**Assignment 10**

**AIM:** Pizza parlor accepting maximum 10 order .Orderare served on first come first serve basis .order once placed can’t be cancel .write c++ to simulate the system using circular Queue

**Objective:** Circular Queue is a linear data structure in which the operations are performed based on FIFO (First In First Out) principle and the last position is connected back to the first position to make a circle. It is also called **‘Ring Buffer’**

**Theory :** 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.
* **deQueue()** This function is used to delete an element from the circular queue. In a circular queue, the element is always deleted from front position.

Source code :

#include<iostream>

using namespace std;

class pizza

{

int front,rear,q[5];

public:

pizza()

{

front=-1;

rear=-1;

}

int isfull()

{

if((front==0&&rear==4)||front==rear+1)

{

return 1;

}

else

{

return 0;

}

}

int isempty()

{

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

{

return 1;

}

else

{

return 0;

}

}

void add()

{

if(isfull()==0)

{

cout<<"\n Enter the Pizza ID: ";

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

{

front=0;

rear=0;

cin>>q[rear];

}

else

{

rear=(rear+1)%5;

cin>>q[rear];

}

char ch;

cout<<" \nDo you want to add another order? \nIf yes ~press Y\nElse ~press N\n";

cin>>ch;

if(ch=='y'||ch=='Y')

add();

}

else

{

cout<<"\n Orders are full ";

check();

}

}

void serve()

{

if(isempty()==0)

{

if(front==rear)

{

cout<<"\n Order served is : "<<q[front];

front=-1;

rear=-1;

}

else

{

cout<<"\n Order served is : "<<q[front];

front=(front+1)%5;

}

}

else

{

cout<<"\n Orders are empty ";

}

}

void display()

{

if(isempty()==0)

{

for(int i=front;i!=rear;i=(i+1)%5)

{

cout<<q[i]<<" <- ";

}

cout<<q[rear];

}

else

{

cout<<"\n Orders are empty";

}

}

void check()

{

int choice;

cout<<"\n\n\*\*\*\*\*\*\*\*\*\*PIZZA HOUSE\*\*\*\*\*\*\*\*\*\*\n\n";

cout<<"\n 1. Add a Pizza \n 2. Display the Orders \n 3. Serve a pizza \n 4. Exit \n Enter your choice : ";

cin>>choice;

switch(choice)

{

case 1:

add();

break;

case 2:

display();

break;

case 3:

serve();

break;

case 4:

exit(0);

default:

cout<<" Invalid choice ";

check();

}

char ch1;

cout<<"\nIf you want to continue ~press Y\nelse ~press N\n ";

cin>>ch1;

if(ch1=='y'||ch1=='Y')

check();

}

};

int main()

{

pizza p1;

p1.check();

return 0;

}

Output :

/\*

\*\*\*\*\*\*\*\*\*\*PIZZA HOUSE\*\*\*\*\*\*\*\*\*\*

1. Add a Pizza

2. Display the Orders

3. Serve a pizza

4. Exit

Enter your choice : 1

Enter the Pizza ID: 12

Do you want to add another order?

If yes ~press Y

Else ~press N

y

Enter the Pizza ID: 17

Do you want to add another order?

If yes ~press Y

Else ~press N

y

Enter the Pizza ID: 44

Do you want to add another order?

If yes ~press Y

Else ~press N

n

If you want to continue ~press Y

else ~press N

y

\*\*\*\*\*\*\*\*\*\*PIZZA HOUSE\*\*\*\*\*\*\*\*\*\*

1. Add a Pizza

2. Display the Orders

3. Serve a pizza

4. Exit

Enter your choice : 2

12 <- 17 <- 44

If you want to continue ~press Y

else ~press N

y

\*\*\*\*\*\*\*\*\*\*PIZZA HOUSE\*\*\*\*\*\*\*\*\*\*

1. Add a Pizza

2. Display the Orders

3. Serve a pizza

4. Exit

Enter your choice : 3

Order served is : 12

If you want to continue ~press Y

else ~press N

n

--------------------------------

Process exited after 73.3 seconds with return value 0

Press any key to continue . . .

\*/

Conclusion :

Developed a program for Pizza parlor accepting maximum 10 order .Orderare served on first come first serve basis .order once placed can’t be cancel .write c++ to simulate the system using circular Queue