public static List<Integer> joinList(List<Integer> list1, List<Integer> list2){  
 List<Integer> mergedList = new ArrayList<>();  
 int i;  
 for(i=0;i<Math.min(list1.size(), list2.size());i++){  
 mergedList.add(list1.get(i));  
 mergedList.add(list2.get(i));  
 }  
  
 if(list1.size() > list2.size()){  
 for(int j=i;j<list1.size();j++){  
 mergedList.add(list1.get(j));  
 }  
 }else {  
 for(int j=i;j<list2.size();j++){  
 mergedList.add(list2.get(j));  
 }  
 }  
 return mergedList;  
 }

public static List<Integer> joinList(List<Integer> list1, List<Integer> list2){  
 List<Integer> mergedList = new ArrayList<>();  
 mergedList.addAll(list1);  
 mergedList.addAll(list2);  
 return mergedList;  
 }

private static List<Integer> joinList(List<Integer> list1, List<Integer> list2){  
 List<Integer> mergedList = new ArrayList<>(list1);  
 mergedList.addAll(list2);  
 return mergedList;  
}

public static List<Integer> joinList(List<Integer> list1, List<Integer> list2) {  
 return List.of(list1, list2)  
 .stream()  
 .flatMap(List::stream)  
 .toList();  
 }

public static List<Integer> joinList(List<Integer> list1, List<Integer> list2) {  
 return Stream.of(list1, list2).flatMap(List::stream).toList();  
 }

public static List<Integer> joinList(List<Integer> list1, List<Integer> list2){  
 return Stream.concat(list1.stream(), list2.stream())  
 .collect(Collectors.toList());  
 }

public static List<Integer> joinList(List<Integer> list1, List<Integer> list2){  
 return Stream.concat(list1.stream(), list2.stream())  
 .toList()  
 }

public static List<Integer> joinList(List<Integer>... lists){  
 return Arrays.stream(lists)  
 .flatMap(List::stream).toList();  
 }