**Pimpri Chinchwad College of Engineering**

**Department of MCA**

**Data Structure Lab**

**Name : Kiran Chavan**

**Roll No : 21**

**Assignment No. 03 : Assignment based on Queue**

**Date of Submission : 5th March 2021**

1. Write a menu driven program using C language to perform following operations on Queue :

* Insert
* Delete
* View

Solution :

#include <stdio.h>

#include <stdlib.h>

#define SIZE 5

int enqueue();

int dequeue();

int display();

int front = -1,rear = -1,n,i,queue[SIZE],ch;

int main()

{

while(1){

printf("\n\n =====Queue Menu=====");

printf("\n 1.Insert");

printf("\n 2.Delete");

printf("\n 3.View");

printf("\n 0.Exit");

printf("\n Enter the Choice : ");

scanf("%d",&ch);

switch(ch){

case 1 : enqueue();

break;

case 2 : dequeue();

break;

case 3 : display();

break;

case 0 : exit(0);

break;

default : printf("\n Invalid Choice");

break;

}

}

return 0;

}

int enqueue(){

if(rear == SIZE)

printf("\n Queue is Full");

else{

printf("\n Enter the Element : ");

scanf("%d",&n);

if(front == -1 && rear == -1)

front = rear = 0;

queue[rear++] = n;

printf("\n Element Inserted!");

}

return 0;

}

int display(){

if(front == -1)

printf("\n Queue is Empty");

else{

printf("\n Queue is : ");

for(i=front;i<rear;i++)

printf("%d ",queue[i]);

}

return 0;

}

int dequeue(){

if(front == -1 || front >= rear)

printf("\n Queue is Empty");

else{

printf("\n %d Element is delete from Queue",queue[front++]);

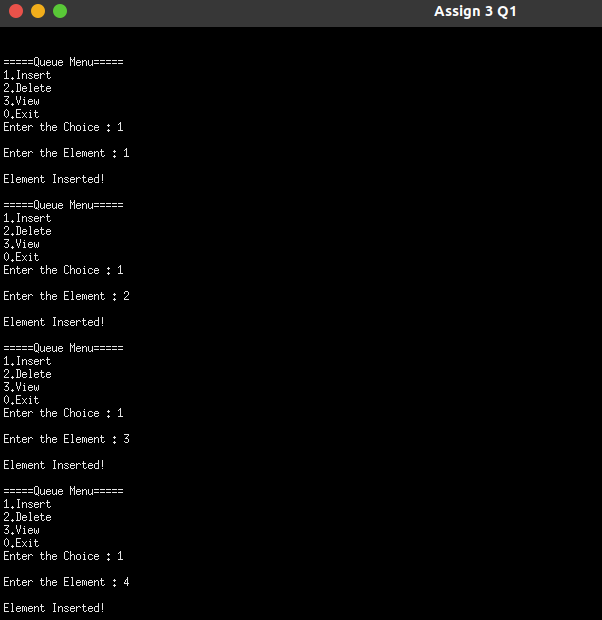
}

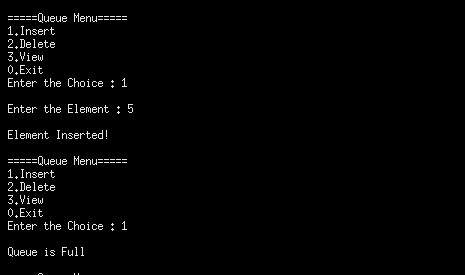
return 0;

}

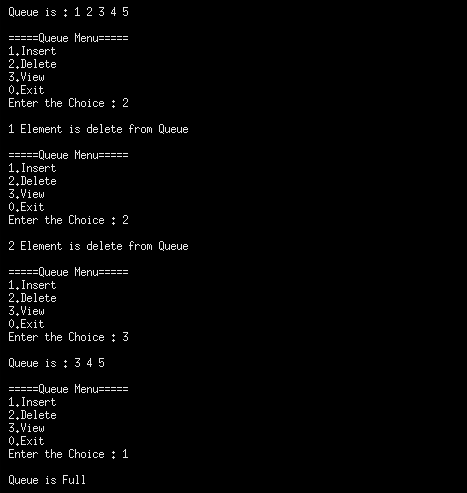
Output :

1.Insert



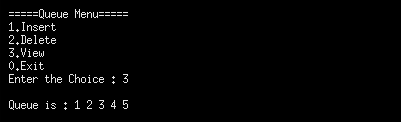


2.Delete :

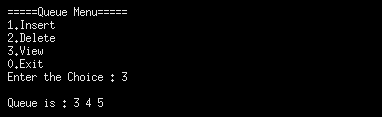


3 . Display :

After inserting all Elements :



After Deleting 2 Elements :



1. Write a menu driven program using C language to perform following operations on Circular Queue :

* Insert
* Delete
* View

Solution :

#include <stdio.h>

#include <stdlib.h>

#define SIZE 5

int enqueue();

int dequeue();

int display();

int front = -1,rear = -1,n,i,queue[SIZE],ch;

int main()

{

while(1){

printf("\n\n =====Circualr Queue=====");

printf("\n 1.Insert");

printf("\n 2.Delete");

printf("\n 3.View");

printf("\n 0.Exit");

printf("\n Enter the Choice : ");

scanf("%d",&ch);

switch(ch){

case 1 : enqueue();

break;

case 2 : dequeue();

break;

case 3 : display();

break;

case 0 : exit(0);

break;

default : printf("\n Invalid Choice");

break;

}

}

return 0;

}

int enqueue(){

if((rear+1)%SIZE == front)

printf("\n Circular Queue is Overflow");

else{

printf("\n Enter the Element : ");

scanf("%d",&n);

if(front == -1 && rear == -1){

front = rear = 0;

queue[rear] = n;

printf("\n Element Inserted!");

}

else{

rear = (rear+1)%SIZE;

queue[rear] = n;

printf("\n Element Inserted!");

}

}

return 0;

}

int display(){

if(front == -1 && rear == -1)

printf("\n Queue is Empty");

else{

printf("\n Circular Queue is : ");

for(i=front;i<rear;i = (i+1)%SIZE)

printf("%d ",queue[i]);

printf("%d ",queue[i]);

}

return 0;

}

int dequeue(){

if(front == -1 && rear == -1)

printf("\n Circular Queue is Empty");

else

if(front == rear){

printf("\n %d Element is delete from Circular Queue",queue[front]);

front = -1;

rear = -1;

}

else{

printf("\n %d Element is delete from Queue",queue[front]);

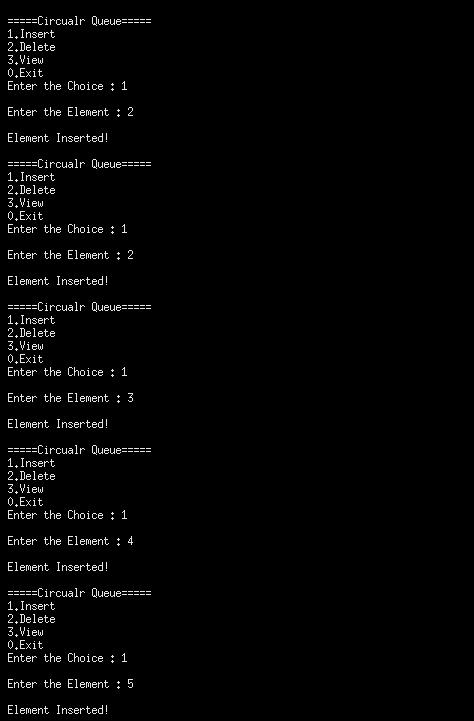
front = (front+1)%SIZE;

}

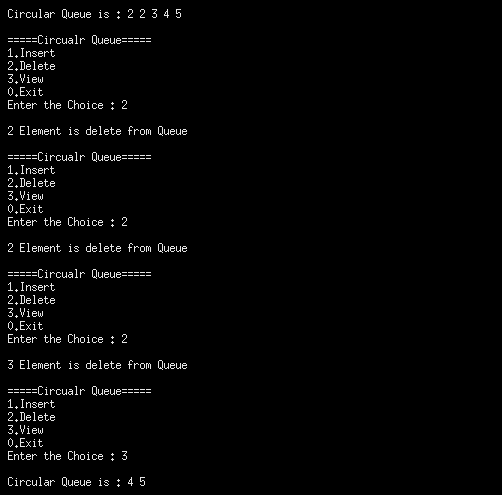
return 0;

}

Output : 1. Insert



2 . Delete



3.Display :

After Inserting all Elements :



After Deleting 3 Elements :

