LeetCode NoteBook

Ruohong Jiao
NUAA

2021 年 3 月 14 日

目录

1 LeetCode 题目分类[1]

1.1 Hash 相关 (2)

q1(A.1), q387(A.28).

1.2 链表操作 (6)

q2(A.2), q19(A.12), q25(A.13), q61(A.19), q138(A.21), q206(A.25).

1.3 双指针遍历/滑动窗口 (8)

q3(A.3), q11(A.8), q15(A.10), q16(A.11), q26(A.14), q42(A.15), q121(A.20), q209(A.26).

1.4 快慢指针遍历 (3)

q141(A.22), q202(A.24), q876(A.30).

1.5 区间合并 (1)

q56(A.18).

1.6 字符串操作 (3)

q6(A.4), q14(A.9), q763(A.29).

1.7 数字操作 (6)

q7(A.5), q8(A.6), q9(A.7), q43(A.16), q172(A.23), q258(A.27).

1.8 数组操作 (6)

q54(A.17), q73, q78, q384, q581, q945.

1.9 栈相关 (6)

q20, q32, q155, q224, q232, q316.

1.10 堆相关 (2)

q215, q347.

1.11 递归 (5)

q21, q101, q104, q226, q236.

1.12 分治法/二分法 (3)

q23, q33, q34.

1.13 动态规划 (9)

q5, q53, q62, q64, q70, q118, q300, q1143, q1277.

[1]Github: Guangxin Yuan

1.14 回溯法 (4)

q10, q22, q40, q46.

1.15 树的遍历 (5)

q94, q102, q110, q144, q145.

1.16 二叉搜索树相关 (3)

q98, q450, q701.

附录 A　Problem List

A.1　Leetcode 1

Problem Description:

两数之和

给定一个整数数组 *nums* 和一个目标值 *target*，请你在该数组中找出和为目标值的那两个整数，并返回他们的数组下标。

你可以假设每种输入只会对应一个答案。但是，数组中同一个元素不能使用两遍。

Sample:

input:

```
给定 nums = [2, 7, 11, 15], target = 9
```

otput:

```
因为 nums[0] + nums[1] = 2 + 7 = 9
所以返回 [0, 1]
```

Solution (Codes at B.1):

没有说明输入数字一定是正整数，不能先排序后提取小于 *target* 的数进行求解。

懒一点，n^2 循环。勤快一点用红黑树、堆进行存储然后查询 *nlogn* 。

A.2　Leetcode 2

Problem Description:

两数相加

给出两个非空的链表用来表示两个非负的整数。其中，它们各自的位数是按照逆序的方式存储的，并且它们的每个节点只能存储一位数字。

如果，我们将这两个数相加起来，则会返回一个新的链表来表示它们的和。

您可以假设除了数字 0 之外，这两个数都不会以 0 开头。

Sample:

input:

```
输入：(2 -> 4 -> 3) + (5 -> 6 -> 4)
```

otput:

```
输出：7 -> 0 -> 8
原因：342 + 465 = 807
```

Solution (Codes at B.2):

大数加法，注意指针/引用转换。

A.3　Leetcode 3

Problem Description:

无重复字符的最长子串

给定一个字符串，请你找出其中不含有重复字符的最长子串的长度。

Sample:

input:

```
输入："abcabcbb"

输入："bbbbb"

输入："pwwkew"
```

otput:

```
输出: 3
解释: 因为无重复字符的最长子串是 "abc"，所以其长度为 3。

输出: 1
解释: 因为无重复字符的最长子串是 "b"，所以其长度为 1。

输出: 3
解释: 因为无重复字符的最长子串是 "wke"，所以其长度为 3。
     请注意，你的答案必须是 子串 的长度，"pwke" 是一个子序列，不是子串。
```

Solution (Codes at B.3):

标记 l 和 r，遍历一遍，更新答案。时间方面，能用数组不用 STL。

A.4 Leetcode 6

Problem Description:

Z 字形变换

将一个给定字符串根据给定的行数，以从上往下、从左到右进行 Z 字形排列。

比如输入字符串为"LEETCODEISHIRING" 行数为 3 时，排列如下：

```
L   C   I   R
E T O E S I I G
E   D   H   N
```

之后，你的输出需要从左往右逐行读取，产生出一个新的字符串，比如："LCIRETOESIIGEDHN"。

Sample:

input:

```
输入: s = "LEETCODEISHIRING", numRows = 3

输入: s = "LEETCODEISHIRING", numRows = 4
```

otput:

```
输出: "LCIRETOESIIGEDHN"

输出: "LDREOEIIECIHNTSG"
解释:

L     D     R
E   O E   I I
E C   I H   N
T     S     G
```

Solution (Codes at B.4):

计算第一行字符 id，前后搜查。

A.5 Leetcode 7

Problem Description:

整数反转

给你一个 32 位的有符号整数 x ，返回将 x 中的数字部分反转后的结果。

如果反转后整数超过 32 位的有符号整数的范围 [−231, 231 − 1] ，就返回 0。

假设环境不允许存储 64 位整数（有符号或无符号）。

Sample:

input:

```
输入: x = 123

输入: x = -123

输入: x = 120
```

```
输入：x = 0
```

otput:

```
输出：321

输出：-321

输出：21

输出：0
```

Solution (Codes at B.5):

对 10 取模，倒序重构。（刚开始写的时候没有用 INT_MAX，INT_MIN，用的字符串去判断，很繁。）

A.6 Leetcode 8

Problem Description:

字符串转换整数 (atoi)

请你来实现一个 myAtoi(string s) 函数，使其能将字符串转换成一个 32 位有符号整数（类似 C/C++ 中的 atoi 函数）。

函数 myAtoi(string s) 的算法如下：

读入字符串并丢弃无用的前导空格检查下一个字符（假设还未到字符末尾）为正还是负号，读取该字符（如果有）。确定最终结果是负数还是正数。如果两者都不存在，则假定结果为正。读入下一个字符，直到到达下一个非数字字符或到达输入的结尾。字符串的其余部分将被忽略。将前面步骤读入的这些数字转换为整数（即，”123” -> 123，”0032” -> 32）。如果没有读入数字，则整数为 0 。必要时更改符号（从步骤 2 开始）。如果整数数超过 32 位有符号整数范围 [−231，231 − 1] ，需要截断这个整数，使其保持在这个范围内。具体来说，小于 −231 的整数应该被固定为 −231 ，大于 231 − 1 的整数应该被固定为 231 − 1 。返回整数作为最终结果。

注意：

本题中的空白字符只包括空格字符’’。除前导空格或数字后的其余字符串外，请勿忽略任何其他字符。

Sample:

input:

```
输入：s = "42"

输入：s = "   -42"

输入：s = "4193 with words"

输入：s = "words and 987"

输入：s = "-91283472332"
```

otput:

```
输出：42
解释：加粗的字符串为已经读入的字符，插入符号是当前读取的字符。
第 1 步："42"（当前没有读入字符，因为没有前导空格）
         ^
第 2 步："42"（当前没有读入字符，因为这里不存在 '-' 或者 '+'）
         ^
第 3 步："42"（读入 "42"）
           ^
解析得到整数 42 。
由于 "42" 在范围 [-231, 231 - 1] 内，最终结果为 42 。

输出：-42
解释：
第 1 步："   -42"（读入前导空格，但忽视掉）
            ^
第 2 步："   -42"（读入 '-' 字符，所以结果应该是负数）
```

```
              ^
第 3 步: "   -42" (读入 "42")
                ^
解析得到整数 -42 。
由于 "-42" 在范围 [-231, 231 - 1] 内，最终结果为 -42 。

输出: 4193
解释:
第 1 步: "4193 with words" (当前没有读入字符，因为没有前导空格)
         ^
第 2 步: "4193 with words" (当前没有读入字符，因为这里不存在 '-' 或者 '+')
         ^
第 3 步: "4193 with words" (读入 "4193"; 由于下一个字符不是一个数字，所以读入停止)
             ^
解析得到整数 4193 。
由于 "4193" 在范围 [-231, 231 - 1] 内，最终结果为 4193 。

输出: 0
解释:
第 1 步: "words and 987" (当前没有读入字符，因为没有前导空格)
         ^
第 2 步: "words and 987" (当前没有读入字符，因为这里不存在 '-' 或者 '+')
         ^
第 3 步: "words and 987" (由于当前字符 'w' 不是一个数字，所以读入停止)
         ^
解析得到整数 0 ，因为没有读入任何数字。
由于 0 在范围 [-231, 231 - 1] 内，最终结果为 0 。

输出: -2147483648
解释:
第 1 步: "-91283472332" (当前没有读入字符，因为没有前导空格)
         ^
第 2 步: "-91283472332" (读入 '-' 字符，所以结果应该是负数)
          ^
第 3 步: "-91283472332" (读入 "91283472332")
                     ^
解析得到整数 -91283472332 。
由于 -91283472332 小于范围 [-231, 231 - 1] 的下界，最终结果被截断为 -231 =
    -2147483648 。
```

Solution (Codes at B.6):

用 INT_MAX，INT_MIN 检测数值边界。

A.7 Leetcode 9

Problem Description:

回文数

给你一个整数 x，如果 x 是一个回文整数，返回 true；否则，返回 false 。

回文数是指正序（从左向右）和倒序（从右向左）读都是一样的整数。例如，121 是回文，而 123 不是。

Sample:

input:

```
输入: x = 121

输入: x = -121

输入: x = 10

输入: x = -101
```

otput:

```
输出: true

输出: false
解释: 从左向右读, 为 -121 。 从右向左读, 为 121- 。因此它不是一个回文数。
```

```
输出：false
解释：从右向左读，为 01 。因此它不是一个回文数。

输出：false
```

Solution (Codes at B.7):

用 INT_MAX，INT_MIN 检测数值边界。翻转数字后和原数值比较。

A.8 Leetcode 11

Problem Description:

盛最多水的容器

给你 n 个非负整数 $a1,a2,\cdots,an$，每个数代表坐标中的一个点 (i,ai)。在坐标内画 n 条垂直线，垂直线 i 的两个端点分别为 (i,ai) 和 $(i,0)$。找出其中的两条线，使得它们与 x 轴共同构成的容器可以容纳最多的水。

说明：你不能倾斜容器，且 n 的值至少为 2。

Sample:

input:

```
输入：[1,8,6,2,5,4,8,3,7]
```

otput:

```
输出：49
```

Solution (Codes at B.8):

两边到中间遍历一遍，贪心选择更高的边维持原状。

A.9 Leetcode 14

Problem Description:

最长公共前缀

编写一个函数来查找字符串数组中的最长公共前缀。

如果不存在公共前缀，返回空字符串""。

Sample:

input:

```
输入：strs = ["flower","flow","flight"]

输入：strs = ["dog","racecar","car"]
```

otput:

```
输出："fl"

输出：""
解释：输入不存在公共前缀。
```

Solution (Codes at B.9):

第一种方法是快排之后对比首位字符串得出结果。

考虑字符串快排复杂度涉及字符串间的比较，java 代码运行速度较慢，因此第二种方法不用排序，直接以第一个字符串为基础，在后续字符串中对比，若不是前缀，则长度减少，直至匹配。

A.10 Leetcode 15

Problem Description:

三数之和

给你一个包含 n 个整数的数组 nums，判断 nums 中是否存在三个元素 a,b,c，使得 $a+b+c=0$？请你找出所有满足条件且不重复的三元组。

注意：答案中不可以包含重复的三元组。

Sample:

input:

```
给定数组 nums = [-1, 0, 1, 2, -1, -4],
```

otput:

```
满足要求的三元组集合为:
[
  [-1, 0, 1],
  [-1, -1, 2]
]
```

Solution (Codes at B.10):

不要依赖 STL，不是所有题目都需要离散化处理。对于一般的链表问题，去重遍历就可以达到离散化的效果。先排序，指定一个数值的基础上再挑选符合条件的另外两个数值。贪心匹配两个数值的大小。

A.11 Leetcode 16

Problem Description:

最接近的三数之和

给定一个包括 n 个整数的数组 nums 和一个目标值 target。找出 nums 中的三个整数，使得它们的和与 target 最接近。返回这三个数的和。假定每组输入只存在唯一答案。

提示：

$$3 <= nums.length <= 10^3$$

$$-10^3 <= nums[i] <= 10^3$$

$$-10^4 <= target <= 10^4$$

Sample:

input:

```
输入：nums = [-1,2,1,-4], target = 1
```

otput:

```
输出：2
解释：与 target 最接近的和是 2 (-1 + 2 + 1 = 2) 。
```

Solution (Codes at B.11):

不要依赖 STL，不是所有题目都需要离散化处理。对于一般的链表问题，去重遍历就可以达到离散化的效果。先排序，指定一个数值的基础上再挑选符合条件的另外两个数值。贪心匹配两个数值的大小。

A.12 Leetcode 19

Problem Description:

Problem Description:

删除链表的倒数第 N 个节点

给定一个链表，删除链表的倒数第 n 个节点，并且返回链表的头结点。

Sample:

input:

```
给定一个链表: 1->2->3->4->5, 和 n = 2.
```

otput:

```
当删除了倒数第二个节点后，链表变为 1->2->3->5.
```

Solution (Codes at B.12):
递归记录个数。

A.13 Leetcode 25

Problem Description:
Problem Description:
K 个一组翻转链表
给你一个链表，每 k 个节点一组进行翻转，请你返回翻转后的链表。
k 是一个正整数，它的值小于或等于链表的长度。
如果节点总数不是 k 的整数倍，那么请将最后剩余的节点保持原有顺序。
说明:
你的算法只能使用常数的额外空间。
你不能只是单纯的改变节点内部的值，而是需要实际进行节点交换。
Sample:
input:

```
给你这个链表：1->2->3->4->5
```

otput:

```
当 k = 2 时，应当返回：2->1->4->3->5

当 k = 3 时，应当返回：3->2->1->4->5
```

Solution (Codes at B.13):
标记子链表首尾，化简为链表反转问题，递归解决。

A.14 Leetcode 26

Problem Description:
删除排序数组中的重复项
给定一个排序数组，你需要在原地删除重复出现的元素，使得每个元素只出现一次，返回移除后数组的新长度。
不要使用额外的数组空间，你必须在原地修改输入数组并在使用 $O(1)$ 额外空间的条件下完成。
说明:
为什么返回数值是整数，但输出的答案是数组呢?
请注意，输入数组是以引用方式传递的，这意味着在函数里修改输入数组对于调用者是可见的。
你可以想象内部操作如下:

```
输入：nums = [-1,2,1,-4], target = 1

// nums 是以“引用”方式传递的。也就是说，不对实参做任何拷贝
int len = removeDuplicates(nums);

// 在函数里修改输入数组对于调用者是可见的。
// 根据你的函数返回的长度，它会打印出数组中该长度范围内的所有元素。
for (int i = 0; i < len; i++) {
    print(nums[i]);
}
```

Sample:
input:

```
给定数组 nums = [1,1,2],

给定 nums = [0,0,1,1,1,2,2,3,3,4],
```

otput:

```
函数应该返回新的长度 2，并且原数组 nums 的前两个元素被修改为 1，2。
你不需要考虑数组中超出新长度后面的元素。

函数应该返回新的长度 5，并且原数组 nums 的前五个元素被修改为 0，1，2，3，4。
你不需要考虑数组中超出新长度后面的元素。
```

Solution (Codes at B.14):

原始数组已经排过序了。注意当数据量大的时候，判断条件越少越好。

A.15 Leetcode 42

Problem Description:

接雨水

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。

上面是由数组 $[0,1,0,2,1,0,1,3,2,1,2,1]$ 表示的高度图，在这种情况下，可以接 6 个单位的雨水（蓝色部分表示雨水）。

Sample:

input:

```
输入：[0,1,0,2,1,0,1,3,2,1,2,1]
```

otput:

```
输出：6
```

Solution (Codes at B.15):

左右打表记录最大值，查询之后得出结果。

老问题，注意当数据量大的时候，判断条件越少越好。以及，多用 C++11 的新初始化特性可以减少时间开销。

A.16 Leetcode 43

Problem Description:

字符串相乘

给定两个以字符串形式表示的非负整数 num1 和 num2，返回 num1 和 num2 的乘积，它们的乘积也表示为字符串形式。

说明：

1. num1 和 num2 的长度小于 110。
2. num1 和 num2 只包含数字 0-9。
3. num1 和 num2 均不以零开头，除非是数字 0 本身。
4. 不能使用任何标准库的大数类型（比如 BigInteger）或直接将输入转换为整数来处理。

Sample:

input:

```
输入：num1 = "2", num2 = "3"

输入：num1 = "123", num2 = "456"
```

otput:

```
输出："6"

输出："56088"
```

Solution (Codes at B.16):

大数相乘。两数相乘的结果位数不超过两数位数之和。根据乘数位数，可以直接推断出该数值在结果数值中的未知。

A.17 Leetcode 54

Problem Description:

螺旋矩阵

给你一个 m 行 n 列的矩阵 matrix ，请按照顺时针螺旋顺序，返回矩阵中的所有元素。

1. m == matrix.length
2. n == matrix[i].length
3. 1 <= m, n <= 10
4. -100 <= matrix[i][j] <= 100

Sample:

input:

```
输入：matrix = [[1,2,3],[4,5,6],[7,8,9]]

输入：matrix = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]
```

otput:

```
输出：[1,2,3,6,9,8,7,4,5]

输出：[1,2,3,4,8,12,11,10,9,5,6,7]
```

Solution (Codes at B.17):

没的说，照做呗。

A.18 Leetcode 56

Problem Description:

合并区间

给出一个区间的集合，请合并所有重叠的区间。。

Sample:

input:

```
输入: intervals = [[1,3],[2,6],[8,10],[15,18]]

输入: intervals = [[1,4],[4,5]]
```

otput:

```
输出: [[1,6],[8,10],[15,18]]
解释: 区间 [1,3] 和 [2,6] 重叠, 将它们合并为 [1,6].

输出: [[1,5]]
解释: 区间 [1,4] 和 [4,5] 可被视为重叠区间。
```

Solution (Codes at B.18):

自定义排序之后合并。需要注意对是 C++ 快排默认比较顺序是挨个从小到大，如果可以使用原始比较函数就不要自己重新写，还是用初始的更快。

注意 java 的自定义比较函数。

A.19 Leetcode 61

Problem Description:

旋转链表

给定一个链表，旋转链表，将链表每个节点向右移动 k 个位置，其中 k 是非负数。

Sample:

input:

```
输入: 1->2->3->4->5->NULL, k = 2

输入: 0->1->2->NULL, k = 4
```

otput:

```
输出: 4->5->1->2->3->NULL
解释:
向右旋转 1 步: 5->1->2->3->4->NULL
向右旋转 2 步: 4->5->1->2->3->NULL

输出: 2->0->1->NULL
解释:
向右旋转 1 步: 2->0->1->NULL
向右旋转 2 步: 1->2->0->NULL
向右旋转 3 步: 0->1->2->NULL
向右旋转 4 步: 2->0->1->NULL
```

Solution (Codes at B.19):

对 k 取模得到 mk，之后更改倒数第 mk 个 node 开始第子链表顺序到首位。

A.20 Leetcode 121

Problem Description:

买卖股票的最佳时机

给定一个数组，它的第 i 个元素是一支给定股票第 i 天的价格。

如果你最多只允许完成一笔交易（即买入和卖出一支股票一次），设计一个算法来计算你所能获取的最大利润。

注意：你不能在买入股票前卖出股票。

Sample:

input:

```
输入: [7,1,5,3,6,4]

输入: [7,6,4,3,1]
```

otput:

```
输出: 5
解释: 在第 2 天（股票价格 = 1）的时候买入，在第 5 天（股票价格 = 6）的时候卖出，最大利润 = 6-1 = 5 。
     注意利润不能是 7-1 = 6, 因为卖出价格需要大于买入价格；同时，你不能在买入前卖出股票。

输出: 0
解释: 在这种情况下, 没有交易完成, 所以最大利润为 0。
```

Solution (Codes at B.20):

从后向前打表记录最大值，查询之后得出结果。

A.21 Leetcode 138

Problem Description:

复制带随机指针的链表

给定一个链表，每个节点包含一个额外增加的随机指针，该指针可以指向链表中的任何节点或空节点。

要求返回这个链表的深拷贝。

我们用一个由 n 个节点组成的链表来表示输入/输出中的链表。每个节点用一个 $[val, random_index]$ 表示：

val：一个表示 Node.val 的整数。

$random_index$：随机指针指向的节点索引（范围从 0 到 n-1）；如果不指向任何节点，则为 null 。

$-10000 <= Node.val <= 10000$

Node.random 为空（null）或指向链表中的节点。

节点数目不超过 1000。

Sample:

input:

```
输入：head = [[7,null],[13,0],[11,4],[10,2],[1,0]]

输入：head = [[1,1],[2,1]]
输入：head = [[3,null],[3,0],[3,null]]
输入：head = []
```

otput:

```
输出：[[7,null],[13,0],[11,4],[10,2],[1,0]]
输出：[[1,1],[2,1]]
输出：[[3,null],[3,0],[3,null]]
输出：[]
解释：给定的链表为空（空指针），因此返回 null。
```

Solution (Codes at B.21):

map 存储节点 pair。

A.22 Leetcode 141

Problem Description:

环形链表

给定一个链表，判断链表中是否有环。

为了表示给定链表中的环，我们使用整数 pos 来表示链表尾连接到链表中的位置（索引从 0 开始）。如果 pos 是-1，则在该链表中没有环。

进阶：

你能用 $O(1)$（即，常量）内存解决此问题吗?

Sample:

input:

```
输入：head = [3,2,0,-4], pos = 1

输入：head = [1,2], pos = 0

输入：head = [1], pos = -1
```

otput:

```
输出：true
解释：链表中有一个环，其尾部连接到第二个节点。

输出：true
解释：链表中有一个环，其尾部连接到第一个节点。

输出：false
解释：链表中没有环。
```

Solution (Codes at B.22):

快慢指针循环跑圈相遇查重。

A.23 Leetcode 172

Problem Description:

阶乘后的零

给定一个整数 n，返回 n! 结果尾数中零的数量。

Sample:

input:

```
输入: 3

输入: 5
```

otput:

```
输出: 0
解释: 3! = 6, 尾数中没有零。

输出: 1
解释: 5! = 120, 尾数中有 1 个零.
```

Solution (Codes at B.23):

产生尾数 0 的原因是阶乘结果含有因子 10，因子 10 的个数等于因子 5 的个数（因子 2 的个数必然大于因子 5 的个数）。因此答案等于因子 5 的个数。

考虑 $n/5$（向下取整），其结果是数 $k <= n$ 中含有因子 5 的个数。推广，$n/5^m$ 就是计算 $k <= n$ 中含有因子 5^m 的个数。

因此连续对 n 向下取整的除以 5 就是逐步计算含有 5^m 因子的数的个数，也就是该因子的个数。

A.24 Leetcode 202

Problem Description:

快乐数

编写一个算法来判断一个数 n 是不是快乐数。

快乐数定义为：对于一个正整数，每一次将该数替换为它每个位置上的数字的平方和，然后重复这个过程直到这个数变为 1，也可能是无限循环但始终变不到 1。如果可以变为 1，那么这个数就是快乐数。

如果 n 是快乐数就返回 True ； 不是，则返回 False 。

Sample:

input:

```
输入: 19
```

otput:

```
输出: true
解释:
12 + 92 = 82
82 + 22 = 68
62 + 82 = 100
12 + 02 + 02 = 1
```

Solution (Codes at B.24):

快慢指针循环跑圈相遇查重。

A.25 Leetcode 206

Problem Description:

反转链表

反转一个单链表。

Sample:

input:

```
输入: 1->2->3->4->5->NULL
```

otput:

```
输出: 5->4->3->2->1->NULL
```

Solution (Codes at B.25):

递归或先找到首尾节点之后 while 循环更新 next。

A.26 Leetcode 209

Problem Description:

长度最小的子数组

给定一个含有 n 个正整数的数组和一个正整数 s，找出该数组中满足其和 $>= s$ 的长度最小的连续子数组，并返回其长度。如果不存在符合条件的子数组，返回 0。

进阶：

如果你已经完成了 $O(n)$ 时间复杂度的解法, 请尝试 $O(nlogn)$ 时间复杂度的解法。

Sample:

input:

```
输入: s = 7, nums = [2,3,1,2,4,3]
```

otput:

```
输出: 2
解释: 子数组 [4,3] 是该条件下的长度最小的子数组。
```

Solution (Codes at B.26):

遍历一遍，滑动窗口更新数值。

或者前缀和，二分搜索数值。

A.27 Leetcode 258

Problem Description:

各位相加

给定一个非负整数 num，反复将各个位上的数字相加，直到结果为一位数。

进阶：

你可以不使用循环或者递归，且在 O(1) 时间复杂度内解决这个问题吗?

Sample:

input:

```
输入: 38
```

otput:

```
输出: 2
解释: 各位相加的过程为: 3 + 8 = 11, 1 + 1 = 2。 由于 2 是一位数，所以返回 2。
```

Solution (Codes at B.27):

产生尾数 0 的原因是阶乘结果含有因子 10，因子 10 的个数等于因子 5 的个数（因子 2 的个数必然大于因子 5 的个数）。因此答案等于因子 5 的个数。

考虑 $n/5$ （向下取整），其结果是数 $k <= n$ 中含有因子 5 的个数。推广，$n/5^m$ 就是计算 $k <= n$ 中含有因子 5^m 的个数。

因此连续对 n 向下取整的除以 5 就是逐步计算含有 5^m 因子的数的个数，也就是该因子的个数。

A.28 Leetcode 387

Problem Description:

字符串中的第一个唯一字符

给定一个字符串，找到它的第一个不重复的字符，并返回它的索引。如果不存在，则返回 -1 。

Sample:

input:

```
leetcode
loveleetcode
```

otput:

```
0
2
```

Solution (Codes at B.28):

找到字符出现的首位和末位进行判断，然后取最早出现的。

A.29 Leetcode 763

Problem Description:

划分字母区间

字符串 S 由小写字母组成。我们要把这个字符串划分为尽可能多的片段，同一字母最多出现在一个片段中。返回一个表示每个字符串片段的长度的列表。

Sample:

input:

```
输入：S = "ababcbacadefegdehijhklij"
```

otput:

```
输出：[9,7,8]
解释：
划分结果为 "ababcbaca", "defegde", "hijhklij"。
每个字母最多出现在一个片段中。
像 "ababcbacadefegde", "hijhklij" 的划分是错误的，因为划分的片段数较少。
```

Solution (Codes at B.29):

相比于合并区间，该题可以只用每个字母的末尾位置即可。

A.30 Leetcode 876

Problem Description:

链表的中间结点

给定一个带有头结点 head 的非空单链表，返回链表的中间结点。

如果有两个中间结点，则返回第二个中间结点。

Sample:

input:

```
输入：[1,2,3,4,5]

输入：[1,2,3,4,5,6]
```

otput:

```
输出：此列表中的结点 3 (序列化形式：[3,4,5])
返回的结点值为 3 。 (测评系统对该结点序列化表述是 [3,4,5])。
注意，我们返回了一个 ListNode 类型的对象 ans，这样：
ans.val = 3, ans.next.val = 4, ans.next.next.val = 5, 以及 ans.next.next.next = NULL.

输出：此列表中的结点 4 (序列化形式：[4,5,6])
由于该列表有两个中间结点，值分别为 3 和 4，我们返回第二个结点。
```

Solution (Codes at B.30):

快慢指针计数。

附录 B Code List

B.1 Leetcode 1

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  vector<int> twoSum(vector<int>& nums, int target) {
    vector<int> ans;
    for (int i = 0; i < nums.size(); i++)
      for (int j = i + 1; j < nums.size(); j++)
        if (nums[i] + nums[j] == target) {
          // u = i;
          // v = j;
          ans.push_back(i);
          ans.push_back(j);
          break;
        }
    return ans;
  }
  void input(void) {
    while (~scanf("%d %d", &n, &m))
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        numbers.push_back(t);
      }
  }
  void solve(void) { twoSum(numbers, m); }
  void otput(void) { printf("%d %d\n", u, v); }

 private:
  int n, m, t;
  int u, v;
  vector<int> numbers;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();
  sol.solve();
  sol.otput();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int[] twoSum(int[] nums, int target) {
    int[] ans = new int[2];
    for (int i = 0; i < nums.length; i++)
```

```java
      for (int j = i + 1; j < nums.length; j++)
        if (nums[i] + nums[j] == target) {
          u = i;
          v = j;
          ans[0] = i;
          ans[1] = j;
          break;
        }
    return ans;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
    sol.solve();
    sol.otput();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();
      for (int i = 0; i < n; i++)
        numbers[i] = in.nextInt();
    }

    in.close();
  }

  public void solve() {
    twoSum(numbers, m);
  }

  public void otput() {
    System.out.println(u + " " + v);
  }

  private int n, m, t;
  private int u, v;
  private int[] numbers = new int[10000];
}
```

B.2 Leetcode 2

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

```

```
typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
    ListNode* res = new ListNode(0);
    ListNode* u = l1;
    ListNode* v = l2;
    ListNode* cur = NULL;
    int t = 0;

    while (u != NULL || v != NULL || t != 0) {
      if (cur == NULL) {
        cur = res;
      } else {
        cur->next = new ListNode(0);
        cur = cur->next;
      }
      cur->next = NULL;

      cur->val = t;
      if (u != NULL) {
        cur->val += u->val;
        u = u->next;
      }
      if (v != NULL) {
        cur->val += v->val;
        v = v->next;
      }
      t = cur->val / 10;
      cur->val %= 10;
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      b = new ListNode(0);
      ListNode* u = a;
      ListNode* v = b;

      for (int i = 0; i < n; i++) {
        u->next = new ListNode(0);
        u = u->next;
        u->next = NULL;

        scanf("%d", &u->val);
      }
      for (int i = 0; i < m; i++) {
        v->next = new ListNode(0);
        v = v->next;
        v->next = NULL;

        scanf("%d", &v->val);
      }

      solve(a, b);
    }
  }
  void show(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
```

```
    }
  }
  void solve(ListNode* x, ListNode* y) {
    ListNode* res;
    res = addTwoNumbers(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x->next;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
    cout << endl;
  }

 private:
  int n, m;
  ListNode *a, *b;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode addTwoNumbers(ListNode l1, ListNode l2) {
    ListNode res = new ListNode(0);
    ListNode u = l1;
    ListNode v = l2;
    ListNode cur = null;
    int t = 0;

    while (u != null || v != null || t != 0) {
      if (cur == null) {
        cur = res;
      } else {
        cur.next = new ListNode(0);
        cur = cur.next;
      }
      cur.next = null;

      cur.val = t;
      if (u != null) {
        cur.val += u.val;
```

```
      u = u.next;
    }
    if (v != null) {
      cur.val += v.val;
      v = v.next;
    }
    t = cur.val / 10;
    cur.val %= 10;
  }

  return res;
}

public static void main(String[] args) throws FileNotFoundException {
  FileInputStream fin = new FileInputStream("./assets/fipt.txt");
  PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

  System.setIn(fin);
  System.setOut(fot);

  Solution sol = new Solution();

  sol.input();
}

public void input() {
  Scanner in = new Scanner(System.in);
  while (in.hasNext()) {
    n = in.nextInt();
    m = in.nextInt();

    ListNode a = new ListNode(0);
    ListNode b = new ListNode(0);
    ListNode u = a;
    ListNode v = b;

    for (int i = 0; i < n; i++) {
      if (i != 0) {
        u.next = new ListNode(0);
        u = u.next;
      }
      u.val = in.nextInt();
    }
    for (int i = 0; i < m; i++) {
      if (i != 0) {
        v.next = new ListNode(0);
        v = v.next;
      }
      v.val = in.nextInt();
    }

    solve(a, b);
  }

  in.close();
}

public void solve(ListNode x, ListNode y) {
  ListNode res;
  res = addTwoNumbers(x, y);
  otput(res);
}

public void otput(ListNode x) {
  while (x != null) {
    System.out.print(x.val);
    x = x.next;
  }
  System.out.println("");
```

```
  }

  private int n, m;
}
```

B.3 Leetcode 3

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int lengthOfLongestSubstring(string s) {
    int res = 0;
    int sz = s.length();
    int l = 0, r = 0;
    if (sz == 0) return 0;
    int mp[256];
    memset(mp, -1, 256 * sizeof(int));

    for (int i = 0; i < sz; i++)
      if (mp[s[i]] == -1) {
        r = i;
        mp[s[i]] = i;
      } else {
        if (mp[s[i]] == -2) {
          r = i;
          mp[s[i]] = i;
        } else {
          res = max(res, r - l + 1);
          int newL = mp[s[i]] + 1;
          for (int j = l; j < mp[s[i]] + 1; j++) mp[s[j]] = -2;
          l = newL;
          r = i;
          mp[s[i]] = i;
        }
      }

    res = max(res, r - l + 1);
    return res;
  }
  void input(void) {
    while (cin >> a) {
      solve(a);
    }
  }
  void solve(string x) {
    int res;
    res = lengthOfLongestSubstring(x);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  string a;
```

```
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Arrays;
import java.util.Scanner;

class Solution {
  public int lengthOfLongestSubstring(String s) {
    int res = 0;
    int sz = s.length();
    int l = 0, r = 0;
    if (sz == 0)
      return 0;
    int[] mp = new int[256];
    Arrays.fill(mp, -1);

    for (int i = 0; i < sz; i++)
      if (mp[s.charAt(i)] == -1) {
        r = i;
        mp[s.charAt(i)] = i;
      } else {
        if (mp[s.charAt(i)] == -2) {
          r = i;
          mp[s.charAt(i)] = i;
        } else {
          res = Math.max(res, r - l + 1);
          int newL = mp[s.charAt(i)] + 1;
          for (int j = l; j < mp[s.charAt(i)] + 1; j++)
            mp[s.charAt(j)] = -2;
          l = newL;
          r = i;
          mp[s.charAt(i)] = i;
        }
      }

    res = Math.max(res, r - l + 1);
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
```

```
    while (in.hasNext()) {
      solve(in.nextLine());
    }

    in.close();
  }

  public void solve(String x) {
    int res;
    res = lengthOfLongestSubstring(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private String a;
}
```

B.4 Leetcode 6

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  string convert(string s, int numRows) {
    if (s.length() <= 1) return s;
    if (numRows == 1) return s;
    string res = "";
    vector<int> v;
    int addItem = 2 * (numRows - 1);
    int sz = s.length();
    int item = 0;
    int len;
    int l, r;
    while (item < sz) {
      v.push_back(item);
      res += s[item];
      item += addItem;
    }
    v.push_back(item);
    for (len = 1; len < numRows - 1; len++) {
      for (int i = 0; i < v.size(); i++) {
        l = v[i] - len;
        r = v[i] + len;
        if (0 <= l && l < sz) {
          res += s[l];
        }
        if (0 <= r && r < sz) {
          res += s[r];
        }
      }
```

```cpp
    }
    for (int i = 0; i < v.size(); i++) {
      r = v[i] + numRows - 1;
      if (r < sz) {
        res += s[r];
      }
    }
    return res;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      cin >> a;

      solve(a, n);
    }
  }

  void solve(string x, int y) {
    string res;
    res = convert(x, y);
    otput(res);
  }
  void otput(string x) { cout << x << endl; }

 private:
  int n;
  string a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.Scanner;

class Solution {
  public String convert(String s, int numRows) {
    if (s.length() <= 1)
      return s;
    if (numRows == 1)
      return s;
    String res = "";
    ArrayList<Integer> v = new ArrayList<Integer>();
    int addItem = 2 * (numRows - 1);
    int sz = s.length();
    int item = 0;
    int len;
    int l, r;
    while (item < sz) {
      v.add(item);

      res += s.charAt(item);
      item += addItem;
    }
    v.add(item);
```

```
    for (len = 1; len < numRows - 1; len++) {
      for (int i = 0; i < v.size(); i++) {
        l = v.get(i) - len;
        r = v.get(i) + len;
        if (0 <= l && l < sz) {
          res += s.charAt(l);
        }
        if (0 <= r && r < sz) {
          res += s.charAt(r);
        }
      }
    }
    for (int i = 0; i < v.size(); i++) {
      r = v.get(i) + numRows - 1;
      if (r < sz) {
        res += s.charAt(r);
      }
    }
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      a = in.next();

      solve(a, n);
    }

    in.close();
  }

  public void solve(String x, int y) {
    String res;
    res = convert(x, y);
    otput(res);

  }

  public void otput(String x) {
    System.out.println(x);
  }

  private int n;
  private String a;
}
```

B.5 Leetcode 7

C++

```
#include <math.h>

```

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int reverse(int x) {
    int res = 0;
    int l = INT_MIN / 10;
    int r = INT_MAX / 10;
    while (x / 10) {
      res = res * 10 + x % 10;
      x /= 10;
    }

    if (res < l || res > r) {
      res = 0;
    } else if (res == l && x < -7) {
      res = 0;
    } else if (res == r && x > 8) {
      res = 0;
    } else {
      res = res * 10 + x;
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
        solve(a);
      }
    }
  }

  void solve(int x) {
    int res;
    res = reverse(x);
    otput(res);
  }

  void otput(int x) { cout << x << endl; }

 private:
  int n;
  int a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int reverse(int x) {
    int res = 0;
    int l = Integer.MIN_VALUE / 10;
    int r = Integer.MAX_VALUE / 10;
    while (0 != (x / 10)) {
      res = res * 10 + x % 10;
      x /= 10;
    }

    if (res < l || res > r) {
      res = 0;
    } else if (res == l && x < -7) {
      res = 0;
    } else if (res == r && x > 8) {
      res = 0;
    } else {
      res = res * 10 + x;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      for (int i = 0; i < n; i++) {
        a = in.nextInt();
        solve(a);
      }
    }

    in.close();
  }

  public void solve(int x) {
    int res;
    res = reverse(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
```

```
68    private int a;
69  }
```

B.6 Leetcode 8

C++

```cpp
#include <math.h>

#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int myAtoi(string s) {
    // 2147483647
    // -2147483648
    int res = 0;
    int l = INT_MIN / 10;
    int r = INT_MAX / 10;
    int sz = s.size();
    int trimmingLeft = 1;
    int tmp = 0;
    int op = 1;
    for (int i = 0; i < sz; i++) {
      if ('0' <= s[i] && s[i] <= '9') {
        trimmingLeft = 0;
        tmp = s[i] - '0';
        if (op > 0) {
          if (res > r) {
            return INT_MAX;
          } else if (res == r && tmp > 7) {
            return INT_MAX;
          } else {
            res = res * 10 + tmp * op;
          }
        } else {
          if (res < l) {
            return INT_MIN;
          } else if (res == l && tmp > 8) {
            return INT_MIN;
          } else {
            res = res * 10 + tmp * op;
          }
        }
      } else if (s[i] == ' ') {
        if (trimmingLeft == 0) {
          break;
        }
      } else if (s[i] == '+') {
        if (trimmingLeft == 0) {
          break;
        }
        trimmingLeft = 0;
        op = 1;
      } else if (s[i] == '-') {
        if (trimmingLeft == 0) {
```

```
          break;
        }
        trimmingLeft = 0;
        op = -1;
      } else {
        break;
      }
    }
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      getline(cin, a);
      for (int i = 0; i < n; i++) {
        // cin >> a;
        getline(cin, a);
        solve(a);
      }
    }
  }

  void solve(string x) {
    int res;
    res = myAtoi(x);
    otput(res);
  }

  void otput(int x) { cout << x << endl; }

 private:
  int n;
  string a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int myAtoi(String s) {
    // 2147483647
    // -2147483648
    int res = 0;
    int l = Integer.MIN_VALUE / 10;
    int r = Integer.MAX_VALUE / 10;
    int sz = s.length();
    int trimmingLeft = 1;
    int tmp = 0;
    int op = 1;
    for (int i = 0; i < sz; i++) {
      if ('0' <= s.charAt(i) && s.charAt(i) <= '9') {
        trimmingLeft = 0;
        tmp = s.charAt(i) - '0';
```

```
        if (op > 0) {
          if (res > r) {
            return Integer.MAX_VALUE;
          } else if (res == r && tmp > 7) {
            return Integer.MAX_VALUE;
          } else {
            res = res * 10 + tmp * op;
          }
        } else {
          if (res < l) {
            return Integer.MIN_VALUE;
          } else if (res == l && tmp > 8) {
            return Integer.MIN_VALUE;
          } else {
            res = res * 10 + tmp * op;
          }
        }
      } else if (s.charAt(i) == ' ') {
        if (trimmingLeft == 0) {
          break;
        }
      } else if (s.charAt(i) == '+') {
        if (trimmingLeft == 0) {
          break;
        }
        trimmingLeft = 0;
        op = 1;
      } else if (s.charAt(i) == '-') {
        if (trimmingLeft == 0) {
          break;
        }
        trimmingLeft = 0;
        op = -1;
      } else {
        break;
      }
    }
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = in.nextLine();
      for (int i = 0; i < n; i++) {
        a = in.nextLine();
        solve(a);
      }
    }

    in.close();
  }

  public void solve(String x) {
    int res;
    res = myAtoi(x);
```

```
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private String a;
}
```

B.7 Leetcode 9

C++

```cpp
#include <math.h>

#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  bool isPalindrome(int x) {
    if (x < 0) {
      return false;
    }
    int res = x;
    int num = 0;
    int r = INT_MAX / 10;
    while (x) {
      if (num > r) {
        return false;
      } else if (num == r && (x % 10) > 7) {
        return false;
      }
      num = num * 10 + x % 10;
      x /= 10;
    }

    return res == num;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
        solve(a);
      }
    }
  }

  void solve(int x) {
    bool res;
    res = isPalindrome(x);
    otput(res);
  }
```

```cpp

  void otput(bool x) { cout << x << endl; }

 private:
  int n;
  int a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public boolean isPalindrome(int x) {
    if (x < 0) {
      return false;
    }
    int res = x;
    int num = 0;
    int r = Integer.MAX_VALUE / 10;
    while (x > 0) {
      if (num > r) {
        return false;
      } else if (num == r && (x % 10) > 7) {
        return false;
      }
      num = num * 10 + x % 10;
      x /= 10;
    }

    return res == num;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      for (int i = 0; i < n; i++) {
        a = in.nextInt();
        solve(a);
      }
    }
```

```
49
50      in.close();
51    }
52
53    public void solve(int x) {
54      boolean res;
55      res = isPalindrome(x);
56      otput(res);
57
58    }
59
60    public void otput(Boolean x) {
61      System.out.println(x);
62    }
63
64    private int n;
65    private int a;
66  }
```

B.8 Leetcode 11

C++

```
1   #include <algorithm>
2   #include <cstdio>
3   #include <cstdlib>
4   #include <cstring>
5   #include <iostream>
6   #include <map>
7   #include <queue>
8   #include <stack>
9   #include <string>
10  #include <vector>
11
12  using namespace std;
13
14  class Solution {
15   public:
16    int maxArea(vector<int>& height) {
17      int res = 0;
18      int sz = height.size();
19      int l = 0, r = sz - 1;
20      while (l < r) {
21        res = max(res, min(height[l], height[r]) * (r - l));
22        if (height[l] < height[r])
23          l++;
24        else
25          r--;
26      }
27
28      return res;
29    }
30
31    void input(void) {
32      while (~scanf("%d", &n)) {
33        int t;
34        for (int i = 0; i < n; i++) {
35          scanf("%d", &t);
36          a.push_back(t);
37        }
38        solve(a);
39      }
40    }
41    void solve(vector<int>& x) {
42      int res;
43      res = maxArea(x);
44      otput(res);
```

```
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int maxArea(int[] height) {
    int res = 0;
    int sz = height.length;
    int l = 0, r = sz - 1;
    while (l < r) {
      res = Math.max(res, Math.min(height[l], height[r]) * (r - l));
      if (height[l] < height[r])
        l++;
      else
        r--;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new int[n];
      for (int i = 0; i < n; i++)
        a[i] = in.nextInt();
      solve(a);
    }

    in.close();
  }

  public void solve(int[] x) {
```

```
    int res;
    res = maxArea(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private int[] a;
}
```

B.9 Leetcode 14

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  string longestCommonPrefix(vector<string>& strs) {
    if (strs.size() == 0) {
      return "";
    }
    string str = strs[0];
    for (int i = 1; i < strs.size(); i++) {
      while (strs[i].find(str) != 0) {
        str = str.substr(0, str.length() - 1);
      }
      if (str.size() == 0) {
        break;
      }
    }
    return str;
  }
  // string longestCommonPrefix(vector<string>& strs) {
  //   string res = "";
  //   int sz = strs.size();

  //   if (sz != 0) {
  //     sort(strs.begin(), strs.end());
  //     int cnt = strs.front().length();
  //     if (cnt != 0) {
  //       if (sz == 1) {
  //         res = strs.front();
  //       } else {
  //         for (int j = 0; j < cnt; j++) {
  //           int i = sz - 1;
  //           if (strs[i][j] != strs.front()[j]) {
  //             return res;
  //           }
  //           res += strs.front()[j];
  //         }
  //       }
```

```
  //     }
  //   }
  //   return res;
  // }
  void input(void) {
    while (~scanf("%d", &n)) {
      a.clear();
      string str;
      for (int i = 0; i < n; i++) {
        cin >> str;
        a.push_back(str);
      }

      solve(a);
    }
  }

  void solve(vector<string>& x) {
    string res;
    res = longestCommonPrefix(x);
    otput(res);
  }
  void otput(string x) { cout << x << endl; }

 private:
  int n;
  vector<string> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public String longestCommonPrefix(String[] strs) {
    if (strs.length == 0) {
      return "";
    }
    String str = strs[0];
    for (int i = 1; i < strs.length; i++) {
      while (strs[i].indexOf(str) != 0) {
        str = str.substring(0, str.length() - 1);
      }
      if (str.length() == 0) {
        break;
      }
    }
    return str;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

```

```
    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new String[n];

      for (int i = 0; i < n; i++) {
        a[i] = in.next();
      }

      solve(a);
    }

    in.close();
  }

  public void solve(String[] x) {
    String res;
    res = longestCommonPrefix(x);
    otput(res);

  }

  public void otput(String x) {
    System.out.println(x);
  }

  private int n;
  private String[] a = null;
}
```

B.10 Leetcode 15

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  vector<vector<int> > threeSum(vector<int>& nums) {
    vector<vector<int> > res;
    int sz = nums.size();
    int i, j, k;
    int r;
    if (sz < 3) return res;
    sort(nums.begin(), nums.end());

    for (i = 0; i < sz; i++) {
```

```
      if (nums[i] > 0) break;
      if (i > 0 && nums[i] == nums[i - 1]) continue;
      j = i + 1;
      k = sz - 1;
      while (j < k) {
        int tmp = nums[i] + nums[j] + nums[k];
        if (tmp < 0) {
          j++;
        } else if (tmp > 0) {
          k--;
        } else {
          res.push_back({nums[i], nums[j], nums[k]});
          j++;
          k--;
          while (j < k && nums[j] == nums[j - 1]) j++;
          while (j < k && nums[k] == nums[k + 1]) k--;
        }
      }
    }

    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      int t;
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        a.push_back(t);
      }
      solve(a);
      a.clear();
    }
  }
  void solve(vector<int>& x) {
    vector<vector<int> > res;
    res = threeSum(x);
    otput(res);
  }
  void otput(vector<vector<int> > x) {
    for (int i = 0; i < x.size(); i++)
      printf("%d %d %d\n", x[i][0], x[i][1], x[i][2]);
    puts("");
  }

 private:
  int n;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.Arrays;
```

```java
import java.util.List;
import java.util.Scanner;

class Solution {
  public List<List<Integer>> threeSum(int[] nums) {
    List<List<Integer>> res = new ArrayList<>();
    Arrays.sort(nums);
    int sz = nums.length;
    for (int i = 0; i < sz; i++) {
      if (nums[i] > 0) {
        break;
      }
      if (i > 0 && nums[i - 1] == nums[i]) {
        continue;
      }
      int j = i + 1, k = sz - 1;
      while (j < k) {
        int t = nums[i] + nums[j] + nums[k];
        if (t > 0) {
          k--;
        } else if (t < 0) {
          j++;
        } else {
          res.add(Arrays.asList(nums[i], nums[j], nums[k]));
          j++;
          k--;
          while (j < k && nums[j - 1] == nums[j]) {
            j++;
          }
          while (j < k && nums[k] == nums[k + 1]) {
            k--;
          }
        }
      }
    }
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new int[n];
      for (int i = 0; i < n; i++)
        a[i] = in.nextInt();
      solve(a);
    }

    in.close();
  }

  public void solve(int[] x) {
    List<List<Integer>> res;
    res = threeSum(x);
    otput(res);

  }
```

```java

  public void otput(List<List<Integer>> x) {
    System.out.println(x);
  }

  private int n;
  private int[] a;
}
```

B.11 Leetcode 16

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int threeSumClosest(vector<int>& nums, int target) {
    int res = nums[0] + nums[1] + nums[2];
    sort(nums.begin(), nums.end());
    int sz = nums.size();
    for (int i = 0; i < sz; i++) {
      if (i != 0 && nums[i - 1] == nums[i]) {
        continue;
      }
      int j = i + 1, k = sz - 1;
      while (j < k) {
        int t = nums[i] + nums[j] + nums[k];
        if (t < target) {
          if (abs(target - t) < abs(target - res)) res = t;
          j++;
          while (j < k && nums[j - 1] == nums[j]) j++;
        } else if (t > target) {
          if (abs(target - t) < abs(target - res)) res = t;
          k--;
          while (j < k && nums[k] == nums[k + 1]) k--;
        } else {
          return target;
        }
      }
    }
    return res;
  }

  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      int t;
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        a.push_back(t);
      }
      solve(a, m);
      a.clear();
    }
  }
  void solve(vector<int>& x, int y) {
```

```
    int res;
    res = threeSumClosest(x, y);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n, m;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Arrays;
import java.util.Scanner;

class Solution {
  public int threeSumClosest(int[] nums, int target) {
    int res = nums[0] + nums[1] + nums[2];
    Arrays.sort(nums);
    int sz = nums.length;
    for (int i = 0; i < sz; i++) {
      if (i != 0 && nums[i - 1] == nums[i]) {
        continue;
      }
      int j = i + 1, k = sz - 1;
      while (j < k) {
        int t = nums[i] + nums[j] + nums[k];
        if (t < target) {
          if (Math.abs(target - t) < Math.abs(target - res))
            res = t;
          j++;
          while (j < k && nums[j - 1] == nums[j])
            j++;
        } else if (t > target) {
          if (Math.abs(target - t) < Math.abs(target - res))
            res = t;
          k--;
          while (j < k && nums[k] == nums[k + 1])
            k--;
        } else {
          return target;
        }
      }
    }
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);
```

```
46
47      Solution sol = new Solution();
48
49      sol.input();
50    }
51
52    public void input() {
53      Scanner in = new Scanner(System.in);
54      while (in.hasNext()) {
55        n = in.nextInt();
56        a = new int[n];
57        for (int i = 0; i < n; i++)
58          a[i] = in.nextInt();
59        solve(a);
60      }
61
62      in.close();
63    }
64
65    public void solve(int[] x, int y) {
66      int res;
67      res = threeSumClosest(x, y);
68      otput(res);
69
70    }
71
72    public void otput(int x) {
73      System.out.println(x);
74    }
75
76    private int n;
77    private int[] a;
78  }
```

B.12 Leetcode 19

C++

```
1   #include <algorithm>
2   #include <cstdio>
3   #include <cstdlib>
4   #include <cstring>
5   #include <iostream>
6   #include <map>
7   #include <queue>
8   #include <stack>
9   #include <string>
10  #include <vector>
11
12  using namespace std;
13
14  typedef struct ListNode {
15    int val;
16    ListNode* next;
17    ListNode(int x) : val(x), next(NULL) {}
18  } ListNode;
19
20  class Solution {
21   public:
22    int deleteNode(ListNode* x, int target) {
23      if (x == NULL) return 0;
24
25      int u = deleteNode(x->next, target);
26
27      if (u != -1) {
28        if (u == target) {
29          ListNode* y = x->next->next;
```

```
        delete x->next;
        x->next = y;
      } else {
        return u + 1;
      }
    }

    return -1;
  }
  ListNode* removeNthFromEnd(ListNode* head, int n) {
    int u = deleteNode(head, n);
    if (u != -1) {
      ListNode* t = head;
      head = head->next;
      delete t;
    }

    return head;
  }

  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      ListNode* u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a, n);
    }
  }
  void solve(ListNode* x, int y) {
    ListNode* res;
    res = removeNthFromEnd(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
  }

 private:
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
```

```java
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public int deleteNode(ListNode x, int target) {
    if (x == null)
      return 0;

    int u = deleteNode(x.next, target);

    if (u != -1) {
      if (u == target) {
        ListNode y = x.next.next;
        x.next = y;
      } else {
        return u + 1;
      }
    }

    return -1;
  }

  public ListNode removeNthFromEnd(ListNode head, int n) {
    int u = deleteNode(head, n);
    if (u != -1) {
      head = head.next;
    }
    return head;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }
```

```java

      solve(a, n);
    }

    in.close();
  }

  public void solve(ListNode x, int y) {
    ListNode res;
    res = removeNthFromEnd(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

B.13 Leetcode 25

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  void reverseGroup(ListNode* u, ListNode* v) {
    if (u != v) {
      if (u->next == v) {
        v->next = u;
      } else {
        ListNode* t = u->next;
        reverseGroup(u->next, v);
        t->next = u;
      }
    }
  }
  ListNode* reverseKGroup(ListNode* head, int k) {
    if (k == 1) return head;

    ListNode* fakeHead = new ListNode(0);
    fakeHead->next = head;
    ListNode* pre = fakeHead;
    ListNode* cur = head;
```

```
    ListNode* u = NULL;
    ListNode* t = NULL;
    ListNode* v = NULL;
    int cnt = 0;
    while (cur != NULL) {
      cnt++;
      if (cnt == k) {
        u = pre->next;
        v = cur;
        t = cur->next;
        reverseGroup(u, v);
        pre->next = v;
        u->next = t;
        pre = u;
        cur = u;
        cnt = 0;
      }
      cur = cur->next;
    }
    return fakeHead->next;
  }
  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      ListNode* u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a, n);
    }
  }
  void solve(ListNode* x, int y) {
    ListNode* res;
    res = reverseKGroup(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
    cout << endl;
  }

 private:
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  void reverseGroup(ListNode u, ListNode v) {
    if (u != v) {
      if (u.next == v) {
        v.next = u;
      } else {
        ListNode t = u.next;
        reverseGroup(u.next, v);
        t.next = u;
      }
    }
  }

  public ListNode reverseKGroup(ListNode head, int k) {
    if (k == 1)
      return head;
    ListNode fakeHead = new ListNode(0);
    fakeHead.next = head;
    ListNode pre = fakeHead;
    ListNode cur = head;
    ListNode u = null;
    ListNode t = null;
    ListNode v = null;
    int cnt = 0;
    while (cur != null) {
      cnt++;
      if (cnt == k) {
        u = pre.next;
        v = cur;
        t = cur.next;
        reverseGroup(u, v);
        pre.next = v;
        u.next = t;
        pre = u;
        cur = u;
        cnt = 0;
      }
      cur = cur.next;
    }
    return fakeHead.next;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }
```

```java

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a, n);
    }

    in.close();
  }

  public void solve(ListNode x, int y) {
    ListNode res;
    res = reverseKGroup(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

B.14 Leetcode 26

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int removeDuplicates(vector<int>& nums) {
    int res = 0;
    int sz = nums.size();
    if (sz == 0) return 0;
    int cur = 0;
    for (int i = 0; i < sz; i++)
      if (nums[cur] != nums[i]) {
```

```
        cur++;
        nums[cur] = nums[i];
      }
    res = cur + 1;
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      int t;
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        a.push_back(t);
      }
      solve(a);
      a.clear();
    }
  }
  void solve(vector<int>& x) {
    int res;
    res = removeDuplicates(x);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int removeDuplicates(int[] nums) {
    int res = 0;
    int sz = nums.length;
    if (sz == 0)
      return 0;
    int cur = 0;
    for (int i = 0; i < sz; i++)
      if (nums[cur] != nums[i]) {
        cur++;
        nums[cur] = nums[i];
      }
    res = cur + 1;
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

```

```
    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new int[n];
      for (int i = 0; i < n; i++)
        a[i] = in.nextInt();
      solve(a);
    }

    in.close();
  }

  public void solve(int[] x) {
    int res;
    res = removeDuplicates(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private int[] a;
}
```

B.15 Leetcode 42

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int trap(vector<int>& height) {
    int res = 0;
    int sz = height.size();
    if (sz == 0) return 0;

    vector<int> lmx(sz, 0), rmx(sz, 0);

    for (int i = 1, j = sz - 2; i < sz; i++, j--) {
      lmx[i] = max(lmx[i - 1], height[i - 1]);
      rmx[j] = max(rmx[j + 1], height[j + 1]);
    }
    for (int i = 1; i < sz - 1; i++)
```

```
      res += max(0, min(lmx[i], rmx[i]) - height[i]);
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      int t;
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        a.push_back(t);
      }
      solve(a);
      a.clear();
    }
  }
  void solve(vector<int>& x) {
    int res;
    res = trap(x);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int trap(int[] height) {
    int res = 0;
    int sz = height.length;
    if (sz == 0)
      return 0;

    int[] lmx = new int[sz];
    int[] rmx = new int[sz];

    for (int i = 1, j = sz - 2; i < sz; i++, j--) {
      lmx[i] = Math.max(lmx[i - 1], height[i - 1]);
      rmx[j] = Math.max(rmx[j + 1], height[j + 1]);
    }
    for (int i = 1; i < sz - 1; i++)
      res += Math.max(0, Math.min(lmx[i], rmx[i]) - height[i]);
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

```

```
    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new int[n];
      for (int i = 0; i < n; i++)
        a[i] = in.nextInt();
      solve(a);
    }

    in.close();
  }

  public void solve(int[] x) {
    int res;
    res = trap(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private int[] a;
}
```

B.16 Leetcode 43

C++

```
#include <math.h>

#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  string multiply(string num1, string num2) {
    string res = "";
    int len1 = num1.size();
    int len2 = num2.size();
    int a[230] = {0};
    int val;

    for (int i = len1 - 1; i >= 0; i--) {
      for (int j = len2 - 1; j >= 0; j--) {
        val = (num1[i] - '0') * (num2[j] - '0');
```

```
        int k = (len1 - i - 1) + (len2 - j - 1);
        a[k] += val;
        a[k + 1] += a[k] / 10;
        a[k] %= 10;
      }
    }

    for (int i = len1 + len2; i >= 0; i--) {
      if (res.size() == 0 && a[i] == 0) {
        continue;
      } else {
        res += '0' + a[i];
      }
    }

    if (res.size() == 0) {
      res = "0";
    }
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
        cin >> b;
        solve(a, b);
      }
    }
  }

  void solve(string x, string y) {
    string res;
    res = multiply(x, y);
    otput(res);
  }

  void otput(string x) { cout << x << endl; }

 private:
  int n;
  string a;
  string b;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public String multiply(String num1, String num2) {
    StringBuilder res = new StringBuilder();
    int len1 = num1.length();
    int len2 = num2.length();
```

```
    int a[] = new int[230];
    int val;

    for (int i = len1 - 1; i >= 0; i--) {
      for (int j = len2 - 1; j >= 0; j--) {
        val = (num1.charAt(i) - '0') * (num2.charAt(j) - '0');
        int k = (len1 - i - 1) + (len2 - j - 1);
        a[k] += val;
        a[k + 1] += a[k] / 10;
        a[k] %= 10;
      }
    }

    for (int i = len1 + len2; i >= 0; i--) {
      if (res.length() == 0 && a[i] == 0) {
        continue;
      } else {
        res.append(a[i]);
      }
    }

    if (res.length() == 0) {
      return "0";
    }
    return res.toString();
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      for (int i = 0; i < n; i++) {
        a = in.next();
        b = in.next();
        solve(a, b);
      }
    }

    in.close();
  }

  public void solve(String x, String y) {
    String res;
    res = multiply(x, y);
    otput(res);

  }

  public void otput(String x) {
    System.out.println(x);
  }

  private int n;
  private String a;
  private String b;
}
```

B.17 Leetcode 54

C++

```cpp
#include <math.h>

#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  vector<int> spiralOrder(vector<vector<int>>& matrix) {
    vector<int> res;
    int m = matrix.size();
    int n = matrix[0].size();

    int i = 0, j = 0;
    int row = m, col = n;

    int sum;

    int total = m * n;
    while (total) {
      if (row == 1) {
        for (int k = 0; k < col; k++) {
          res.push_back(matrix[i][j + k]);
        }
        sum = col;
      } else if (col == 1) {
        for (int k = 0; k < row; k++) {
          res.push_back(matrix[i + k][j + col - 1]);
        }
        sum = row;
      } else {
        for (int k = 0; k < col - 1; k++) {
          res.push_back(matrix[i][j + k]);
        }
        for (int k = 0; k < row - 1; k++) {
          res.push_back(matrix[i + k][j + col - 1]);
        }
        for (int k = 0; k < col - 1; k++) {
          res.push_back(matrix[i + row - 1][j + col - 1 - k]);
        }
        for (int k = 0; k < row - 1; k++) {
          res.push_back(matrix[i + row - 1 - k][j]);
        }
        sum = 2 * (row + col) - 4;
      }

      total -= sum;

      row -= 2;
      col -= 2;

      i++;
      j++;
    }

    return res;
```

```
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
        cin >> b;
        int v;
        vector<vector<int>> data;

        for (int j = 0; j < a; j++) {
          vector<int> item;
          for (int k = 0; k < b; k++) {
            cin >> v;
            // cout << v << endl;
            item.push_back(v);
          }
          data.push_back(item);
        }
        solve(data);
      }
    }
  }

  void solve(vector<vector<int>>& x) {
    vector<int> res;
    res = spiralOrder(x);
    otput(res);
  }

  void otput(vector<int> x) {
    for (int i = 0; i < x.size(); i++) {
      cout << x[i] << endl;
    }
    cout << endl;
  }

 private:
  int n;
  int a;
  int b;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class Solution {
  public List<Integer> spiralOrder(int[][] matrix) {
    List<Integer> res = new ArrayList<>();
    int m = matrix.length;
    int n = matrix[0].length;
```

```
14
15      int i = 0, j = 0;
16      int row = m, col = n;
17
18      int sum;
19
20      int total = m * n;
21      while (total > 0) {
22        if (row == 1) {
23          for (int k = 0; k < col; k++) {
24            res.add(matrix[i][j + k]);
25          }
26          sum = col;
27        } else if (col == 1) {
28          for (int k = 0; k < row; k++) {
29            res.add(matrix[i + k][j + col - 1]);
30          }
31          sum = row;
32        } else {
33          for (int k = 0; k < col - 1; k++) {
34            res.add(matrix[i][j + k]);
35          }
36          for (int k = 0; k < row - 1; k++) {
37            res.add(matrix[i + k][j + col - 1]);
38          }
39          for (int k = 0; k < col - 1; k++) {
40            res.add(matrix[i + row - 1][j + col - 1 - k]);
41          }
42          for (int k = 0; k < row - 1; k++) {
43            res.add(matrix[i + row - 1 - k][j]);
44          }
45          sum = 2 * (row + col) - 4;
46        }
47
48        total -= sum;
49
50        row -= 2;
51        col -= 2;
52
53        i++;
54        j++;
55      }
56      return res;
57    }
58
59    public static void main(String[] args) throws FileNotFoundException {
60      FileInputStream fin = new FileInputStream("./assets/fipt.txt");
61      PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));
62
63      System.setIn(fin);
64      System.setOut(fot);
65
66      Solution sol = new Solution();
67
68      sol.input();
69    }
70
71    public void input() {
72      Scanner in = new Scanner(System.in);
73      while (in.hasNext()) {
74        n = in.nextInt();
75        for (int i = 0; i < n; i++) {
76          a = in.nextInt();
77          b = in.nextInt();
78
79          int[][] v = new int[a][b];
80          for (int j = 0; j < a; j++) {
81            for (int k = 0; k < b; k++) {
82              v[j][k] = in.nextInt();
```

```
            }
          }
          solve(v);
        }
      }

      in.close();
    }

    public void solve(int[][] x) {
      List<Integer> res;
      res = spiralOrder(x);
      otput(res);

    }

    public void otput(List<Integer> x) {
      System.out.println(x);
    }

    private int n;
    private int a;
    private int b;
}
```

B.18 Leetcode 56

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:

  vector<vector<int> > merge(vector<vector<int> >& intervals) {
    vector<vector<int> > res;
    sort(intervals.begin(), intervals.end());

    for (int i = 0; i < intervals.size(); i++) {
      if (res.empty()) {
        res.push_back(intervals[i]);
      } else {
        if (res.back()[1] < intervals[i][0]) {
          res.push_back(intervals[i]);
        } else {
          if (res.back()[1] < intervals[i][1]) res.back()[1] = intervals[i][1];
        }
      }
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      int u, v;
      for (int i = 0; i < n; i++) {
```

```
        vector<int> t;
        scanf("%d %d", &u, &v);

        t.push_back(u);
        t.push_back(v);
        a.push_back(t);
      }

      solve(a);
      a.clear();
    }
  }
  void solve(vector<vector<int> >& x) {
    vector<vector<int> > res;
    res = merge(x);
    otput(res);
  }
  void otput(vector<vector<int> >& x) {
    vector<vector<int> > cur = x;
    for (int i = 0; i < cur.size(); i++)
      printf("%d %d\n", cur[i][0], cur[i][1]);
  }

 private:
  int n;
  vector<vector<int> > a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Comparator;
import java.util.Scanner;

class Solution {
  public int[][] merge(int[][] intervals) {
    ArrayList<int[]> res = new ArrayList<int[]>();
    Arrays.sort(intervals, new Comparator<int[]>() {
      @Override
      public int compare(int[] l, int[] r) {
        return l[0] - r[0];
      }
    });
    int sz = intervals.length;
    for (int i = 0; i < sz; i++) {
      int l = intervals[i][0];
      int r = intervals[i][1];
      while (i < sz - 1 && intervals[i + 1][0] <= r) {
        r = Math.max(r, intervals[i + 1][1]);
        i++;
      }
      res.add(new int[] { l, r });
    }
```

```java

    return res.toArray(new int[res.size()][2]);
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      vals = new int[n][2];

      for (int i = 0; i < n; i++) {
        vals[i][0] = in.nextInt();
        vals[i][1] = in.nextInt();
      }

      solve(vals);
    }

    in.close();
  }

  public void solve(int[][] x) {
    int[][] res;
    res = merge(x);
    otput(res);

  }

  public void otput(int[][] x) {
    int sz = x.length;
    for (int i = 0; i < sz; i++)
      System.out.println(x[i][0] + " " + x[i][1]);
  }

  private int n;
  private int[][] vals;
}
```

B.19 Leetcode 61

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;
```

```

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  ListNode* rotateRight(ListNode* head, int k) {
    if (head == NULL) return head;
    ListNode* res = head;
    ListNode* cur = head;
    ListNode* pre = NULL;
    ListNode* st = NULL;
    ListNode* en = NULL;
    vector<ListNode*> v;
    while (cur != NULL) {
      v.push_back(cur);

      if (cur->next == NULL) en = cur;
      cur = cur->next;
    }

    int lenOfList = v.size();
    int mk = k % lenOfList;
    if (mk != 0) {
      pre = v[lenOfList - mk - 1];
      st = v[lenOfList - mk];
      pre->next = NULL;
      en->next = res;
      res = st;
    }

    return res;
  }

  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      a = new ListNode(0);
      ListNode* u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a, n);
    }
  }
  void solve(ListNode* x, int y) {
    ListNode* res;
    res = rotateRight(x, y);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d", cur->val);
      cur = cur->next;
    }
    cout << endl;
  }

 private:
```

```
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.LinkedList;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode rotateRight(ListNode head, int k) {
    if (head == null)
      return head;
    ListNode res = head;
    ListNode cur = head;
    ListNode pre = null;
    ListNode st = null;
    ListNode en = null;
    LinkedList<ListNode> v = new LinkedList<ListNode>();
    while (cur != null) {
      v.add(cur);

      if (cur.next == null)
        en = cur;
      cur = cur.next;
    }

    int lenOfList = v.size();
    int mk = k % lenOfList;
    if (mk != 0) {
      pre = v.get(lenOfList - mk - 1);
      st = v.get(lenOfList - mk);
      pre.next = null;
      en.next = res;
      res = st;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
```

```
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < m; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a, n);
    }

    in.close();
  }

  public void solve(ListNode x, int y) {
    ListNode res;
    res = rotateRight(x, y);
    otput(res);
  }

  public void otput(ListNode x) {
    while (x != null) {
      System.out.print(x.val);
      x = x.next;
    }
    System.out.println("");
  }

  private int n, m;
}
```

B.20 Leetcode 121

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int maxProfit(vector<int>& prices) {
```

```
    int res = 0;
    int rmx = 0;
    for (int i = prices.size() - 2; i >= 0; i--) {
      rmx = max(rmx, prices[i + 1]);
      res = max(res, max(0, (rmx - prices[i])));
    }
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      int t;
      for (int i = 0; i < n; i++) {
        scanf("%d", &t);
        a.push_back(t);
      }
      solve(a);
      a.clear();
    }
  }
  void solve(vector<int>& x) {
    int res;
    res = maxProfit(x);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int maxProfit(int[] prices) {
    int res = 0;
    int rmx = 0;
    for (int i = prices.length - 2; i >= 0; i--) {
      rmx = Math.max(rmx, prices[i + 1]);
      res = Math.max(res, Math.max(0, (rmx - prices[i])));
    }
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

```

```
    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new int[n];
      for (int i = 0; i < n; i++)
        a[i] = in.nextInt();
      solve(a);
    }

    in.close();
  }

  public void solve(int[] x) {
    int res;
    res = maxProfit(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private int[] a;
}
```

B.21 Leetcode 138

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Node {
 public:
  int val;
  Node* next;
  Node* random;

  Node(int _val) {
    val = _val;
    next = NULL;
    random = NULL;
  }
};
class Solution {
 public:
  Node* copyRandomList(Node* head) {
    if (head == NULL) return NULL;

```

```
    Node* res = new Node(head->val);
    Node* cur_h = head;
    Node* cur_r = res;
    map<Node*, Node*> mp;
    mp.insert(pair<Node*, Node*>(NULL, NULL));

    while (cur_h != NULL) {
      if (cur_h != head) {
        cur_r->next = new Node(cur_h->val);
        cur_r = cur_r->next;
      }
      mp.insert(pair<Node*, Node*>(cur_h, cur_r));
      cur_h = cur_h->next;
    }

    cur_h = head;
    cur_r = res;
    while (cur_h != NULL) {
      cur_r->random = mp[cur_h->random];
      cur_r = cur_r->next;

      cur_h = cur_h->next;
    }

    return res;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      a = new Node(0);
      Node* u = a;
      int x;
      vector<int> v;
      vector<Node*> record;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u->next = new Node(0);
          u = u->next;
        }

        scanf("%d %d", &u->val, &x);
        v.push_back(x);
        record.push_back(u);
      }
      for (int i = 0; i < n; i++) {
        if (i + 1 < n) {
          record[i]->next = record[i + 1];
        }
        if (v[i] == 11111) {
          continue;
        }
        record[i]->random = record[v[i]];
      }

      solve(a);
    }
  }
  void solve(Node* x) {
    Node* res;
    res = copyRandomList(x);
    otput(res);
  }
  void otput(Node* x) {
    Node* cur = x;
    int cnt = 0;
    map<Node*, int> mp;
    while (cur != NULL) {
      mp.insert(pair<Node*, int>(cur, cnt));
      cnt++;
```

```
      cur = cur->next;
    }
    cur = x;
    while (cur != NULL) {
      printf("%d ", cur->val);
      if (cur->random == NULL)
        printf("null\n");
      else
        printf("%d\n", mp[cur->random]);

      cur = cur->next;
    }
  }

 private:
  int n;
  Node* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.HashMap;
import java.util.LinkedList;
import java.util.Map;
import java.util.Scanner;

class Node {
  int val;
  Node next;
  Node random;

  public Node(int val) {
    this.val = val;
    this.next = null;
    this.random = null;
  }
}

class Solution {
  public Node copyRandomList(Node head) {
    if (head == null)
      return null;

    Node res = new Node(head.val);
    Node cur_h = head;
    Node cur_r = res;
    Map<Node, Node> mp = new HashMap<Node, Node>();
    mp.put(null, null);

    while (cur_h != null) {
      if (cur_h != head) {
        cur_r.next = new Node(cur_h.val);
        cur_r = cur_r.next;
      }
```

```
      mp.put(cur_h, cur_r);
      cur_h = cur_h.next;
    }

    cur_h = head;
    cur_r = res;
    while (cur_h != null) {
      cur_r.random = mp.get(cur_h.random);
      cur_r = cur_r.next;

      cur_h = cur_h.next;
    }

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      Node a = new Node(0);
      Node u = a;
      LinkedList<Integer> v = new LinkedList<Integer>();
      LinkedList<Node> record = new LinkedList<Node>();

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u.next = new Node(0);
          u = u.next;
        }
        u.val = in.nextInt();

        v.add(in.nextInt());
        record.add(u);
      }
      for (int i = 0; i < n; i++) {
        if (i + 1 < n) {
          record.get(i).next = record.get(i + 1);
        }
        if (v.get(i) == 11111) {
          continue;
        }
        record.get(i).random = record.get(v.get(i));
      }

      solve(a);
    }

    in.close();
  }

  public void solve(Node x) {
    Node res;
    res = copyRandomList(x);
    otput(res);

```

```
  }

  public void otput(Node x) {
    Node cur = x;
    int cnt = 0;
    Map<Node, Integer> mp = new HashMap<Node, Integer>();
    while (cur != null) {
      mp.put(cur, cnt);
      cnt++;
      cur = cur.next;
    }
    cur = x;
    while (cur != null) {
      System.out.print(cur.val + " ");
      if (cur.random == null)
        System.out.println("null");
      else
        System.out.println(mp.get(cur.random));

      cur = cur.next;
    }

  }

  private int n, m;
}
```

B.22 Leetcode 141

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
  bool hasCycle(ListNode* head) {
    if (NULL == head) return false;
    ListNode *slow = head, *fast = head->next;
    while (NULL != fast) {
      if (slow == fast) return true;
      slow = slow->next;
      fast = fast->next;
      if (NULL != fast) fast = fast->next;
    }
    return false;
  }

  void input(void) {
    while (~scanf("%d %d", &n, &m)) {
      int t;
```

```
37
38        scanf("%d", &t);
39        a = new ListNode(t);
40        ListNode* cur = a;
41
42        for (int i = 1; i < n; i++) {
43          scanf("%d", &t);
44          cur->next = new ListNode(t);
45          cur = cur->next;
46        }
47        ListNode* back = cur;
48        cur = a;
49        if (m >= 0) {
50          for (int i = 0; i < m - 1; i++) cur = cur->next;
51          back->next = cur;
52        }
53
54        solve(a);
55      }
56    }
57
58    void solve(ListNode* x) {
59      bool res;
60      res = hasCycle(x);
61      otput(res);
62    }
63    void otput(int x) { printf("%d\n", x); }
64
65   private:
66    int n, m;
67    ListNode* a;
68  };
69
70  int main() {
71    freopen("./assets/fipt.txt", "r", stdin);
72    freopen("./assets/fopt.txt", "w", stdout);
73
74    Solution sol;
75
76    sol.input();
77
78    return 0;
79  }
```

Java

```
1  import java.io.FileInputStream;
2  import java.io.FileNotFoundException;
3  import java.io.FileOutputStream;
4  import java.io.PrintStream;
5  import java.util.Scanner;
6
7  class Solution {
8    class ListNode {
9      int val;
10     ListNode next;
11
12     ListNode(int x) {
13       val = x;
14       next = null;
15     }
16   }
17
18   public boolean hasCycle(ListNode head) {
19     if (null == head)
20       return false;
21     ListNode slow = head, fast = head.next;
22     while (null != fast) {
23       if (slow == fast)
```

```java
        return true;
      slow = slow.next;
      fast = fast.next;
      if (null != fast)
        fast = fast.next;
    }
    return false;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();
      a = new ListNode(0);

      int cnt = 0;
      for (int i = 0; i < n; i++) {
        if (cnt == 0) {
          a.val = in.nextInt();
        } else {
          a.next = new ListNode(in.nextInt());
        }
      }

      solve(a);
    }

    in.close();
  }

  public void solve(ListNode x) {
    boolean res;
    res = hasCycle(x);
    otput(res);

  }

  public void otput(boolean x) {
    System.out.println(x);
  }

  private int n, m;
  private ListNode a;
}
```

B.23 Leetcode 172

C++

```cpp
#include <math.h>

#include <algorithm>
#include <cstdio>
#include <cstdlib>
```

```cpp
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int trailingZeroes(int n) {
    int res = 0;
    while (n) {
      res += n / 5;
      n /= 5;
    }
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
        solve(a);
      }
    }
  }

  void solve(int x) {
    int res;
    res = trailingZeroes(x);
    otput(res);
  }

  void otput(int x) { cout << x << endl; }

 private:
  int n;
  int a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int trailingZeroes(int n) {
    int res = 0;
    while (n != 0) {
      res += n / 5;
      n /= 5;
    }
```

```java
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      for (int i = 0; i < n; i++) {
        a = in.nextInt();
        solve(a);
      }
    }

    in.close();
  }

  public void solve(int x) {
    int res;
    res = trailingZeroes(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private int a;
}
```

B.24 Leetcode 202

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int cal(int x) {
    int res = 0;
    int t;
    while (x != 0) {
      t = x % 10;
```

```
      x /= 10;
      res += t * t;
    }
    return res;
  }
  bool isHappy(int n) {
    if (n == 1) return true;
    int slow = n, fast = cal(n);
    while (slow != fast) {
      if (slow == 1 || fast == 1) return true;
      slow = cal(slow);
      fast = cal(cal(fast));
    }
    return false;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      solve(n);
    }
  }

  void solve(int x) {
    bool res;
    res = isHappy(x);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n, m;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int cal(int x) {
    int res = 0;
    int t;
    while (x != 0) {
      t = x % 10;
      x /= 10;
      res += t * t;
    }
    return res;
  }

  public boolean isHappy(int n) {
    if (n == 1)
      return true;
    int slow = n, fast = cal(n);
    while (slow != fast) {
```

```java
      if (slow == 1 || fast == 1)
        return true;
      slow = cal(slow);
      fast = cal(cal(fast));
    }
    return false;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      solve(n);
    }

    in.close();
  }

  public void solve(int x) {
    boolean res;
    res = isHappy(x);
    otput(res);

  }

  public void otput(boolean x) {
    System.out.println(x);
  }

  private int n, m;
  private ListNode a;
}
```

B.25 Leetcode 206

C++

```cpp
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;
```

```

class Solution {
 public:
  ListNode* reverseList(ListNode* head) {
    if (head == NULL) return NULL;

    ListNode* st = head;
    ListNode* en = NULL;
    ListNode* cur = head;
    while (cur->next != NULL) {
      cur = cur->next;
    }
    en = cur;
    while (st != en) {
      cur = st->next;
      st->next = en->next;
      en->next = st;
      st = cur;
    }
    return en;
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      a = new ListNode(0);
      ListNode* u = a;
      int x;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u->next = new ListNode(0);
          u = u->next;
        }

        scanf("%d", &u->val);
      }

      solve(a);
    }
  }
  void solve(ListNode* x) {
    ListNode* res;
    res = reverseList(x);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d\n", cur->val);
      cur = cur->next;
    }
  }

 private:
  int n;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode reverseList(ListNode head) {
    if (head == null)
      return null;

    ListNode st = head;
    ListNode en = null;
    ListNode cur = head;
    while (cur.next != null) {
      cur = cur.next;
    }
    en = cur;
    while (st != en) {
      cur = st.next;
      st.next = en.next;
      en.next = st;
      st = cur;
    }
    return en;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      ListNode a = new ListNode(0);
      ListNode u = a;

      for (int i = 0; i < n; i++) {
        if (i != 0) {
          u.next = new ListNode(0);
          u = u.next;
        }
        u.val = in.nextInt();
      }

      solve(a);
    }

    in.close();
```

```
69    }
70
71    public void solve(ListNode x) {
72      ListNode res;
73      res = reverseList(x);
74      otput(res);
75
76    }
77
78    public void otput(ListNode x) {
79      ListNode cur = x;
80      while (cur != null) {
81        System.out.println(cur.val);
82        cur = cur.next;
83      }
84
85    }
86
87    private int n, m;
88  }
```

B.26 Leetcode 209

C++

```
1   #include <algorithm>
2   #include <cstdio>
3   #include <cstdlib>
4   #include <cstring>
5   #include <iostream>
6   #include <map>
7   #include <queue>
8   #include <stack>
9   #include <string>
10  #include <vector>
11
12  using namespace std;
13
14  class Solution {
15   public:
16    int minSubArrayLen(int s, vector<int>& nums) {
17      int res = nums.size();
18      int sz = res;
19      int sum = 0;
20      int u = 0, v = 0;
21      bool occur = false;
22
23      for (v = 0; v < sz; v++) {
24        sum += nums[v];
25        while (sum >= s && u <= v) {
26          occur = true;
27          res = min(res, v - u + 1);
28          sum -= nums[u];
29          u++;
30        }
31      }
32      if (occur == false) return 0;
33
34      return res;
35    }
36
37    void input(void) {
38      while (~scanf("%d %d", &n, &m)) {
39        int t;
40        for (int i = 0; i < n; i++) {
41          scanf("%d", &t);
42          a.push_back(t);
```

```
      }
      solve(m, a);
      a.clear();
    }
  }
  void solve(int s, vector<int>& x) {
    int res;
    res = minSubArrayLen(s, x);
    otput(res);
  }
  void otput(int x) { printf("%d\n", x); }

 private:
  int n, m;
  vector<int> a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int minSubArrayLen(int s, int[] nums) {
    int res = nums.length;
    int sz = res;
    int sum = 0;
    int u = 0, v = 0;
    boolean occur = false;

    for (v = 0; v < sz; v++) {
      sum += nums[v];
      while (sum >= s && u <= v) {
        occur = true;
        res = Math.min(res, v - u + 1);
        sum -= nums[u];
        u++;
      }
    }
    if (occur == false)
      return 0;

    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
```

```
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      m = in.nextInt();
      a = new int[n];
      for (int i = 0; i < n; i++)
        a[i] = in.nextInt();
      solve(m, a);
    }

    in.close();
  }

  public void solve(int s, int[] x) {
    int res;
    res = minSubArrayLen(s, x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n, m;
  private int[] a;
}
```

B.27 Leetcode 258

C++

```
#include <math.h>

#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int addDigits(int num) {
    if (num < 10) {
      return num;
    } else {
      int res = num % 9;
      if (res == 0) {
        res = 9;
      }
      return res;
    }
  }
  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
```

```
        solve(a);
      }
    }
  }

  void solve(int x) {
    int res;
    res = addDigits(x);
    otput(res);
  }

  void otput(int x) { cout << x << endl; }

 private:
  int n;
  int a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int addDigits(int num) {
    if (num < 10) {
      return num;
    } else {
      int res = num % 9;
      if (res == 0) {
        res = 9;
      }
      return res;
    }
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      for (int i = 0; i < n; i++) {
        a = in.nextInt();
        solve(a);
```

```
      }
    }

    in.close();
  }

  public void solve(int x) {
    int res;
    res = addDigits(x);
    otput(res);

  }

  public void otput(int x) {
    System.out.println(x);
  }

  private int n;
  private int a;
}
```

B.28 Leetcode 387

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  int firstUniqChar(string s) {
    int ans = -1;
    int u, v;
    for (char ch = 'a'; ch <= 'z'; ch++) {
      u = s.find(ch);
      v = s.rfind(ch);
      if (u == v && u != -1) {
        if (ans == -1) {
          ans = u;
        } else {
          if (u < ans) ans = u;
        }
      }
    }
    return ans;
  }
  void input(void) {
    while (cin >> str) {
      solve(str);
    }
  }
  void solve(string s) {
    int res;
    res = firstUniqChar(s);
    otput(res);
  }
  void otput(int id) { printf("%d\n", id); }
```

```

 private:
  int n, m, t;
  string str;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public int firstUniqChar(String s) {
    int ans = -1;
    int u, v;
    for (char ch = 'a'; ch <= 'z'; ch++) {
      u = s.indexOf(ch);
      v = s.lastIndexOf(ch);
      if (u == v && u != -1) {
        if (ans == -1) {
          ans = u;
        } else {
          if (u < ans)
            ans = u;
        }
      }
    }
    return ans;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      String str = in.nextLine();
      solve(str);
    }

    in.close();
  }

  public void solve(String s) {
    int res;
    res = firstUniqChar(s);
```

```
    otput(res);
  }

  public void otput(int idx) {
    System.out.println(idx);
  }

  private int n, m, t;
  private int u, v;
  private int[] numbers = new int[10000];
}
```

B.29 Leetcode 763

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

class Solution {
 public:
  vector<int> partitionLabels(string S) {
    vector<int> res;
    int u[26] = {0};

    int sz = S.size();
    for (int i = 0; i < sz; i++) {
      u[S[i] - 'a'] = i;
    }

    int l = 0, r = u[S[0] - 'a'];
    for (int i = 0; i < sz; i++) {
      r = max(r, u[S[i] - 'a']);
      if (r == i) {
        res.push_back(r - l + 1);
        l = i + 1;
      }
    }
    return res;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
      for (int i = 0; i < n; i++) {
        cin >> a;
        solve(a);
      }
    }
  }

  void solve(string x) {
    vector<int> res;
    res = partitionLabels(x);
    otput(res);
  }
  void otput(vector<int> x) {
    for (int i = 0; i < x.size(); i++) cout << x[i] << endl;
```

```
  }

 private:
  int n;
  string a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;

class Solution {
  public List<Integer> partitionLabels(String S) {
    List<Integer> res = new ArrayList<>();
    List<Integer> u = new ArrayList<Integer>(Collections.nCopies(26, -1));

    int sz = S.length();
    for (int i = 0; i < sz; i++) {
      u.set(S.charAt(i) - 'a', i);
    }

    int l = 0, r = u.get(S.charAt(0) - 'a');
    for (int i = 0; i < sz; i++) {
      r = Math.max(r, u.get(S.charAt(i) - 'a'));
      if (r == i) {
        res.add(r - l + 1);
        l = i + 1;
      }
    }
    return res;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();

      for (int i = 0; i < n; i++) {
        a = in.next();
```

```
      }

      solve(a);
    }

    in.close();
  }

  public void solve(String x) {
    List<Integer> res;
    res = partitionLabels(x);
    otput(res);

  }

  public void otput(List<Integer> x) {
    System.out.println(x);
  }

  private int n;
  private String a = null;
}
```

B.30 Leetcode 876

C++

```
#include <algorithm>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <map>
#include <queue>
#include <stack>
#include <string>
#include <vector>

using namespace std;

typedef struct ListNode {
  int val;
  ListNode* next;
  ListNode(int x) : val(x), next(NULL) {}
} ListNode;

class Solution {
 public:
 public:
  ListNode* middleNode(ListNode* head) {
    if (NULL == head) return NULL;
    ListNode* slow = head;
    ListNode* fast = head->next;
    while (fast != NULL) {
      slow = slow->next;
      fast = fast->next;
      if (fast == NULL) {
        return slow;
      } else {
        fast = fast->next;
      }
    }
    return slow;
  }

  void input(void) {
    while (~scanf("%d", &n)) {
```

```
      a = new ListNode(0);
      ListNode* cur = a;
      for (int i = 0; i < n; i++) {
        int t;
        scanf("%d", &t);
        if (i == 0) {
          cur->val = t;
        } else {
          cur->next = new ListNode(t);
          cur = cur->next;
        }
      }
      solve(a);
    }
  }

  void solve(ListNode* x) {
    ListNode* res;
    res = middleNode(x);
    otput(res);
  }
  void otput(ListNode* x) {
    ListNode* cur = x;
    while (cur != NULL) {
      printf("%d\n", cur->val);
      cur = cur->next;
    }
  }

 private:
  int n, m;
  ListNode* a;
};

int main() {
  freopen("./assets/fipt.txt", "r", stdin);
  freopen("./assets/fopt.txt", "w", stdout);

  Solution sol;

  sol.input();

  return 0;
}
```

Java

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;

class Solution {
  public class ListNode {
    int val;
    ListNode next;

    ListNode(int x) {
      val = x;
    }
  }

  public ListNode middleNode(ListNode head) {
    if (null == head)
      return null;
    ListNode slow = head;
    ListNode fast = head.next;
    while (fast != null) {
```

```
      slow = slow.next;
      fast = fast.next;
      if (fast == null) {
        return slow;
      } else {
        fast = fast.next;
      }
    }
    return slow;
  }

  public static void main(String[] args) throws FileNotFoundException {
    FileInputStream fin = new FileInputStream("./assets/fipt.txt");
    PrintStream fot = new PrintStream(new FileOutputStream("./assets/fopt.txt"));

    System.setIn(fin);
    System.setOut(fot);

    Solution sol = new Solution();

    sol.input();
  }

  public void input() {
    Scanner in = new Scanner(System.in);
    while (in.hasNext()) {
      n = in.nextInt();
      a = new ListNode(0);
      ListNode cur = a;
      for (int i = 0; i < n; i++) {
        if (i == 0) {
          cur.val = in.nextInt();
        } else {
          cur.next = new ListNode(in.nextInt());
          cur = cur.next;
        }
      }
      solve(a);
    }

    in.close();
  }

  public void solve(ListNode x) {
    ListNode res;
    res = middleNode(x);
    otput(res);

  }

  public void otput(ListNode x) {
    ListNode cur = x;
    while (cur != null) {
      System.out.println(cur.val);
      cur = cur.next;
    }
  }

  private int n, m;
  private ListNode a;
}
```