MCA Semester 1	Subject : Advanced Data Structures Lab
Name : Mukund Gangurde	Topic: Circular Queues
Roll No. : MCA2511	Date : 30-10-2025

1) Working of Circular Queues

Code:

081ACQueue.java

```
import java.util.*;

class CQueue
{
	int max;
	int[] cqArray;
	int front;
	int rear;
	int count;

	//Constructor
	public CQueue(int size)
	{
		max = size;
		cqArray = new int[max];
		front = -1;
		rear = -1;
		count = 0;
	}

	//Enqueue
	public void Enqueue(int x)
	{
		//1. Check Queue is Full
		if(count == max)
		{
			System.out.println("Queue Overflowed!");
			return;
		}
		else
		{
			//2. 1st element in the queue
			if(front == -1)
			{
				front = 0;
```

```
                rear = 0;
            }
            else
            {
                //3. Any other element
                rear = (rear+1)%max;
            }

            //4. Insert the element at the row
            cqArray[rear] = x;

            //5. Display the inserted element
            System.out.println("Element inserted is: " + x);
            count++;
        }
    }

    //Dequeue
    public void Dequeue()
    {
        //1. Check Queue is Empty
        if(count == 0)
        {
            System.out.println("Queue Underflowed!");
            return;
        }

        int x = cqArray[front];

        //Display the deleted element
        System.out.println("Element Deleted is: " + x);

        if(front==rear)
        {
            front = -1;
            rear = -1;
        }
        else
        {
            front = (front+1)%max;
        }

        count--;
    }

    //PeekFront
```

```
	public void PeekFront()
	{
		//1. Check Queue is Empty
		if(count == 0)
		{
			System.out.println("Queue Underflowed!");
			return;
		}
		else
		{
			System.out.println("Element at Front: " + cqArray[front]);
		}

	}

	//PeekRear
	public void PeekRear()
	{
		if (count == 0)
		{
		System.out.println("Queue is empty!");
			return;
		}
		System.out.println("Element at Rear: " + cqArray[rear]);
	}

	//Display
	public void Display()
	{
		if (count == 0)
		{
		System.out.println("Queue is empty!");
			return;
		}

		int i,j;
		j = front;
		for(i=1; i<=count; i++)
		{
			System.out.print(cqArray[j] + " ");
			j = (j+1) % max;
		}
	}

}
```

```
class ACQueue
{
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);

		CQueue q = new CQueue(4);
		int ch;

		do
		{
			System.out.println("\nCircular Queue\n");
			System.out.println("1. Enqueue an element");
			System.out.println("2. Dequeue an element");
			System.out.println("3. Display the queue");
			System.out.println("4. Peek Front");
			System.out.println("5. Peek Rear");
			System.out.println("6. Exit\n");

			System.out.print("Enter your Choice: ");
			ch = sc.nextInt();

			switch(ch)
			{
				case 1:
					System.out.println("Enter an element: ");
					int x = sc.nextInt();
					q.Enqueue(x);
					break;
				case 2:
					q.Dequeue();
					break;
				case 3:
					q.Display();
					break;
				case 4:
					q.PeekFront();
					break;
				case 5:
					q.PeekRear();
					break;
				case 6:
					System.out.println("Exiting");
					break;
				default:
					System.out.println("Incorrect Choice!");
```

```
                                    break;
                            }
                    } while (ch!=6);
            }//end of psvm

}
```

Output:

```
A:\MCA2511\DS_LAB>java ACQueue

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
20
Element inserted is: 20

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
30
Element inserted is: 30
```

```
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
40
Element inserted is: 40

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
50
Element inserted is: 50

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
60
Queue Overflowed!
```

```
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 3
20 30 40 50
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 4
Element at Front: 20

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 5
Element at Rear: 50
```

```
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 2
Element Deleted is: 20

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 3
30 40 50
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 2
Element Deleted is: 30

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit
```

```
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 3
40 50
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
60
Element inserted is: 60

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 3
40 50 60
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
```

```
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 1
Enter an element:
60
Element inserted is: 60

Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 3
40 50 60
Circular Queue

1. Enqueue an element
2. Dequeue an element
3. Display the queue
4. Peek Front
5. Peek Rear
6. Exit

Enter your Choice: 6
Exiting
```