write a java program to implement the simple queue data structure where 2 queues are needed say q1 and q2 - in which q1 can only have odd values while q2 can only have even value. But you must create a single display method. now, when the display method is called by the user it shall print in the following sequece - first element of q1, first element of q2, second element of q1, second element of q2 and so on

public class QueueExample {

private static final int QUEUE\_SIZE = 100;

private int[] q1;

private int[] q2;

private int front1, rear1;

private int front2, rear2;

public QueueExample() {

q1 = new int[QUEUE\_SIZE];

q2 = new int[QUEUE\_SIZE];

front1 = front2 = -1;

rear1 = rear2 = -1;

}

public void enqueue(int value) {

if (value % 2 == 0) {

if (rear2 == QUEUE\_SIZE - 1) {

System.out.println("Queue 2 is full.");

return;

}

if (front2 == -1)

front2 = 0;

q2[++rear2] = value; // Even values go to q2

} else {

if (rear1 == QUEUE\_SIZE - 1) {

System.out.println("Queue 1 is full.");

return;

}

if (front1 == -1)

front1 = 0;

q1[++rear1] = value; // Odd values go to q1

}

}

public void display() {

int i = front1;

int j = front2;

while (i <= rear1 && j <= rear2) {

System.out.print(q1[i++] + " ");

System.out.print(q2[j++] + " ");

}

while (i <= rear1)

System.out.print(q1[i++] + " ");

while (j <= rear2)

System.out.print(q2[j++] + " ");

System.out.println();

}

public static void main(String[] args) {

QueueExample queueExample = new QueueExample();

queueExample.enqueue(1);

queueExample.enqueue(2);

queueExample.enqueue(3);

queueExample.enqueue(4);

queueExample.enqueue(5);

queueExample.enqueue(6);

queueExample.display(); // Output: 1 2 3 4 5 6

}

}