MET
Bhujbal Knowledge City
MET INSTITUTE OF COMPUTER SCIENCE

Advanced Data Structures Lab
ACADEMIC YEAR 2025-2026

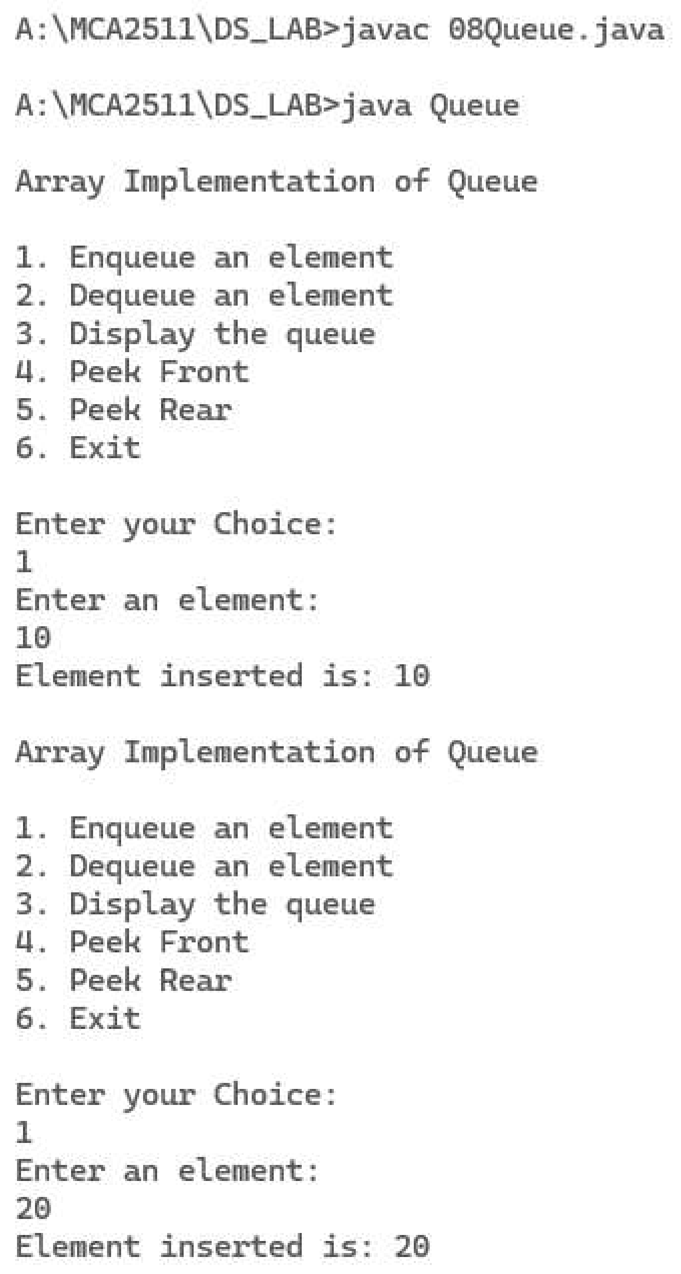
MCA Semester 1	Subject : Advanced Data Structures Lab
Name : Mukund Gangurde	Topic: Queue
Roll No. : MCA2511	Date : 27-10-2025

1) Array Based Ordinary Queues

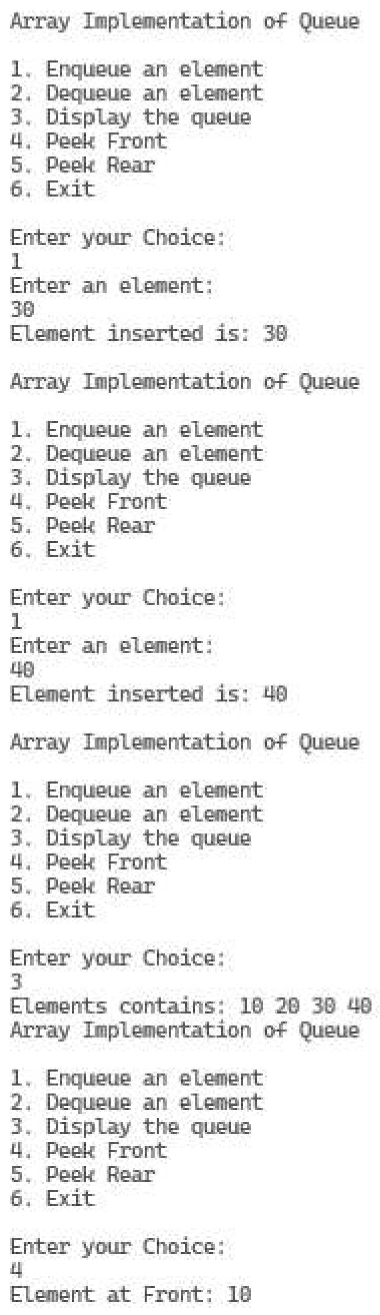
Code:
08Queue.java
```
import java.util.*;

class AQueue
{
	private int max;
	private int[] qArray;
	private int front;
	private int rear;

	public AQueue(int size)
	{
		max = size;
		qArray = new int[max];
		front = -1;
		rear = -1;
	}

	public void Enqueue(int x)
	{
		//1. Queue Full
		if(rear == max-1)
		{
			System.out.println("Queue Overflow!");
			return;
		}

		//2. 1st element in the queue
		if(front == -1)
		{
			front = 0;
			rear = 0;
		}

		//3. Any other element
		else
		{
			rear++;
		}

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

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

	//Dequeue
	public void Dequeue()
	{
		//1. Queue Empty
		if(front == -1)
		{
			System.out.println("Queue Underflow!");
			return;
		}

		//2. Store the element at the front in x
		int x = qArray[front];

		//3. Single element deletion
		if(front == rear)
		{
			front = -1;
			rear = -1;
		}

		//4. Any other element
		else
		{
			front++;
		}

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

	//PeekFront
	public void PeekFront()
	{
		//1. Queue Empty
		if(front == -1)
		{
			System.out.println("Queue Underflow!");
			return;
		}
		else
		{
			System.out.println("Element at Front: " + qArray[front]);
		}
	}

	//PeekRear
	public void PeekRear()
	{
		//1. Queue Full
		//if(rear == max-1)
		//{
			//System.out.println("Queue Overflow!");
			//return;
		//}
		//else
		//{
			System.out.println("Element at Rear: " + qArray[rear]);
		//}
	}

	//Display
	public void Display()
	{
		//1. Queue Empty
		if(front == -1)
		{
			System.out.println("Queue Empty!");
			return;
		}
		else
		{
			System.out.print("Elements contains: ");
			for(int i = front; i<=rear;i++)
			{
				System.out.print(qArray[i] + " ");
			}
		}
	}

}//end of class AQueue
```

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

                AQueue q = new AQueue(4);
                int x, ch;

                do
                {
                        System.out.println("\nArray Implementation of 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.println("Enter your Choice: ");
                        ch = sc.nextInt();

                        switch(ch)
                        {
                                case 1:
                                        System.out.println("Enter an element: ");
                                        x = sc.nextInt();
                                        q.Enqueue(x);
                                        break;
                                case 2:
                                        //System.out.println("Enter an element: ");
                                        //x = sc.nextInt();
                                        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
}//end of class Queue
```

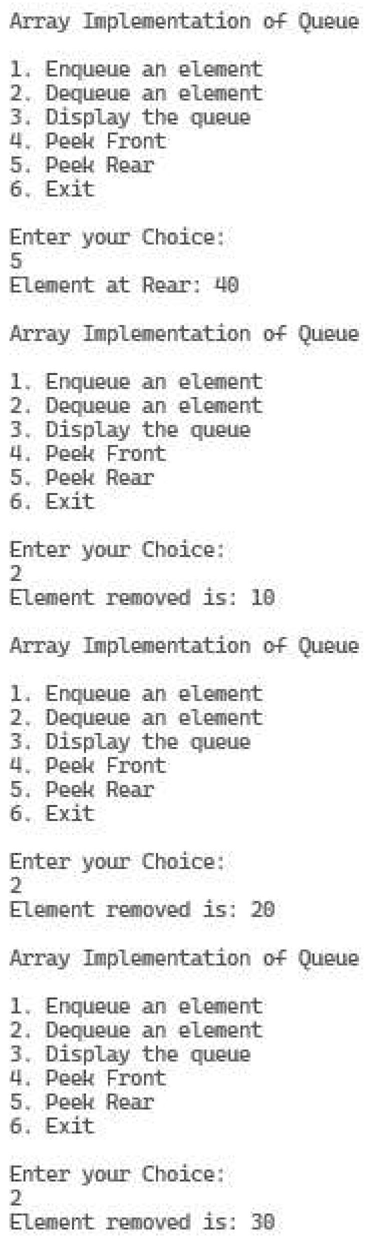
Output:



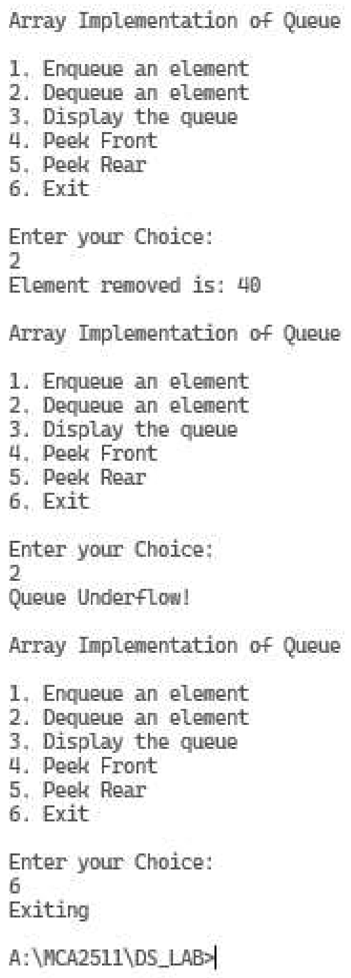
```
A:\MCA2511\DS_LAB>javac 08Queue.java

A:\MCA2511\DS_LAB>java Queue

Array Implementation of 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:
10
Element inserted is: 10

Array Implementation of 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
```



```
Array Implementation of 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

Array Implementation of 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

Array Implementation of 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
Elements contains: 10 20 30 40
Array Implementation of 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: 10
```



```
Array Implementation of 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: 40

Array Implementation of 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 removed is: 10

Array Implementation of 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 removed is: 20

Array Implementation of 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 removed is: 30
```



```
Array Implementation of 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 removed is: 40

Array Implementation of 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
Queue Underflow!

Array Implementation of 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

A:\MCA2511\DS_LAB>
```

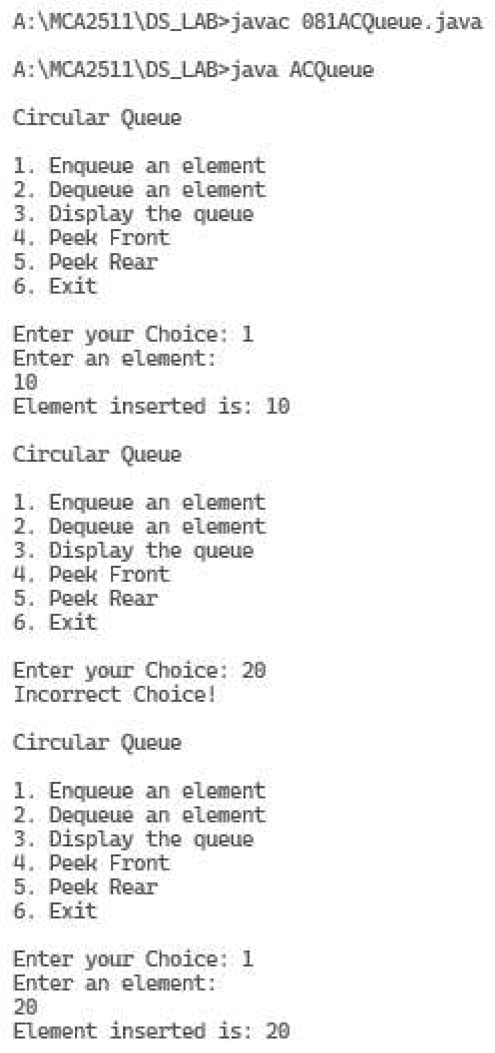
2) Array Based Circular Queues

Code:
081ACQueue.java

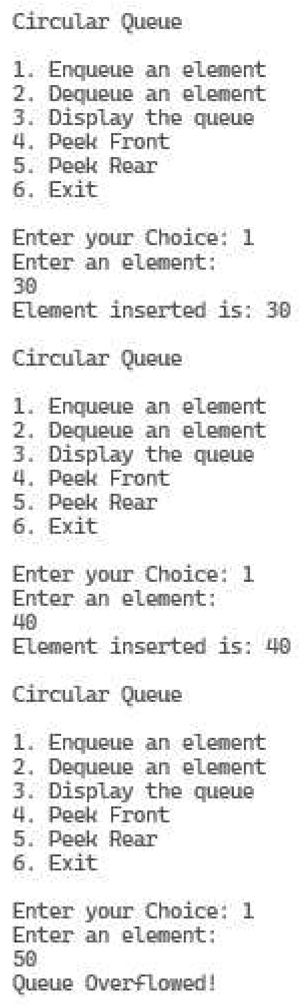
```
import java.util.*;

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;
		}

		//2. 1st element in the queue
		if(front == -1)
		{
			front = 0;
			rear = 0;
		}
		//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!");
                }

        }

        //PeekFront
        public void PeekFront()
        {

        }

        //PeekRear
        public void PeekRear()
        {

        }

        //Display
        public void Display()
        {

        }

}

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>javac 081ACQueue.java

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:
10
Element inserted is: 10

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: 20
Incorrect Choice!

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
Queue Overflowed!
```