Program

```c
#include<stdio.h>
# define size 7
int queue[size], front = -1, rear = -1, item;

void enqueue(int item)
{
    if ((rear + 1) % size == front)
    {
        printf("Overflow Condition\n");
    }
    else if (front == -1 && rear == -1)
    {
        front = rear = 0;
        queue[rear] = item;
    }
    else
    {
        rear = (rear + 1) % size;
        queue[rear] = item;
    }
}

void dequeue()
{
    if (front == -1 && rear == -1)
    {
        printf("Underflow Condition\n");
    }
    else if (front == rear)
    {
        printf("%d ", queue[front]);
        front = rear = -1;
    }
    else
    {
        printf("%d ", queue[front]);
        front = (front + 1) % size;
    }
}

void display()
{
    if (front == -1)
    {
        printf("Queue is Empty\n");
    }
    else
    {
        printf("Queue elements: ");
        int i = front;
        while (i != rear)
        {
            printf("%d ", queue[i]);
            i = (i + 1) % size;
        }
        printf("%d\n", queue[rear]);
    }
}

void main()
```

Experiment No: 9
Date:

CIRCULAR QUEUE

Aim:

To implement Circular Queue.

Algorithm:

1. Start
2. Define size as 7.
3. Create queue of size 7, and initialize front = -1, rear = -1.
4. Create function enqueue(item)

```
If (rear + 1) % size == front:
      Print "Overflow Condition"
Else if front == -1 && rear == -1:
      Set front = 0
      Set rear = 0
      Set queue[rear] = item
Else:
      Set rear = (rear + 1) % size
      Set queue[rear] = item
```

5. Create function dequeue()

```
If front == -1 && rear == -1:
    Print "Underflow Condition"
Else if front == rear:
    Print queue[front]
    Set front = -1 and rear = -1
Else:
    Print queue[front]
    Set front = (front + 1) % size
```

6. Create function Display()

```
If front == -1:
    Print "Queue is Empty"
Else:
```

```
{
    int choice;
    char y;
    do
    {
        printf("1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n");
        printf("Choose an option : ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            printf("Enter the element: ");
            scanf("%d", &item);
            enqueue(item);
            break;
        case 2:
            dequeue();
            break;
        case 3:
            display();
            break;
        default:
            printf("Exit....\n");
            break;
        }
        printf("Do you want to continue (y/n) : ");
        scanf(" %c", &y);

    } while (y == 'Y' || y == 'y');
}
```

Output

```
1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 1
Enter the element: 10
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 1
Enter the element: 20
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 1
Enter the element: 30
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
```

```
Print "Queue elements: "
Set i = front
While i != rear:
    Print queue[i]
    Set i = (i + 1) % size
Print queue[rear]
```

7. Create function main()

```
Repeat until user chooses to exit:
Print "1.Enqueue 2.Dequeue 3.Display 4.Exit"
Read choice
If choice == 1:
   Print "Enter the element"
   Read item
   Call enqueue(item)
If choice == 2:
   Call dequeue()
If choice == 3:
   Call display()
If choice == 4:
   Print "Exit"
   Exit loop
Print "Do you want to continue? (y/n)"
Read y
```

```
Choose an option : 3
Queue elements: 10 20 30
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 1
Enter the element: 40
Overflow Condition
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 2
10
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 2
20
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 2
30
Do you want to continue (y/n) : y

1. Enqueue
2. Dequeue
3. Display
4. Exit
Choose an option : 2
Underflow Condition
Do you want to continue (y/n) : n
```

Result:

Program has been executed successfully and obtained the output