04/10/2020

**Experiment No:11**

**DEQUE USING ARRAYS**

**AIM:**

Write a program to implement Deque using arrays.

**DATA STRUCTURES USED:**

Queue

**ALGORITHM:**

Algorithm INSERT\_FRONT (ITEM)

1. If (FRONT==0) // N is the size of Queue
2. print “Insertion not possible”
3. Exit
4. Else
5. If (FRONT =-1)
6. FRONT=REAR=0
7. Queue[REAR]=ITEM
8. Else
9. Queue[--FRONT]=ITEM
10. EndIf
11. EndIf

Algorithm INSERT\_REAR (ITEM)

1. If (REAR=N-1)
2. print “Queue is full”
3. Exit
4. Else
5. If (REAR =-1 && FRONT =-1)
6. FRONT=REAR=0
7. Queue[REAR]=ITEM
8. Else
9. Queue[++REAR]=ITEM
10. EndIf
11. EndIf

Algorithm DELETE\_FRONT

1. If (FRONT=-1)
2. Print “Queue is empty”
3. Exit
4. Else
5. If (FRONT == REAR)
6. ITEM = Queue[FRONT]
7. FRONT=REAR=-1
8. Else
9. ITEM = Queue[FRONT]
10. FRONT++
11. EndIf
12. EndIf

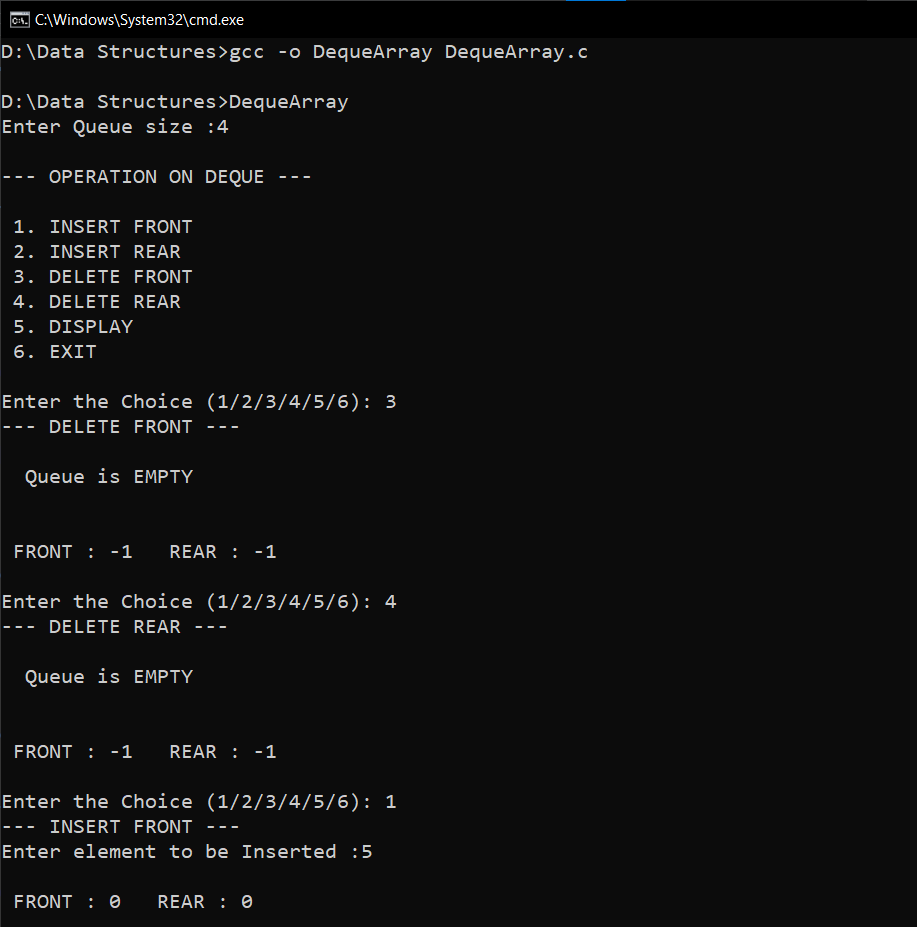
Algorithm DELETE\_REAR

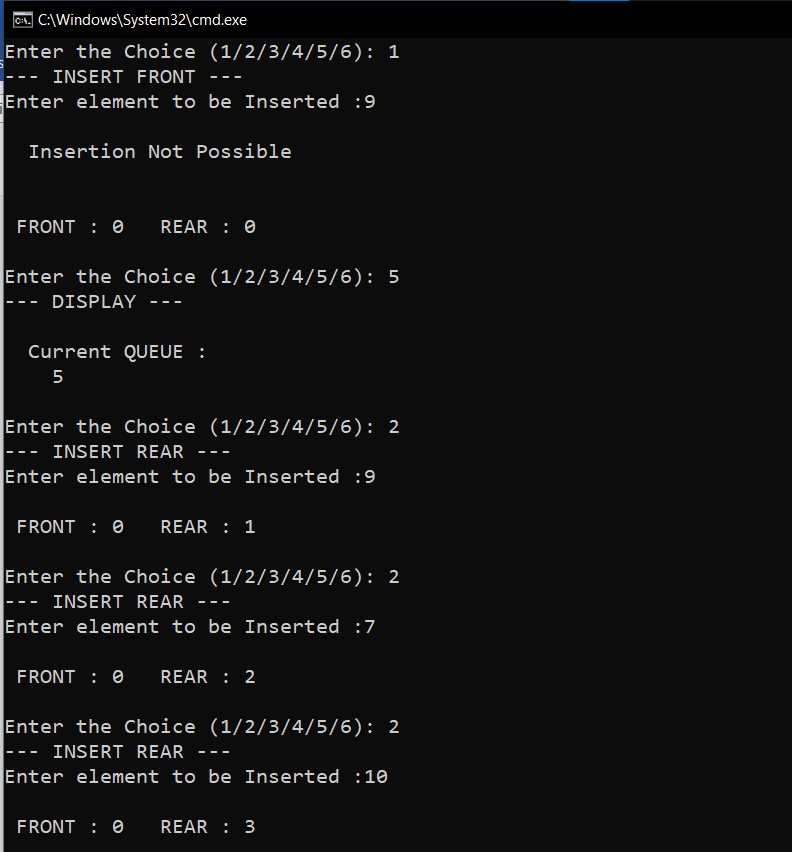
1. If (REAR=N-1)
2. Print “Deletion not possible”
3. Exit
4. Else
5. If (FRONT == REAR)
6. ITEM = Queue[FRONT]
7. FRONT=REAR=-1
8. Else
9. ITEM = Queue[FRONT]
10. REAR--
11. EndIf
12. EndIf

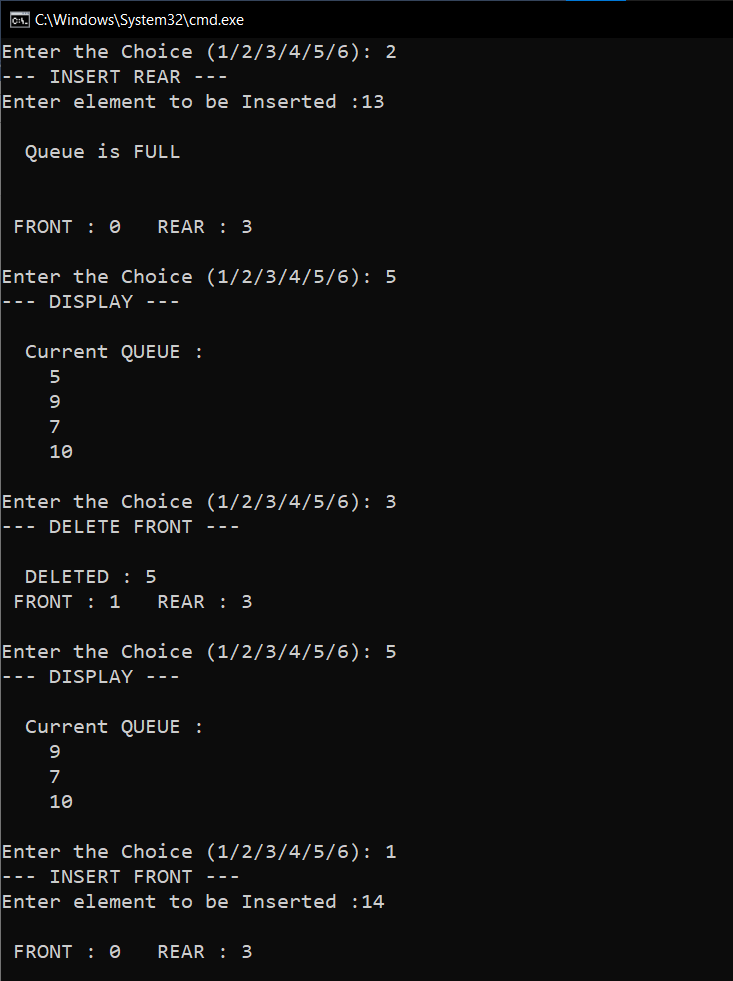
**PROGRAM:**

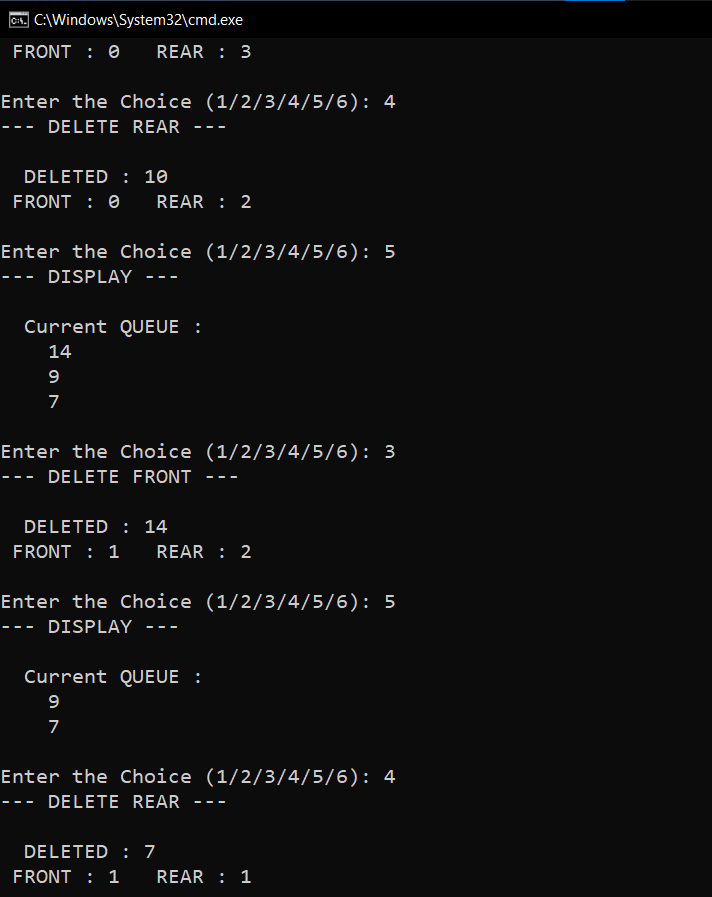
#include <stdio.h>  
#include <stdlib.h>  
int size;  
int front;  
int rear;  
int \*arr;  
  
void insertFront(int item){  
 if(front==0){  
 printf("\n Insertion Not Possible \n\n");  
 }  
 else if(front==-1){  
 arr[++front] = item;  
 rear++;  
 }  
 else {  
 arr[--front] = item;  
 }  
 printf("\n FRONT : %d REAR : %d \n",front,rear);  
}  
void insertRear(int item){  
 if(rear==size-1){  
 printf("\n Queue is FULL\n\n");  
 }  
 else if(front==-1){  
 arr[++front] = item;  
 ++rear;  
 }  
 else {  
 arr[++rear] = item;  
 }  
 printf("\n FRONT : %d REAR : %d \n",front,rear);  
}  
void deleteFront(){  
 if(front==-1){  
 printf("\n Queue is EMPTY\n\n");  
 }  
 else if(front==rear){  
 int item = arr[front];  
 printf("\n DELETED : %d ",item);  
 front=-1;  
 rear=-1;  
  
 }else{  
 int item = arr[front];  
 front++;  
 printf("\n DELETED : %d ",item);  
 }  
 printf("\n FRONT : %d REAR : %d \n",front,rear);  
}  
void deleteRear(){  
 if(rear==-1){  
 printf("\n Queue is EMPTY\n\n");  
 }  
 else if(front==rear){  
 int item = arr[front];  
 printf("\n DELETED : %d ",item);  
 front=-1;  
 rear=-1;  
  
 }else{  
 int item = arr[rear];  
 --rear;  
 printf("\n DELETED : %d ",item);  
 }  
 printf("\n FRONT : %d REAR : %d \n",front,rear);  
}  
void display(){  
 printf("\n Current QUEUE :\n");  
 if(front==-1){  
 printf("\n Queue is EMPTY \n");  
 }else{  
 for(int i=front; i<=rear; i++){  
 printf(" %d \n",arr[i]);  
 }  
 }  
}  
  
void main(){  
 int n,x,y;  
 char ans='y';  
 printf("Enter Queue size :");  
 scanf("%d", &size);  
 arr = (int\*) malloc (size \* sizeof(int));  
 front=-1,rear=-1;  
 printf("\n--- OPERATION ON DEQUE --- \n\n");  
 printf(" 1. INSERT FRONT \n");  
 printf(" 2. INSERT REAR\n");  
 printf(" 3. DELETE FRONT\n");  
 printf(" 4. DELETE REAR\n");  
 printf(" 5. DISPLAY\n");  
 printf(" 6. EXIT\n");  
 while(ans=='y'){  
 printf("\nEnter the Choice (1/2/3/4/5/6): ");  
 scanf("%d",&n);  
 switch(n){  
 case 1:printf("--- INSERT FRONT ---\n");  
 printf("Enter element to be Inserted :");  
 scanf("%d", &x);  
 insertFront(x);  
 break;  
 case 2:printf("--- INSERT REAR ---\n");  
 printf("Enter element to be Inserted :");  
 scanf("%d", &x);  
 insertRear(x);  
 break;  
 case 3:printf("--- DELETE FRONT ---\n");  
 deleteFront();  
 break;  
 case 4:printf("--- DELETE REAR ---\n");  
 deleteRear();  
 break;  
 case 5:printf("--- DISPLAY ---\n");  
 display();  
 break;  
 case 6:ans='n';  
 break;  
 default:printf("Enter a Valid Input\n");  
 }  
 }  
}

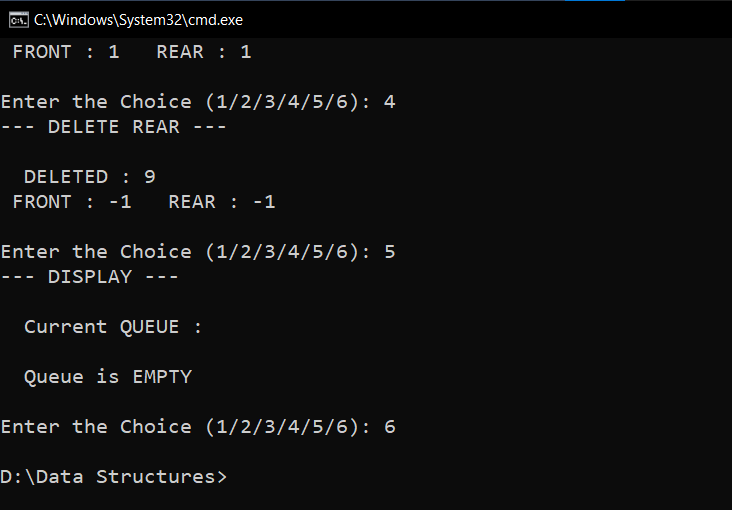
**OUTPUT:**











**RESULT:**

The Deque was successfully implemented and the required output was obtained.

Time complexity of INSERT\_REAR() operation is O(1).

Time complexity of DELETE\_FRONT() operation is O(1).

Time complexity of INSERT\_FRONT() operation is O(1).

Time complexity of DELETE\_REAR() operation is O(1).