Assignment #7: 矩阵、队列、贪心

Updated 1315 GMT+8 Oct 21, 2025

2025 fall, Complied by 郭旭杰、化学与分子工程学院

OpenJudge账号：25n2500011906，昵称：郭旭杰

CodeForces账号：LittleBeetroot，段位：Newbie（新手）

> **说明：**
>
> 1. **解题与记录：**
>
> 对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge， Codeforces，LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora https://typoraio.cn 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。
>
> 2. 提交安排：**提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的本人头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。
> 3. **延迟提交：**如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。
>
> 请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

M12560: 生存游戏

matrices, http://cs101.openjudge.cn/pctbook/M12560/

耗时：15min速通，没什么难度。

思路：就是要在矩阵外围加一层保护圈，输出前再去掉。

M12560:生存游戏

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

全局题号 12560
添加于 2025-03-13
提交次数 246
尝试人数 176
通过人数 175

你的提交记录

#	结果	时间
1	Accepted	2025-10-21

描述

有如下生存游戏的规则:

给定一个n*m(1<=n,m<=100)的数组，每个元素代表一个细胞，其初始状态为活着(1)或死去(0)。<p="">

每个细胞会与其相邻的8个邻居（除数组边缘的细胞）进行交互，并遵守如下规则:

任何一个活着的细胞如果只有小于2个活着的邻居，那它就会由于人口稀少死去。

任何一个活着的细胞如果有2个或者3个活着的邻居，就可以继续活下去。

任何一个活着的细胞如果有超过3个活着的邻居，那它就会由于人口拥挤而死去。

任何一个死去的细胞如果有恰好3个活着的邻居，那它就会由于繁殖而重新变成活着的状态。

请写一个函数用来计算所给定初始状态的细胞经过一次更新后的状态是什么。

注意：所有细胞的状态必须同时更新，不能使用更新后的状态作为其他细胞的邻居状态来进行计算。

输入

第一行为n和m，而后n行，每行m个元素，用空格隔开。

输出

n行，每行m个元素，用空格隔开。

代码

```
n, m = list(map(int, input().split()))
matrix = []
row0 = [0 for i in range(m + 2)]
matrix.append(row0)
for i in range(1, n + 1):
    row = list(map(int, input().split()))
    row1 = [0]
    for j in range(m):
        row1.append(row[j])
    row1.append(0)
    matrix.append(row1)
matrix.append(row0)

new_matrix = []
for i in range(1, n + 1):
    new_row = []
    for j in range(1, m + 1):
        new_row.append(matrix[i][j])
    new_matrix.append(new_row)

for i in range(1, n + 1):
```

```
    for j in range(1, m + 1):
        sgm = matrix[i - 1][j - 1] + matrix[i - 1][j] + matrix[i - 1][j + 1] + matrix[i][j
- 1] + matrix[i][j + 1] + matrix[i + 1][j - 1] + matrix[i + 1][j] + matrix[i + 1][j + 1]
        if matrix[i][j] == 1:
            if 2 <= sgm <= 3:
                new_matrix[i - 1][j - 1] = 1
            else:
                new_matrix[i - 1][j - 1] = 0
        if matrix[i][j] == 0:
            if sgm == 3:
                new_matrix[i - 1][j - 1] = 1
            else:
                new_matrix[i - 1][j - 1] = 0



for new_row in new_matrix:
    print(*new_row)
```

代码运行截图 （至少包含有"Accepted"）

状态: Accepted

源代码

```
n, m = list(map(int, input().split()))
matrix = []
row0 = [0 for i in range(m + 2)]
matrix.append(row0)
for i in range(1, n + 1):
    row = list(map(int, input().split()))
    row1 = [0]
    for j in range(m):
        row1.append(row[j])
    row1.append(0)
    matrix.append(row1)
matrix.append(row0)

new_matrix = []
for i in range(1, n + 1):
    new_row = []
    for j in range(1, m + 1):
        new_row.append(matrix[i][j])
    new_matrix.append(new_row)

for i in range(1, n + 1):
    for j in range(1, m + 1):
        sgm = matrix[i - 1][j - 1] + matrix[i - 1][j] + matrix[i - 1][j
        if matrix[i][j] == 1:
            if 2 <= sgm <= 3:
                new_matrix[i - 1][j - 1] = 1
            else:
                new_matrix[i - 1][j - 1] = 0
        if matrix[i][j] == 0:
            if sgm == 3:
                new_matrix[i - 1][j - 1] = 1
            else:
                new_matrix[i - 1][j - 1] = 0



for new_row in new_matrix:
    print(*new_row)
```

基本信息

#:	50480457
题目:	M12560
提交人:	25n2500011906
内存:	4084kB
时间:	37ms
语言:	Python3
提交时间:	2025-10-21 15:31:01

M04133:垃圾炸弹

matrices, http://cs101.openjudge.cn/pctbook/M04133/

耗时：找思路找了两天，找到思路后20min速通。

思路：与雷达安装问题相似，涉及变换参考系，可以以垃圾为参考系，在其周围一定半径内安装炸弹所能产生的有效值相应增加。找到有效值最大的点即可。

涉及到max(),min()函数的巧妙运用。

代码

```
d = int(input())
n = int(input())
matrix = [[0 for i in range(1025)] for j in range(2025)]
js = []
for _ in range(n):
    x, y, h = list(map(int, input().split()))
```

```
    for i in range(max(x - d, 0), min(x + d + 1, 1025)):
        for j in range(max(y - d, 0), min(y + d + 1, 1025)):
            matrix[i][j] += h
for i in range(1025):
    for j in range(1025):
        js.append(matrix[i][j])
js.sort(reverse=True)
cnt = 0
while js[cnt] == js[0]:
    cnt += 1
print(cnt, js[0])
```

代码运行截图 (至少包含有"Accepted")

M04133:垃圾炸弹

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

全局题号 7213
添加于 2025-03-13
提交次数 1181
尝试人数 183
通过人数 177

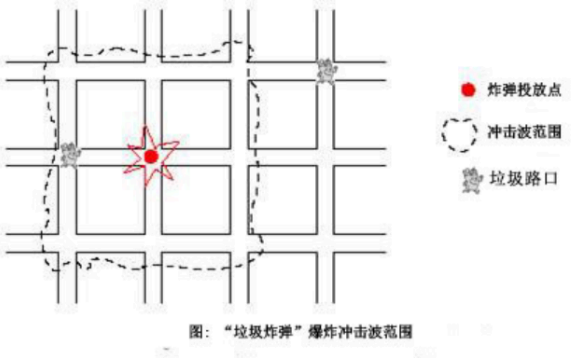
你的提交记录

#	结果	时间
1	Accepted	2025-10-24

描述

2018年俄罗斯世界杯（2018 FIFA World Cup）开踢啦！为了方便球迷观看比赛，莫斯科街道上很多路口都放置了的直播大屏幕，但是人群散去后总会在这些路口留下一堆垃圾。为此俄罗斯政府决定动用一种最新发明——“垃圾炸弹”。这种“炸弹”利用最先进的量子物理技术，爆炸后产生的冲击波可以完全清除波及范围内的所有垃圾，并且不会产生任何其他不良影响。炸弹爆炸后冲击波是以正方形方式扩散的，炸弹威力（扩散距离）以d给出，表示可以传播d条街道。

例如下图是一个d=1的“垃圾炸弹”爆炸后的波及范围。



图：“垃圾炸弹”爆炸冲击波范围

假设莫斯科的布局为严格的1025*1025的网格状，由于财政问题市政府只买得起一枚“垃圾炸弹”，希望你帮他们找到合适的投放地点，使得一次清除的垃圾总量最多（假设垃圾数量可以用一个非负整数表示，并且除设置大屏幕的路口以外的地点没有垃圾）。

输入

第一行给出"炸弹"威力d(1 <= d <= 50)。第二行给出一个数组n(1 <= n <= 20)表示设置了大屏幕(有垃圾)的路口数目。接下来n行每行给出三个数字x, y, i, 分别代表路口的坐标(x, y)以及垃圾数量i. 点坐标(x, y)保证是有效的（区间在0到1024之间），同一坐标只会给出一次。

OpenJudge 题目ID, 标题, 描述 25n2500011906 信箱 账号

CS101 / 计算思维算法实践

题目 排名 状态 提问

#50529656提交状态 查看 提交 统计 提问

状态: Accepted

源代码

```
d = int(input())
n = int(input())
matrix = [[0 for i in range(1025)] for j in range(2025)]
js = []
for _ in range(n):
    x, y, h = list(map(int, input().split()))
    for i in range(max(x - d, 0), min(x + d + 1, 1025)):
        for j in range(max(y - d, 0), min(y + d + 1, 1025)):
            matrix[i][j] += h
for i in range(1025):
    for j in range(1025):
        js.append(matrix[i][j])
js.sort(reverse=True)
cnt = 0
while js[cnt] == js[0]:
    cnt += 1
print(cnt, js[0])
```

基本信息

#: 50529656
题目: M04133
提交人: 25n2500011906
内存: 29896kB
时间: 258ms
语言: Python3
提交时间: 2025-10-24 01:44:18

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

M02746: 约瑟夫问题

implementation, queue, http://cs101.openjudge.cn/pctbook/M02746/

耗时：25分钟通过，有一点难度。

思路：让列表环化有点难度，还有就是在不断淘汰猴子的过程中，猴子的总数发生变化，这也会使列表的长度减小。

代码

```
blacklist = 0
while blacklist == 0:
    n, m = list(map(int, input().split()))
    if m + n >= 1:
        otto = []
        cnt = 0
        o = 0
        for i in range(1, n + 1):
            otto.append(i)
        while len(otto) > 1:
            cnt += 1
            o += 1
            if cnt == m:
                cnt = 0
                ot = (o - 1) % len(otto)
                otto.remove(otto[ot])
                o = ot
        print(*otto)
```

```
    else:
        blacklist = 1
        break
```

代码运行截图 （至少包含有"Accepted"）

M26976:摆动序列

greedy, http://cs101.openjudge.cn/pctbook/M26976/

思路：30min通过，思路好找，但有点难。

转化列表，先排除相等情况，再逐对排查，有动态规划思想。

代码

```
n = int(input())
js = list(map(int, input().split()))

if n == 1 or max(js) - min(js) == 0:
    print(1)
elif n == 2 and js[0]!=js[1]:
    print(2)
else:
    k = 0

    stt = 1
    v = 0
    ks = [js[i] - js[i - 1] for i in range(1, n)]
    if 0 in ks:
        ks.remove(0)
```

```
    if ks[0] < 0:
        stt = -1

    for i in range(0, len(ks)):
        if ks[i] // stt > 0:
            stt *= (-1)
            k += 1

    print(k + 1)
```

代码运行截图 （至少包含有"Accepted"）

CS101 / 计算思维算法实践

题目 排名 状态 提问

M26976:摆动序列

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

全局题号 **26976**

添加于 **2025-10-21**

提交次数 **512**

尝试人数 **153**

通过人数 **150**

你的提交记录

#	结果	时间
7	Accepted	2025-10-23
6	Runtime Error	2025-10-23
5	Runtime Error	2025-10-23
4	Runtime Error	2025-10-23
3	Runtime Error	2025-10-23
2	Wrong Answer	2025-10-23
1	Runtime Error	2025-10-23

描述

如果连续数字之间的差严格地在正数和负数之间交替，则数字序列称为 **摆动序列** 。第一个差（如果存在的话）可能是正数或负数。仅有一个元素或者含两个不等元素的序列也视作摆动序列。

例如，[1, 7, 4, 9, 2, 5] 是一个 **摆动序列** ，因为差值 (6, -3, 5, -7, 3) 是正负交替出现的。

相反，[1, 4, 7, 2, 5]，[1, 7, 4, 5, 5]，不是摆动序列，第一个序列是因为它的前两个差值都是正数，第二个序列是因为它的最后一个差值为零。

子序列 可以通过从原始序列中删除一些（也可以不删除）元素来获得，剩下的元素保持其原始顺序。

给你一个整数数组 nums ，返回 nums 中作为 **摆动序列** 的 **最长子序列的长度** 。

输入

第一行包含一个整数n。1 <= n <= 1000

第二行包含n个整数，相邻整数间以空格隔开。0 <= nums[i] <= 1000

输出

一个整数

T26971:分发糖果

greedy, http://cs101.openjudge.cn/pctbook/T26971/

耗时：5天（找了3天思路后发现思路还是错的，后来看了LeetCode上面的原题题解才恍然大悟）。

思路：需要注意，如果两个小孩评分相等，那么这两个小孩得到的糖无论谁多谁少，差距多大，在规则上都是合理的，所以相对于定义的mounti函数，更适用于hillgravedown函数。

定义之后直接调用函数即可。

注意贪心算法使用时也不要理所当然地认为相邻两个小孩的糖个数相差1就是最优解。

代码

```
def hillgravedown(listy):
    hillgravedown_n = len(listy)
    hilldown_left = [0] * hillgravedown_n
    for i in range(hillgravedown_n):
        if i > 0 and listy[i] > listy[i - 1]:
            hilldown_left[i] = hilldown_left[i - 1] + 1
        else:
            hilldown_left[i] = 1
```

```
    hilldown_right = hillgravedown_dust = 0
    for i in range(hillgravedown_n - 1, -1, -1):
        if i < hillgravedown_n - 1 and listy[i] > listy[i + 1]:
            hilldown_right += 1
        else:
            hilldown_right = 1
        hillgravedown_dust += max(hilldown_left[i], hilldown_right)

    return hillgravedown_dust


n = int(input())
js = list(map(int, input().split()))
print(hillgravedown(js))
```

代码运行截图 (至少包含有"Accepted")

OpenJudge 题目ID, 标题, 描述 25n2500011906 信箱 账号

CS101 / 计算思维算法实践

题目 排名 状态 提问

#50573909提交状态 查看 提交 统计 提问

状态: Accepted

源代码

```
def hillgravedown(listy):
    hillgravedown_n = len(listy)
    hilldown_left = [0] * hillgravedown_n
    for i in range(hillgravedown_n):
        if i > 0 and listy[i] > listy[i - 1]:
            hilldown_left[i] = hilldown_left[i - 1] + 1
        else:
            hilldown_left[i] = 1

    hilldown_right = hillgravedown_dust = 0
    for i in range(hillgravedown_n - 1, -1, -1):
        if i < hillgravedown_n - 1 and listy[i] > listy[i + 1]:
            hilldown_right += 1
        else:
            hilldown_right = 1
        hillgravedown_dust += max(hilldown_left[i], hilldown_right)

    return hillgravedown_dust


n = int(input())
js = list(map(int, input().split()))
print(hillgravedown(js))
```

基本信息

#: 50573909
题目: T26971
提交人: 25n2500011906
内存: 5052kB
时间: 29ms
语言: Python3
提交时间: 2025-10-26 19:36:16

English 帮助 关于

1868A. Fill in the Matrix

constructive algorithms, implementation, 1300, https://codeforces.com/problemset/problem/1868/A

耗时：5天（找了一天思路，代码调整了4天）。

思路：只要让数字从小往大逐个排列，然后不断把第一个数放到队列末尾，直到达到n或者只剩一种情况，如果没有达到n就一直填原始序列genlistie直到n行为止。

代码

```
def to_listie(n_to_listie):
    return [i_to_listie for i_to_listie in range(n_to_listie)]


def str_to_listie(n_to_listie):
    return [str(i_to_listie) for i_to_listie in range(n_to_listie)]


def cata_findm(n0, m0):
    if m0 == 1:
        return 0
    else:
```

```
        return min(n0 + 1, m0)


def alter_listie(listy):
    first = listy.pop(0)
    listy.append(first)
    return listy


def kpj(strry_listy):
    return ' '.join(strry_listy)


Lne = int(input())
for _l in range(Lne):
    n, m = list(map(int, input().split()))
    print(cata_findm(n, m))
    if m == 1:
        for _ in range(n):
            print(0)
    else:
        genlistie = str_to_listie(m)
        ltt = str_to_listie(m)
        cnt = 1
        while cnt <= min(m - 1, n):
            print(kpj(ltt))
            cnt += 1
            ltt = alter_listie(ltt)
        if m - 1 < n:
            for _ in range(n - m + 1):
                print(kpj(genlistie))
```

代码运行截图 （至少包含有"Accepted"）

PROBLEMS SUBMIT CODE MY SUBMISSIONS STATUS HACKS ROOM STANDINGS CUSTOM INVOCATION

My Submissions

#	When	Who	Problem	Lang	Verdict	Time	Memory
345883033	Oct/26/2025 20:03 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3	Accepted	281 ms	18500 KB
345669283	Oct/25/2025 08:52 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3	Wrong answer on test 2	202 ms	4300 KB
345664774	Oct/25/2025 06:39 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3	Runtime error on test 1	109 ms	2100 KB
345645771	Oct/25/2025 02:58 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3	Wrong answer on test 2	218 ms	4200 KB
345645516	Oct/25/2025 02:56 UTC+8	LittleBeetroot	A - Fill in the Matrix	Python 3	Wrong answer on test 2	61 ms	200 KB
345645406	Oct/25/2025 02:56 UTC+8	LittleBeetroot	A - Fill in the Matrix	Python 2	Compilation error	0 ms	0 KB
345645205	Oct/25/2025 02:54 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 1	93 ms	0 KB
345644716	Oct/25/2025 02:52 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 2	124 ms	3100 KB
345638102	Oct/25/2025 02:16 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 2	171 ms	3600 KB
345637326	Oct/25/2025 02:12 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 1	46 ms	0 KB
345630358	Oct/25/2025 02:00 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 1	62 ms	0 KB
345630033	Oct/25/2025 01:59 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Runtime error on test 1	92 ms	2700 KB
345476436	Oct/24/2025 17:31 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Time limit exceeded on test 12	2000 ms	92300 KB
345476206	Oct/24/2025 17:30 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 2	171 ms	4200 KB
345405518	Oct/24/2025 03:03 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Time limit exceeded on test 1	2000 ms	1300 KB
345405258	Oct/24/2025 03:01 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 1	78 ms	0 KB
345404231	Oct/24/2025 02:52 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Memory limit exceeded on test 7	968 ms	262100 KB
345404045	Oct/24/2025 02:51 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 2	171 ms	4300 KB
345403836	Oct/24/2025 02:49 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Runtime error on test 1	93 ms	2700 KB
345402966	Oct/24/2025 02:42 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 2	139 ms	4200 KB
345402115	Oct/24/2025 02:35 UTC+8	LittleBeetroot	A - Fill in the Matrix	PyPy 3-64	Wrong answer on test 2	155 ms	3500 KB

PROBLEMS SUBMIT CODE MY SUBMISSIONS STATUS HACKS ROOM STANDINGS CUSTOM INVOCATION

General

#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged		
345883033	Practice: LittleBeetroot	1868A - 34	PyPy 3	Accepted	281 ms	18452 KB	2025-10-26 15:03:17	2025-10-26 15:03:17		Compare

→ Source

Copy

```
def to_listie(n_to_listie):
    return [i_to_listie for i_to_listie in range(n_to_listie)]


def str_to_listie(n_to_listie):
    return [str(i_to_listie) for i_to_listie in range(n_to_listie)]


def cata_findm(n0, m0):
    if m0 == 1:
        return 0
    else:
        return min(n0 + 1, m0)


def alter_listie(listy):
    first = listy.pop(0)
    listy.append(first)
    return listy


def kpj(strry_listy):
    return ' '.join(strry_listy)


Lne = int(input())
for _l in range(Lne):
    n, m = list(map(int, input().split()))
    print(cata_findm(n, m))
    if m == 1:
        for _ in range(n):
            print(0)
    else:
        genlistie = str_to_listie(m)
        ltt = str_to_listie(m)
        cnt = 1
        while cnt <= min(m - 1, n):
            print(kpj(ltt))
            cnt += 1
            ltt = alter_listie(ltt)
        if m - 1 < n:
            for _ in range(n - m + 1):
                print(kpj(genlistie))
```

Click to see test details

2. 学习总结和收获

学习了矩阵相关问题（M12560: 生存游戏、M04133:垃圾炸弹、1868A. Fill in the Matrix）和序列贪心（M02746: 约瑟夫问题、M26976:摆动序列、T26971:分发糖果）问题

如果作业题目简单，有否额外练习题目，比如：OJ“计概2025fall每日选做”、CF、LeetCode、洛谷等网站题目。

做了约瑟夫问题的升级版：

M03254:约瑟夫问题No.2

implantation，OpenJudge - M03254:约瑟夫问题No.2

耗时：15分钟速通。

思路：与约瑟夫问题差不多。

代码

```
blacklist = 0
while blacklist == 0:
    n, p, m = list(map(int, input().split()))
    if m + n + p >= 1:
        otto = []
        cnt = 0
        o = p - 1
        shuodedaoli = []
        for i in range(1, n + 1):
            otto.append(i)
        while len(otto) > 1:
            cnt += 1
            o += 1
            if cnt == m:
                cnt = 0
                ot = (o - 1) % len(otto)
                shuodedaoli.append(otto[ot])
                otto.remove(otto[ot])
                o = ot
        shuodedaoli.append(otto[0])
        print(','.join(map(str, shuodedaoli)))
    else:
        blacklist = 1
        break
```

代码运行截图 （至少包含有"Accepted"）

M03254:约瑟夫问题No.2

总时间限制: 1000ms 内存限制: 65536kB

描述

n 个小孩围坐成一圈，并按顺时针编号为1,2,...,n，从编号为 p 的小孩顺时针依次报数，由1报到m ，当报到 m 时，该小孩从圈中出去，然后下一个再从1报数，当报到 m 时再出去。如此反复，直至所有的小孩都从圈中出去。请按出去的先后顺序输出小孩的编号。

输入

每行是用空格分开的三个整数，第一个是n,第二个是p,第三个是m (0 < m,n < 300)。最后一行是:
0 0 0

输出

按出圈的顺序输出编号，编号之间以逗号间隔。

样例输入

```
8 3 4
0 0 0
```

样例输出

```
6,2,7,4,3,5,1,8
```

提示

可以利用模拟方法解题.
与 http://net.pku.edu.cn/~course/cs101/book2/pp_list.txt
中的 poj 2746 Joseph issue极为相似。

来源

cs10107 C++ Final Exam

全局题号 2255
添加于 2025-03-13
提交次数 74
尝试人数 46
通过人数 44

你的提交记录

#	结果	时间
2	Accepted	2025-10-23
1	Runtime Error	2025-10-23

又做了一些简单题练手。

自学了def函数的用法以及面向对象的编程方法（LeetCode上面几乎每一题前两行都是class和def）

为解决问题引入的定义：

```
def to_listie(n_to_listie):
    return [i_to_listie for i_to_listie in range(n_to_listie)]


def str_to_listie(n_to_listie):
    return [str(i_to_listie) for i_to_listie in range(n_to_listie)]


def to_str(listy):
    return[str(i_in_listy) for i_in_listy in listy]


def cata_findm(n0, m0):
    if m0 == 1:
        return 0
    else:
        return min(n0 + 1, m0)
```

```python
def alter_listie(listy):
    first = listy.pop(0)
    listy.append(first)
    return listy


def kpj(strry_listy):
    return ' '.join(strry_listy)


def mounti(listie):
    mount_altitude = [0]
    for i in range(1, len(listie)):
        if listie[i] > listie[i - 1]:
            mount_altitude.append(mount_altitude[-1] + 1)
        elif listie[i] < listie[i - 1]:
            mount_altitude.append(mount_altitude[-1] - 1)
        else:
            mount_altitude.append(mount_altitude[-1])
    base_altitude = min(mount_altitude) - 1
    mount_waterfill = -len(listie) * base_altitude
    mount_seasum = sum(mount_altitude)
    mount_landsum = mount_seasum + mount_waterfill
    return mount_landsum


def hillgravedown(listy):
    hillgravedown_n = len(listy)
    hilldown_left = [0] * hillgravedown_n
    for i in range(hillgravedown_n):
        if i > 0 and listy[i] > listy[i - 1]:
            hilldown_left[i] = hilldown_left[i - 1] + 1
        else:
            hilldown_left[i] = 1

    hilldown_right = hillgravedown_dust = 0
    for i in range(hillgravedown_n - 1, -1, -1):
        if i < hillgravedown_n - 1 and listy[i] > listy[i + 1]:
            hilldown_right += 1
        else:
            hilldown_right = 1
        hillgravedown_dust += max(hilldown_left[i], hilldown_right)

    return hillgravedown_dust
```

有的函数可以简化输入（如kpj可以将列表转化为带空格的一串数字输出，常用且好用）

有的可以简化逻辑，避免模块需要逐个排查的问题（mounti就在某些题目中有类似用法）

还有的因题而异（cata_findm就是针对特定的题目而非题目类型定义的）

一般函数名越长，意味着使用得可能越少（如mounti的使用频率要显著高于hillgravedown）