NORMAL QUEUE=>

```
import java.util.Scanner;

class Queue
{
	private int Q[],front,rear,MaxSize;
	public Queue(int size)
{
		MaxSize=size;
		front=0;
		rear=-1;
		Q=new int[MaxSize];
}
	public void Enqueue(int e)
	{
		rear++;
		Q[rear]=e;
	}
	public boolean isFull()
	{
		if(rear==MaxSize-1)
			return(true);
		else
			return(false);
	}
	public int Dequeue()
	{
		int temp=Q[front];
		front++;
		return (temp);
	}
	public boolean isEmpty()
	{
		if(front>rear)
			return(true);
		else
			return(false);
	}
	public void printQueue()
	{
		for(int i=front;i<=rear;i++)
			System.out.println(Q[i]);
	}}
	public class Normal
	{
		public static void main(String[] args)
		{
			int value,choice;
			Scanner sc =new Scanner(System.in);
			Queue q= new Queue(5);

			do {
				System.out.println("1. Enqueue  \n2. Dequeue  \n3.
PrintQueue  \nEnter choice: ");
```

```
                        choice=sc.nextInt();

                        switch(choice) {

                           case 1 : if(q.isFull())
                                        System.out.println("Queue full");
                                    else
                                    {
                                           System.out.println("Enter the
Element");
                                           value=sc.nextInt();
                                           q.Enqueue(value);
                                    }
                                   break;
                           case 2 : if(q.isEmpty())
                                       System.out.println("Queue Empty");
                                    else
                                    {
                                           value=q.Dequeue();
                                           System.out.println("removed value :
"+value);
                                    }
                               break;
                           case 3  :if(q.isEmpty())
                                           System.out.println("Stack is Empty");
                                           else
                                           {
                                                  q.printQueue();

                                           }
                                           break;

                                   }
                        }while(choice!=0);
                }
        }
*****************************************************************************
CIRCULAR QUEUE=>

import java.util.Scanner;

class Test
{
        private int Q[],front,rear,MaxSize,count;
        public Test(int size)
{
                count=0;
                MaxSize=size;
                front=0;
                rear=-1;
                Q=new int[MaxSize];
}
        public void Enqueue(int e)
        {
                count++;
```

```
		rear=(rear+1)%MaxSize;
		Q[rear]=e;
	}
	public boolean isFull()
	{
		if(count==MaxSize)
			return(true);
		else
			return(false);
	}
	public int Dequeue()
	{
		count--;
		int temp=Q[front];
		front=(front+1)%MaxSize;
		return (temp);
	}
	public boolean isEmpty()
	{
		if(count==0)
			return(true);
		else
			return(false);
	}
	public void printQueue()
	{
		int i,c;
		i=front;
		for(c=0;c<count;c++)
		{
			System.out.println(Q[i]);
			i=(i+1)%MaxSize;
		}
	}
}
	public class Circular
	{
		public static void main(String[] args)
		{
			int value,choice;
			Scanner sc =new Scanner(System.in);
			Test q= new Test(5);

			do {
				System.out.println("\n1. Enqueue  \n2. Dequeue  \n3.
PrintQueue  \nEnter choice: ");
				choice=sc.nextInt();

				switch(choice) {

				   case 1 : if(q.isFull())
						System.out.println("Queue full");
					    else
					    {
```

```
                                        System.out.println("Enter the
Element");
                                        value=sc.nextInt();
                                        q.Enqueue(value);
                                    }
                                   break;
                            case 2 : if(q.isEmpty())
                                        System.out.println("Queue Empty");
                                     else
                                     {
                                        value=q.Dequeue();
                                        System.out.println("removed value :
"+value);
                                     }
                                break;
                            case 3  :if(q.isEmpty())
                                        System.out.println("Stack is Empty");
                                        else
                                        {
                                              q.printQueue();
                                        }
                                        break;

                                }
                        }while(choice!=0);
                }
        }
*********************************************************************************
PRIORITY QUEUE=>

import java.util.Scanner;

class Demo
        {
                private int Q[];
                int front,rear,MaxSize;
                public Demo(int size)
                {
                        MaxSize=size;
                        front=0;
                        rear=-1;
                        Q=new int[MaxSize];
                }
                public void Enqueue(int e)
                {
                        int i,j,temp;
                rear++;
                Q[rear]=e;
                {
                        for(i=front;i<rear;i++)
                        {
                                for(j=front;j<rear;j++)
                                {
                                        if(Q[j]>Q[j+1])
                                        {
```

```
                                        temp=Q[j];
                                        Q[j]=Q[j+1];
                                        Q[j+1]=temp;
                                }
                        }
                }
        }
        }
        public boolean isFull()
        {
                if(rear==MaxSize-1)
                        return(true);
                else
                        return(false);
        }
        public int Dequeue()
        {
                int temp=Q[front];
                front++;
                return (temp);
        }
        public boolean isEmpty()
        {
                if(front>rear)
                        return(true);
                else
                        return(false);
        }
        public void printQueue()
        {
                for(int i=front;i<=rear;i++)
                        System.out.println(Q[i]);
        }}
        public class Priority
        {
                public static void main(String[] args)
                {
                        int value,choice;
                        Scanner sc =new Scanner(System.in);
                        Queue q= new Queue(5);

                        do {
                                System.out.println("1. Enqueue  \n2. Dequeue  \n3.
PrintQueue  \nEnter choice: ");
                                choice=sc.nextInt();

                                switch(choice) {

                                   case 1 : if(q.isFull())
                                                System.out.println("Queue full");
                                            else
                                            {
                                                  System.out.println("Enter the
Element");

                                                  value=sc.nextInt();
```

```
                                                q.Enqueue(value);
                                            }
                                           break;
                                    case 2 : if(q.isEmpty())
                                                System.out.println("Queue Empty");
                                             else
                                             {
                                                value=q.Dequeue();
                                                System.out.println("removed
value : "+value);
                                             }
                                        break;
                                    case 3  :if(q.isEmpty())
                                                System.out.println("Stack is
Empty");
                                             else
                                             {
                                                   q.printQueue();

                                             }
                                             break;

                                        }
                                  }while(choice!=0);
                        }
                }
```

****