DATE: PAGE:

3 b) Write a program to simulate the working of the circular queue of integers using an integers provide the following operations 1) Insert, 2) Delete 3) Display
The program should print queue empty, Overflow condition

Algorithm (or) Pseudocode

1) Start
2) set front = -1
3) set Rear = -1
4) set size = N
5) Declare queue(size)

=> Enqueue (Insert)

If (front == 0 and Rear == size - 1
(Front == Rear + 1) then
    print ("queue is full"
else
    if front == -1 then
        front=0
    Endif
    Rear = (Rear + 1) Mod size
    Queue [Rear] = element
    print "Inserted:", element
endif

=> Dequeue (Delete)

procedure Dequeue ()
    if front == -1 Then
        print ("queue is empty")
    else
        print "deleted:", Queue [front]
        if front == Rear Then
            front = -1
            Rear = -1
        Else
            front = (front + 1);
Endif

Check

DATE: PAGE:

=> Display

```
procedure Display()
    if front == -1 then
        print "Queue is empty"
    else
        print "Queue elements"
        i = front
        while True
            print queue[i]
            if i == Rear then
                break
            endif
            i = (i+1) Mod size
        endwhile
    endif
end procedure
```

```
#include <stdio.h>
#define n 100

int queue[n];
int front = -1;
int rear = -1;
void enqueue(int x)
{
  if ((rear+1) % n == front) {
      print("Queue overflow\n");
  }
  else if ( front == -1 && rear == -1) {
        front = rear = 0;
        queue[rear] = x;
  }
```

```
    else {
        rear = (rear + 1) % n;
        queue[rear] = x;
    }
}
void dequeue()
{
    if (front == -1 && rear == -1) {
        printf("Queue is empty\n");
    }
    else if (front == rear) {
        printf("Deleted element: %d\n", queue[front]);
        front = rear = -1;
    }
    else {
        printf("Deleted element: %d\n", queue[front]);
        front = (front + 1) % n;
    }
}
void display() {
    int i;
    if (front == -1 && rear == -1) {
        printf("Queue is empty\n");
    }
    else {
        printf("Queue elements:");
        i = front;
        while (i != rear) {
            printf("%d ", queue[i]);
            i = (i + 1) % n;
        }
        printf("%d\n", queue[rear]);
    }
}
```

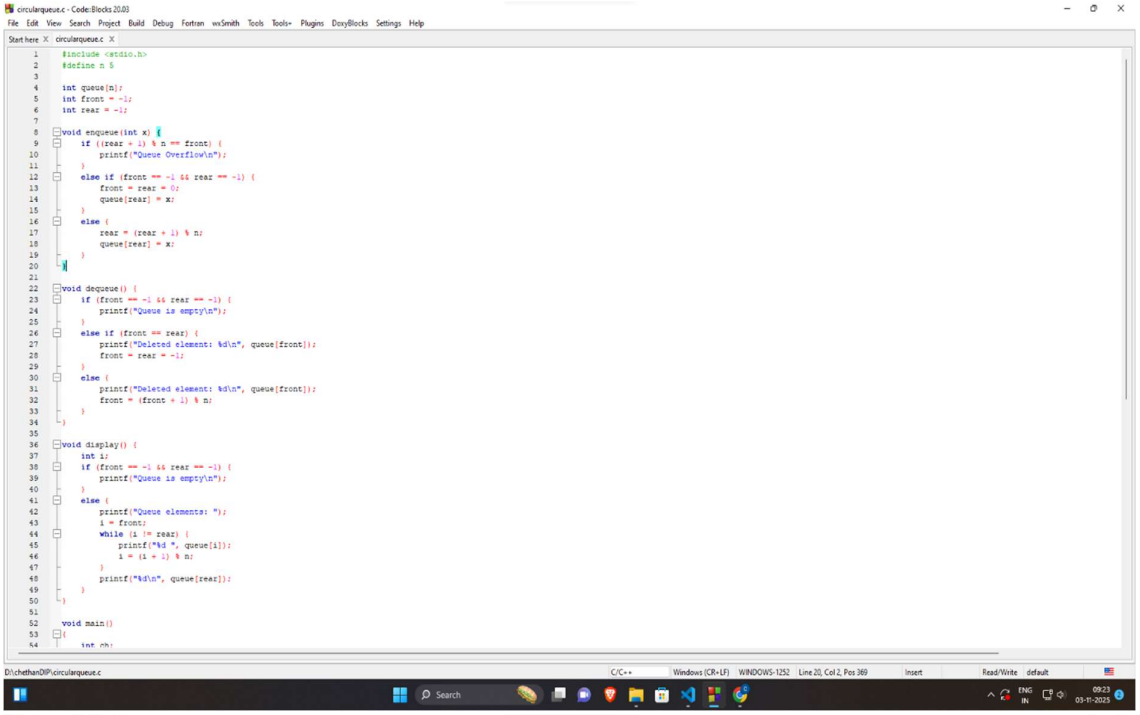
DATE: PAGE:

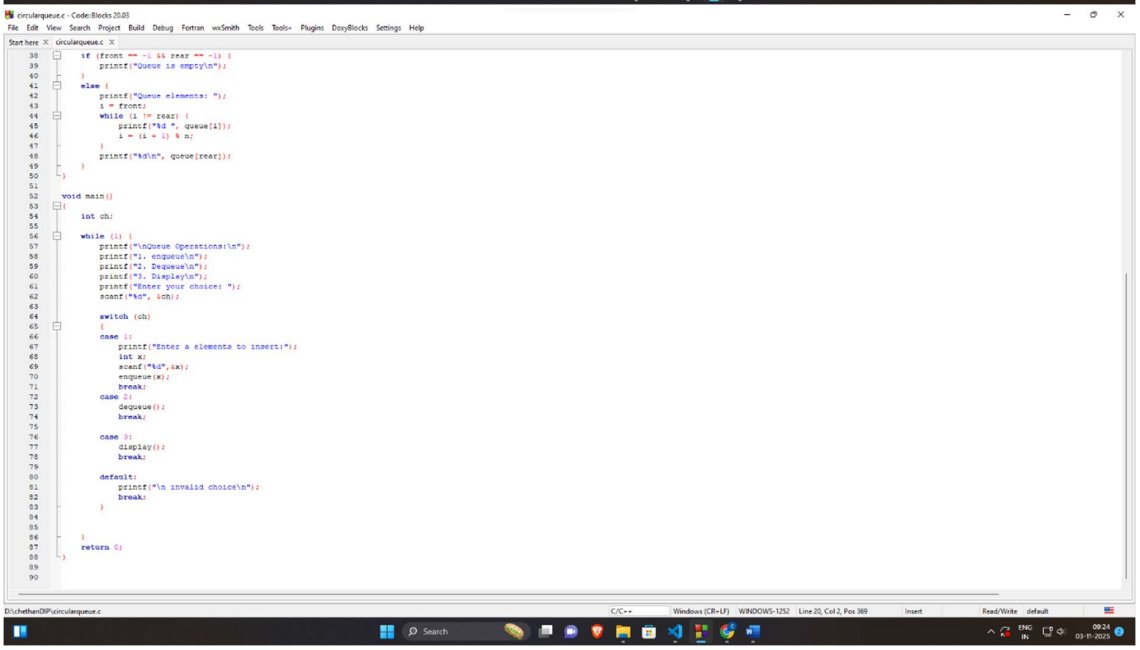
```
void main() {
 int ch;
 while (1) {
    printf("\n queue operations:\n");
    printf("1. enqueue\n");
    printf("2. Dequeue\n");
    printf("3. display\n");
    printf("Enter your choice:");
    scanf("%d", &ch);
    switch (ch)
    {
    case 1:
        printf("Enter a elements to insert");
        int x;
        scanf("%d", &x);
        enqueue(x);
        break;
    case 2:
        dequeue();
        break;
    case 3:
        display();
        break
    default:
        printf("\n invalid choice\n");
        break;
    }
 }
 return 0;
}
```

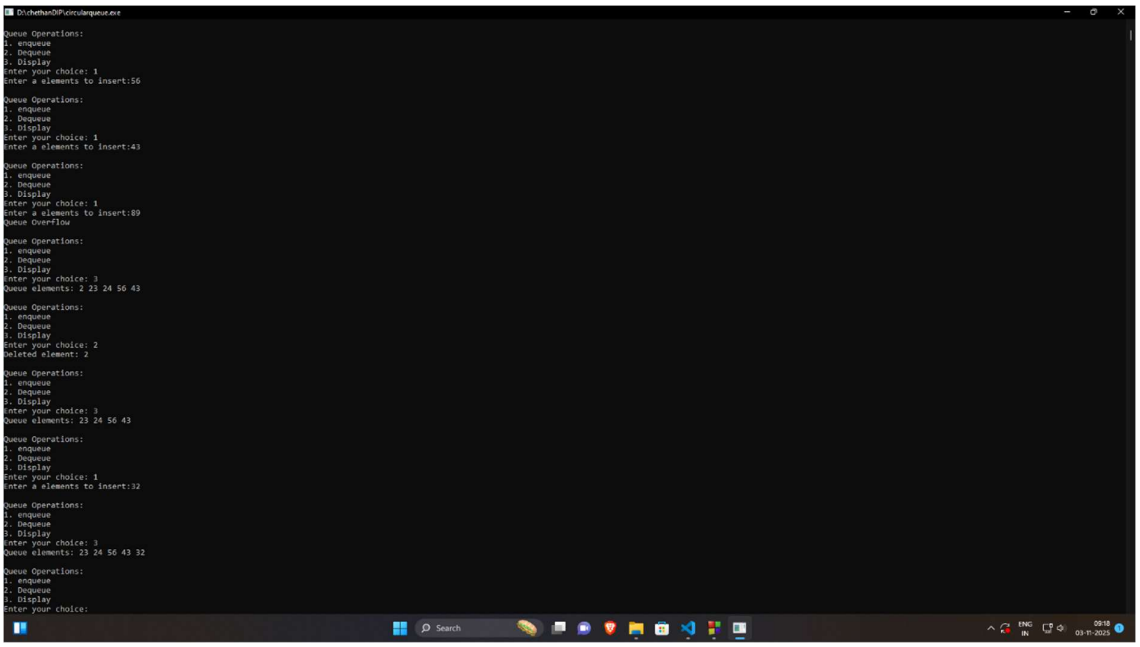
DATE:

Output: Queue operations:
1: enqueue
2. dequeue
3. display
Enter your choice: 1
Enter your elements to insert: 56
Enter your choice: 1
Enter your elements to insert: 43
Enter your choice: 1
Enter your elements to insert: 89
Enter your choice: 1
Enter your elements to insert: 2
Enter your choice: 1
Enter your elements to insert: 4
Queue Flow
Enter your choice: 3
Queue elements: 56 43 89 2
Enter your choice: 2
Deleted element: 56
Enter your choice 3
queue elements: 43 89 2
Enter your choice: 1
Enter your element to insert: 100
Enter your choice: 3
queue element: 43 89 2 100

3/11/25
O/P


circularqueue.c - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
circularqueue.c
#include <stdio.h>
#define n 5
int queue[n];
int front = -1;
int rear = -1;
void enqueue(int x) {
if ((rear + 1) % n == front) {
printf("Queue Overflow\n");
}
else if (front == -1 && rear == -1) {
front = rear = 0;
queue[rear] = x;
}
else {
rear = (rear + 1) % n;
queue[rear] = x;
}
}
void dequeue() {
if (front == -1 && rear == -1) {
printf("Queue is empty\n");
}
else if (front == rear) {
printf("Deleted element: %d\n", queue[front]);
front = rear = -1;
}
else {
printf("Deleted element: %d\n", queue[front]);
front = (front + 1) % n;
}
}
void display() {
int i;
if (front == -1 && rear == -1) {
printf("Queue is empty\n");
}
else {
printf("Queue elements: ");
i = front;
while (i != rear) {
printf("%d ", queue[i]);
i = (i + 1) % n;
}
printf("%d\n", queue[rear]);
}
}
void main()
{
int ch;
D:\chethanDIP\circularqueue.c
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 20, Col 2, Pos 369
Insert
Read/Write
default
Search
ENG
IN
09:23
03-11-2025


circularqueue.c - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
circularqueue.c
if (front == -1 && rear == -1) {
printf("Queue is empty\n");
}
else {
printf("Queue elements: ");
i = front;
while (i != rear) {
printf("%d ", queue[i]);
i = (i + 1) % n;
}
printf("%d\n", queue[rear]);
}
}
void main()
{
int ch;
while (1) {
printf("\nQueue Operations:\n");
printf("1. enqueue\n");
printf("2. Dequeue\n");
printf("3. Display\n");
printf("Enter your choice: ");
scanf("%d", &ch);
switch (ch)
{
case 1:
printf("Enter a elements to insert:");
int x;
scanf("%d",&x);
enqueue(x);
break;
case 2:
dequeue();
break;
case 3:
display();
break;
default:
printf("\n invalid choice\n");
break;
}
}
return 0;
}
D:\chethanDIP\circularqueue.c
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 20, Col 2, Pos 369
Insert
Read/Write
default
Search
ENG
IN
09:24
03-11-2025


D:\chethanDIP\circularqueue.exe
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 1
Enter a elements to insert:56
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 1
Enter a elements to insert:43
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 1
Enter a elements to insert:89
Queue Overflow
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 3
Queue elements: 2 23 24 56 43
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 2
Deleted element: 2
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 3
Queue elements: 23 24 56 43
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 1
Enter a elements to insert:32
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice: 3
Queue elements: 23 24 56 43 32
Queue Operations:
1. enqueue
2. Dequeue
3. Display
Enter your choice:
Search
ENG
IN
09:18
03-11-2025