WAP to stimulate the working of a circular queue of integers using an array. Provide the following operations insert, delete and display. The program should print appropriate messages for queue empty, queue overflow condition.

Pseudo code:

```
# define N 5
int queue [N]
int front = -1
int rear = -1
void enqueue (int x)
  {
    if (rear == N-1) if (front == 0 && rear == N-1)
     {
      printf ("Queue Overflow\n");
     }
    else if (front == -1)
      { front = 0;
        rear = (rear + 1) % N;
        queue [rear] = x;
        printf ("Inserted %d\n", x);
      }
  }
void dequeue()
  { if (front == -1)
    { printf ("Queue empty\n");
  else
      { printf ("Deleted element = %d\n",
                queue[front]);
```

```
if (front == rear)
  {
    front = rear = -1; // queue becomes emp
  }
else {
       front = (front + 1) % N;
     }
}
Void display ()
{ if (front == -1 && rear == -1)
    {
      printf ("Queue is empty");
    }
  else if (rear == front)
    set rear = front = -1.
  else
      setinti = front;
  for (i = front ; i <= rear ; i++);
      set i = (i + 1) % N
  printf ("%d", queue [ i ]);
```

MG.

Code:

```
#include <stdio.h>
#define size 5
int queue[size];
int front = -1, rear = -1;
void enqueue(int value)
{ if ((rear +1) % size == front)
  { printf("\nQueue overflow");
    return;
  }
  if (front == -1) {
      front = rear = 0;
  } else {
      rear = (rear +1) % size;
  }
  queue[rear] = value;
  printf("\n value of %d inserted in the
        queue.", value);
}
int dequeue()
{ if (front == -1)
  {
    printf("\n Queue is Empty, Underflo
    return -1;
  }
  int item = queue[front];
  if (front != rear) {
    front = rear = -1;
```

```
else
  {
    front = (front + 1) % size;
  }
  printf ("\n Value of %d deleted from
          the queue.", item);
  return item;
}
void display ()
{ if (front == -1)
    {
      printf ("\nQueue is empty.");
      return ;
    }
    printf ("\n Queue items are: ");
     int i = front;
     while (i != rear ) {
        printf ("%d ", queue [i]);
         i = (i+1) % size ;
     }
     printf ("%d \n", queue [rear]);
  }
int main ()
int choice, value ;
  printf ("\n options Available : \n");
  printf ("1. En Queue \n2. DeQueue \n3. Display\
          Exit \n");
```

```
while (1) {
    printf("\nEnter your choice : ");
    scanf("%d", &choice);

switch (choice)
    {
  case 1;
    printf("Enter value to insert : ");
     scanf("%d", &value);
     enqueue (value);

     break;
   case 2:
    dequeue ();
     break;
   case 3:
     display ();

     break;
   case 4:
      printf ("Exiting program.\n");
       return 0;
    default:
      printf ("Invalid choice\n");

   }
  }
}
```

Output:

Options Available:

1. EnQueue
2. DeQueue
3. Display
4. Exit

Enter your choice: 2

Queue is Empty, Underflow

Enter your choice: 1

Enter value to insert: 1

Value of 1 inserted in the queue.

Enter your choice: 1

Enter value to insert: 2

Value of 2 inserted in the queue

Enter your choice: 1

Enter value to insert: 3

Value of 3 inserted in the queue

Enter your choice: 1

Enter value to insert: 4

value of 4 inserted in the queue

Enter your choice: 1

Enter value to insert: 5

Value of 5 inserted in the queue

Enter your choice : 1
Enter value to insert: 6
Queue overflow

Enter your choice : 3
Queue items are : 1, 2, 3, 4, 5

Enter your choice : 2
value of 1 deleted from the queue.

Enter your choice : 3
Queue items are : 2, 3, 4, 5

Enter your choice : 4
Exiting program.

3/11/25