = 0;
    }

    a->rear = (a->rear +1)%a->size;
    a->Q[a->rear] = item;
}

int deQueue(CQueue *a){
   if(a->front == -1){
        printf("You have made a grave mistake, the CQueue was empty\n\n");
        deleteQueue(a);
        exit(1);
        return -1;

   }
   else{
        int item = a->Q[a->front];
        if(a->front == a->rear){
            a->front = -1;
            a->rear = -1;
        }
        else{
            a->front = (a->front+1)%(a->size);
        }
        return item;
   }
}

void displayQueue(CQueue *a){
    int i = a->front;

    while(i!=(a->rear+1)%(a->size)){
        printf("%d ",a->Q[i]);
        i = (i+1)%a->size;
    }
    printf("\n");
}


int menu(CQueue *a){
        int RUN=1;
        int c;      //For the corresponding choice
        int item;   //To receive the item to push or pop from the array
        while (RUN){
            printf("\n");
            printf("-----------------------------------------------\n");
            printf("Circular Queue Implementation using structure\n");
            printf("-----------------------------------------------\n");
            printf("1.Insert\n");
            printf("2.Delete\n");
            printf("3.Print the queue\n");
            printf("4.Exit\n");
```

```
            printf("Enter the required choice --> ");
            scanf("%d%*c",&c);

            switch(c){
                case 1:printf("Enter the element to be inserted into the queue -->
");
                       scanf("%d%*c",&item);
                       enQueue(a,item);
                       break;

                case 2:item = deQueue(a);
                       printf("Item removed is  is --> %d\n",item);
                       break;

                case 3:printf("The Circular Queue is --> ");
                       displayQueue(a);
                       break;

                case 4: RUN=0;
                        break;

                default:printf("Entered command is unknown");
            }

        }
        deleteQueue(a);

        printf("Finished excecuting the code ALL DONE\n");
        return RUN;
}

int main(){
    CQueue *a;
    a = initializeQueue();
    return menu(a);
}
```

Result: The Program was compiled successfully and the desired output was obtained.

Sample input/Output:

```
┌─rohit@iris ~/Programing/C/CSL201/2020-11-05
└─➤  gcc -Wall circ_queue.c -o circ_queue.o
┌─rohit@iris ~/Programing/C/CSL201/2020-11-05
└─➤  ./circ_queue.o

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 1
Enter the element to be inserted into the queue --> 12

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 1
Enter the element to be inserted into the queue --> 54

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 1
Enter the element to be inserted into the queue --> 73

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 2
Item removed is  is --> 12
```

```
------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 2
Item removed is  is --> 54

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 3
The Queue is --> 73

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 2
Item removed is  is --> 73

------------------------------------------------
Circular Queue Implementation using structure
------------------------------------------------
1.Insert
2.Delete
3.Print the queue
4.Exit
Enter the required choice --> 2
You have made a grave mistake, the Queue was empty

┌─rohit@iris ~/Programing/C/CSL201/2020-11-05
└─➤
```