题目描述

现在有多组乱序的数组，需要将它们合并成一个新的数组。

合并规则，从每个数组取出顺序取出固定长度的内容合并到新的数组中，取完的内容会删除掉，如果进行不足固定长度或者已经为空，则直接取出剩余部分的内容放到新的数组中，继续下一行。

输入描述

第一行是每次选取的固定长度，0 < 长度 < 10

第二行是整数数组组数目，0 < 组目 < 1000

第3-n行是需要合并的数组，不同的数组用英文逗号分隔，数组内部用空格分隔，最大不超过100个元素。

输出描述

输出一个新的数组，用空格分隔。

用例

输入	3 2 2,5,6,7,9,5,7 1,7,4,3,4
输出	2,5,6,1,7,4,7,9,5,3,4,7
说明	1、获得长度3和数组数目2 2、先读取第一行，获得2,5 3、再读取第二行，获得1,7,4 4、再将第三行第一行，获得7,9,5 5、再读取第二行，获得3,4 6、最后获取一行，获得7，按照先对长度最取完的原

输入	4 3 1,2,3,4,5,6 1,2,3 1,2,3,4
输出	1,2,3,4,1,2,3,1,2,3,4,5,6
说明	无

题目解析

假设，第一行输入的固定长度是k,k1,k2。

则这道题思路是如下：

首先，定义一个 动态数组 * ans ，来记录结果。

然后，将输入的3-n行的数组数组，放到一个容器中。

取n第一组容器的头部，然后将容器头部数组前的k,k1,k2个元素，并以k,k1,k2元素记录的顺序放入ans中！

- 如果剩余后，该头部数组长度依旧大于0，则将该头部数组重新放置到容器尾部
- 如果剩余后，该头部数组的长度为0，则不做操作

直到容器长度为0时，停止。

此时ans中记录的就是结果。

本题，主要难点在于，如何剩余后（或刚好用尽）固定长度k,k1,k2个元素，并实现且 删除容器 * 中第个元素的所有元素。在下面代码中，我们对不同语言，选择了一个特殊，能简单直观地对剩余的容器的第个元素的位置删除。

Java算法源码

LeetCode提供的长度删除指定区域的元素（使用Ferkowa/Range）

```
import java.util.*;
import java.util.stream.Collectors;

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

    int subLen = Integer.parseInt(sc.nextLine());

    int n = Integer.parseInt(sc.nextLine());

    LinkedList<LinkedList<Integer>> lists = new LinkedList<>();
    for (int i = 0; i < n; i++) {
      lists.add(
          new LinkedList<>(
              Arrays.stream(sc.nextLine().split(","))
                  .map(Integer::parseInt)
                  .collect(Collectors.toList())));
    }

    System.out.println(getResult(subLen, lists));
  }

  public static String getResult(int subLen, LinkedList<LinkedList<Integer>> lists) {
    ArrayList<Integer> ans = new ArrayList<>();

    while (lists.size() > 0) {
      LinkedList<Integer> list = lists.removeFirst();
      ans.addAll(removeRange(list, 0, subLen));
      if (list.size() > 0) lists.addLast(list);
    }

    StringJoiner sj = new StringJoiner(",");
    for (Integer an : ans) {
      sj.add(an + "");
    }
    return sj.toString();
  }

  public static List<Integer> removeRange(LinkedList<Integer> list, int start, int end) {
    if (end > list.size()) end = list.size();

    List<Integer> tmp = list.subList(start, end);

    List<Integer> ans = new ArrayList<>(tmp);
    tmp.clear();

    return ans;
  }
}
```

JS算法源码

Array.prototype.splice() - JavaScript | MDN (mozilla.org)

```
// Powered by Moss 记录模式 请勿修改此模板 //
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

const lines = [];
rl.on("line", (line) => {
  lines.push(line);

  if (lines[1] && lines.length === 2 + parseInt(lines[1])) {
    const subLen = lines.shift() - 0;
    lines.shift();
    const lists = lines.map((line) => line.split(",").map(Number));

    console.log(getResult(subLen, lists));

    lines.length = 0;
  }
});

function getResult(subLen, lists) {
  const ans = [];

  while (lists.length > 0) {
    const list = lists.shift();
    ans.push(...list.splice(0, subLen));
    if (list.length > 0) lists.push(list);
  }

  return ans.join(",");
}
```

Python算法源码

python删除列表指定数组一段区域

```
# 输入内容
subLen = int(input())
n = int(input())
lists = [list(i.split(",") for _ in range(n)]

# 定义一
def getResult():
    ans = []

    while len(lists) > 0:
        lst = lists.pop(0)
        tmp = lst[:subLen]
        del lst[:subLen]
        ans.extend(tmp)
        if len(lst) > 0:
            lists.append(lst)

    return ",".join(ans)

# 主函数调用
```