NAME: KAMRAN ANSARI
REG NO: 22MCA0223
QUESTION NO: 1

1. Given an integer k and a queue of integers, we need to reverse the order of the first k elements of the queue, leaving the other elements in the same relative order. Implement this program dynamically and only following standard operations are allowed on queue. a. enqueue(x) : Add an item x to rear of queue b. dequeue() : Remove an item from front of queue c. size() : Returns number of elements in queue. d. front() : Finds front item Input : Q = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100], k = 5 Output : Q = [50, 40, 30, 20, 10, 60, 70, 80, 90, 100]

Node.java

```
public class Node {
    public int value;
    public Node next;

    public Node(int value) {
        this.value = value;
        this.next = null;
    }
}
```

Queue.java

```
public class Queue {
    public Node front;
    public Node rear;
    private int size;

    public Queue() {
        front = null;
        rear = null;
        size = 0;
    }

    public int size() {
        return size;
    }

    public int front() {
        return front.value;
    }

    public boolean enqueue(int x) {
```

```
    Node newNode = new Node(x);

    if (front == null && rear == null) {
        front = newNode;
        rear = newNode;
        size++;
        return true;
    }

    rear.next = newNode;
    rear = newNode;
    size++;
    return true;
}

public Integer dequeue() {
    if (front == null && rear == null) {
        System.out.println("Queue Emtpy");
        return null;
    }

    if (front == rear) {
        int valueToReturn = front.value;
        front = null;
        rear = null;
        size = 0;
        return valueToReturn;
    }

    int dequeuedValue = front.value;
    front = front.next;
    size--;
    return dequeuedValue;
}

public void print() {
    Node itr = front;

    System.out.println("");

    if (itr == null) {
        System.out.println("Queue Empty");
    }

    while (itr != null) {
```

```
            System.out.print(itr.value + " ");
            itr = itr.next;
        }
    }
}
```

Stack.java

```
public class Stack {
    public int[] arr;
    public int size;
    private int lastFreeIndex;

    Stack() {
        size = 50;
        arr = new int[size];
        lastFreeIndex = 0;
    }

    public boolean isEmpty() {
        return lastFreeIndex == 0;
    }

    public void push(int value) {
        if (lastFreeIndex >= size - 1) {
            System.out.println("Overlfow");
            return;
        }

        arr[lastFreeIndex++] = value;
    }

    public Integer pop() {
        if (lastFreeIndex <= 0) {
            System.out.println("Underflow");
            return null;
        }

        return arr[--lastFreeIndex];
    }

    public void print() {
        if (lastFreeIndex == 0) {
            System.out.println("Empty");
            return;
        }
```

```
        for (int i = 0; i < lastFreeIndex; i++) {
            System.out.println(arr[i] + " ");
        }
    }
}
```

Main.java

```
import java.util.Scanner;

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

        System.out.println("Enter Queue Size: ");
        int queueSize = Integer.valueOf(scanner.nextLine());

        System.out.println("Enter Queue: ");
        for (int i = 0; i < queueSize; i++) {
            System.out.println("Enter element " + (i + 1));
            int valueToInsert = Integer.valueOf(scanner.nextLine());

            queue.enqueue(valueToInsert);
        }

        System.out.println("Enter the number of elements to reverse: ");
        int numberOfElementsToReverse = Integer.valueOf(scanner.nextLine());

        reverse(queue, numberOfElementsToReverse);

        System.out.println("Queue with " + numberOfElementsToReverse + " elements
reversed is: ");
        queue.print();
        System.out.println("");
    }

    public static void reverse(Queue q, int k) {
        Stack stack = new Stack();
        Queue auxiallryQueue = new Queue();

        int i = 0;

        while (i < k && q.size() > 0) {
            stack.push(q.dequeue());
```

```
            i++;
        }

        while (q.size() > 0) {
            auxiallryQueue.enqueue(q.dequeue());
        }

        while (!stack.isEmpty()) {
            q.enqueue(stack.pop());
        }

        while (auxiallryQueue.size() > 0) {
            q.enqueue(auxiallryQueue.dequeue());
        }
    }
}
```

OUTPUT

```
Enter Queue Size:
10
Enter Queue:
Enter element 1
10
Enter element 2
20
Enter element 3
30
Enter element 4
40
Enter element 5
50
Enter element 6
60
Enter element 7
70
Enter element 8
80
Enter element 9
90
Enter element 10
100
Enter the number of elements to reverse:
5
Queue with 5 elements reversed is:

50 40 30 20 10 60 70 80 90 100
```