**Assignmennt-7**

**Qus-1: Write a program implementing insert, delete and display operation of Circular Queue.**

**Ans:** **Operations on Circular Queue 🡪**

* **Front:** Get the front item from queue.
* **Rear:** Get the last item from queue.
* **enQueue(value):**This function is used to insert an element into the circular queue. In a circular queue, the new element is always inserted at Rear position.

***Steps to insret element in circular queue;***

**Step 1** 🡪IF (REAR +1)%SIZE = FORENT

Write “Overflow”

GOTO STEP 4

[END OF IF]

**Step 2** 🡪 IF FORENT = -1 AND REAR = -1

SET FORENT = REAR = 0

ELSE IF REAR = SIZE – 1 AND FORENT != 0

SET REAR = 0

ELSE

SET REAR = (REAR + 1)%SIZE

[END OF IF]

**Step 3** 🡪 SET QUEUE[REAR] = VALUE

**Step 4** 🡪 EXIT

***Steps to delete element in circular queue;***

Step 1 🡪 IF FORENT = -1

WRITE “UnderflOW”

GOTO STEP 4

[END OF IF]

Step 2 🡪 SET VALUE = QUEUE[FORENT]

Step 3 🡪 IF FORENT = REAR

SET FORENT = REAR = -1

ELSE IF FORENT = SIZE -1

SET FORENT = 0

ELSE

SET FORENT =FORENT + 1

[END OF IF]

Step 4 🡪 EXIT

***Implementation of circular queue:***

#include<stdio.h>

#define size 5

int queue[size];

int f= -1;

int r = -1;

//*function to insert element in circular queue.*

void enqueuer(int x)

{

if (f == -1 && r == -1) //*condition to check queue is empty.*

{

f = 0;

r = 0;

queue[r] = x;

}

else if(f == (r + 1)%size) //*condition to check queue is full.*

{

printf(“Overflow”);

}

else

{

r = (r + 1)%size; //*rear is incremented.*

queue[rear] = x; //*assign a value to the queue at the rear position.*

}

}

//*function to delete the element from the queue.*

int deQueue()

{

if(f == -1 && r == -1) //*condition to check to queue is empty.*

{

printf(“Underflow\n”);

}

else if(f ==r)

{

printf(“Deleted element is: %d\n”,queue[f]);

f = -1;

r = -1;

}

else

{

printf(“Deleted element is: %d\n”,queue[f]);

f = (f + 1)%size;

}

}

void display()

{

int i;

if(f == -1 && r == -1)

{

printf(“Queue is Empty\n”);

}

else

{

printf(“Elements in Queue are:\n”);

while(i<=r)

{

printf(“%d”, queue[i]);

i = (i + 1)%size;

}

}

}

int main()

{

int choice ,x;

while(choice < 4 && choice != 0)

{

printf(“Press 1: Insert an element\n”);

printf(“Press 2: Delete an element\n”);

printf(“Press 3: Display the element\n”);

printf(“Enter your choice”);

scanf(“%d”, &choice);

switch(choice)

{

case 1:

printf(“Enter the element which is to be inserted”);

scanf(“%d”, &x);

enQueue(x);

break;

case 2:

deQueue();

break;

case 3:

display();

}

}

return 0;

}