

~~CATP~~

Output

- 1. Insert Rear
- 2. Delete Front
- 3. Display
- 4. Exit

Enter the choice:

1

Enter item to be inserted : 11

① 1. insertRear

2. DeleteFront

3. DISPLAY

4. exit

Enter the choice : 11

2

~~Entered~~ Item deleted = 11

1. insertRear

2. DeleteFront

3. DISPLAY

4. exit

Enter the choice :

Enter item to be inserted : 13

1. insertRear

2. DeleteFront

3. DISPLAY

4. exit

Enter the choice :

Enter item to be inserted : 14

1. Insert Rear
2. Delete Front
3. Display
4. exit

{ DISPLAY }

Enter choice : 3

Content & Of Queue is given below

13, 44

(value - mid value) below

X

$$((1 - 2512) \rightarrow 5096 \quad \& \quad 0 \rightarrow 11087) \text{ di } \\ ((1 - 2512) \& (1 - 11087) \rightarrow 5095)$$

$$((1 - 11087) \& (1 - 5095) \rightarrow 27169) \\ (11087 - 5095)$$

$$(0 = 1000 - 11087) \& 3819$$

$$\text{value} = (5095) 30100$$

$$(0 = 11087 \& 1 - 3819 = 5095) \text{ di } 3213$$

$$10 = 5095$$

$$\text{value} = (5095) 10200$$

$$3819$$

$$11087$$

Q
By MAP to simulate the working of a Circular Queue of integers using an array, provide the following operations : insert, delete & display. The program should put appropriate conditions for underflow and overflow conditions.

```
#include<stdio.h>
#define SIZE 5
```

```
int Queue[SIZE], front = -1, rear = -1;
```

```
void insert(int value)
```

```
{
```

```
if(front == 0 && rear == SIZE-1) ||  
(rear == (front-1)% (SIZE-1))
```

```
{
```

```
printf("Queue is overflow\n");
```

```
return;
```

```
}
```

```
else if (front == -1)
```

```
{
```

```
front = rear = 0;
```

```
Queue[rear] = value;
```

```
}
```

```
else if (rear == SIZE-1 && front != 0)
```

```
{
```

```
rear = 0;
```

```
Queue[rear] = value;
```

```
}
```

```
else
```

```
{
```

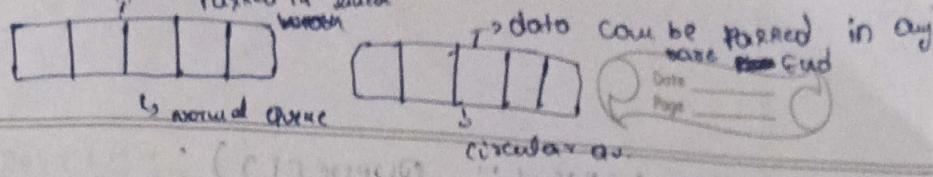
```
rear +=;
```

```
Queue[rear] = value;
```

```
}
```

```
}
```





void delete()

{

if (front == -1)

{

cout << "Queue is empty\n";

return (1);

}

cout << "Deleted element is ", Queue[front];

Queue[front] = -1;

cout << "Queue is now ", Queue;

if (front == rear) {

front = rear = -1;

}

else if (front == SIZE - 1)

{

front = 0; rear = SIZE - 1;

}

else { front++; rear++; }

cout << "Queue is now ", Queue;

cout << "Queue is now ", Queue;

cout << "Queue is now ", Queue;

void display()

{

if (front == -1) {

cout << "Queue is empty\n";

return;

cout << "Queue is now ", Queue;

if (rear > front) {

{

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

```
pointf("y.d", queue[i]);  
y  
y  
edge ?  
for (int i = front; i < size; i++)  
{  
    cout << queue[i] << endl;  
    pointf("y.d", queue[i]);  
}  
if (front == size) break;  
for (int i = 0; i < size; i++)  
{  
    pointf("y.d", queue[i]);  
}  
y  
y  
Pointf("In");  
y  
int main()  
{  
    int choice, value; char  
    while (1)  
    {  
        Pointf("In, Insert, Delete, Display, Exit In");  
        Pointf("Enter your choice: ");  
        scanf("y.d", &choice);  
    }  
}
```

```
switch (choice)  
{  
    case 1:  
    {  
        Pointf("Enter the value to  
        -insert: ");  
        scanf("y.d", &value);  
        insert(value);  
        break;  
    }  
    case 2:  
    {  
        Pointf("Delete the value  
        from index: ");  
        int index;  
        scanf("y.d", &index);  
        delete(index);  
    }  
    case 3:  
    {  
        Pointf("Display the  
        elements in the queue: ");  
        display();  
    }  
    case 4:  
    {  
        Pointf("Exit the program: ");  
        exit(0);  
    }  
}
```

delete();

break;

case 3;

display();

break;

case 4;

return 0;

default:

printf("Invalid choice, try again.
");

3

3

~~return 0;~~

3

off

if ((a < 0) || (a > 100)) {

 cout << "Error! Input must be between 0 and 100.";

 exit(1);

 cout << "Enter a value between 0 and 100: ";

 cin >> a;

 cout << "Value entered: " << a;

 cout << endl;

 cout << "Square root of " << a;

 cout << " is " << sqrt(a);

 cout << endl;

Q.

#include

pseudo code

#include <stdio.h>

int full(int value)

1

Pseudo Code

WAP

void Enqueue(int data)

{

if (data ==

Pseudo Code for Circular Queue

define SIZE -> 5

int Queue, Front = rear = -1;

void Enqueue (value)

{

if (front == 0 && rear == size - 1) ||

(rear == (front + n) - (SIZE - 1))

printf "Queue is Empty or full"

else if (front == -1)

front = rear = 0;

Queue[rear] = value;

else if (rear == SIZE - 1 && front != 0)

rear = 0

Queue[rear] = value;

else {

rear++;

Queue[rear] = value;



```
void deQueue()
{
    if (isEmpty) {
        cout << "Queue is empty" << endl;
        return;
    }
    cout << "Queue is not empty" << endl;
    if (front == rear)
    {
        front = rear = -1;
    }
    else
    {
        cout << "Value at front is " << arr[front] << endl;
        front++;
    }
}

void display()
{
    if ("is empty")
    {
        cout << "Queue is empty" << endl;
        return 0;
    }
    cout << "Queue elements are";
    if (!q.empty())
    {
        for (int i = front; i <= rear; i++)
        {
            cout << queue[i];
        }
    }
}
```

else

{
 for(int i = front; i < size; i++)

{
 print("int " + i + " = " + queue[i] + ";");

{
 Point queue[i];

{
 cout << "front = " << front << endl;

{
 cout << "back = " << back << endl;

int main()

{
 // define choice & value

switch(choice)

{

case 1: cout << "Enter value : ";

enqueue();

case 2: cout << "Enter value : ";

dequeue();

case 3: cout << "Enter value : ";

display();

default:

OR select 3 to display current list

4 to enter new value

5 to exit program

✓ 21/10/2023 - 11:15:07

OIP

1. Insert
2. Delete
3. DISPLAY
4. Exit

{ Insert }

Enter your choice : 1

Enter the value to insert: 23

Enter the value to insert: 24

Enter the value to insert: 24

Enter the value to insert: 29

Enter the value to insert: 38

1. ~~ENTER~~ Insert

2. Delete

3. DISPLAY

4. Exit

Enter your choice: 1

Queue is overflow

1. Insert

2. Delete

3. DISPLAY

4. Exit

Enter your choice: 3

Queue elements are:

23

24

24

29

38

1. Insert
2. Delete
3. DISPLAY
4. Exit

9. Delete
10. Exit

Enter your choice : 2

Deleted 23 31 20 30 31 32

33 34 35 36 37 38

Deleted 24

33 34 35 36 37 38

Deleted 24

33 34 35 36 37 38

Deleted 24

33 34 35 36 37 38

Deleted 38

33 34 35 36 37 38

1. Insert

2. Delete

3. DISPLAY

4. Exit

0 Enter your choice 2023-24

Queue is full underflow

1. Insert

2. Delete

3. DISPLAY

4. Exit

Enter your choice ; 38+603-24

Please choose correct option to

push the data into circular queue

seen