

# Assignment #6: 矩阵、贪心

Updated 1432 GMT+8 Oct 14, 2025

2025 fall, Compiled by 郭旭杰、化学与分子工程学院

注: OpenJudge用户名: 25n2500011906 / 郭旭杰(10月17日及以前), 昵称: 郭旭杰

CodeForces用户名: LittleBeetroot

## 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

### M18211: 军备竞赛

greedy, two pointers, <http://cs101.openjudge.cn/pctbook/M18211>

耗时: 20min通过

思路: 基础的双指针问题, 需要贪心算法。

首先判定鸣人是否能买得起最便宜的武器, 如果买不起直接输出0。如果买得起最便宜的武器, 鸣人先按价格由低到高尽可能多地买入便宜的武器, 然后反复执行卖掉最贵的一种武器并用所得的钱按照价格由低到高依次买入最便宜的武器, 直到剩下0或1种武器终止, 即可得到正确结果。

代码

```
# Python
# two pointers, greedy.
p = int(input())
js = list(map(int, input().split()))
js.sort()
jsr = []
for _ in range(1, len(js)+1):
```

```
jsr.append(js[-_])

lt = len(js)
if p < js[0]:
    print(0)
else:
    a = b = 0
    a += 1
    p -= js[0]
    dta = 1
    while a + b < lt:
        while p >= js[a]:
            p -= js[a]
            a += 1
            dta += 1
        if jsr[b] >= js[a]:
            p += jsr[b] - js[a]
            a += 1
            b += 1

    print(dta)
```

代码运行截图 (至少包含有"Accepted")



M18211:军备竞赛

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

描述

鸣人是木叶村的村长，最近在跟敌国进行军备竞赛，他手边有N份武器设计图，每张设计图有制作成本（大于等于零）且最多使用一次，可以选择花钱制作或是以同样的价钱卖给敌国，同时任意时刻敌国的武器不能比我国更多，鸣人的目标是在不负债的前提下武器种类比敌国越多越好。

输入

第一行为起始整数经费p,并且0≤p。且要求任何时刻p不能小于0。  
第二行为n个整数，以空格分隔，并且0≤每个整数。代表每张设计图的制作成本，同时也是卖价，最多用一次(无法又制作又卖)。

输出

一个整数，代表武器种类最多比敌国多多少。

样例输入

```
Sample1 Input:
10
20 30 40

Sample1 Output:
0

解释：10元不足以制作20元的武器，所以为0，也不能先卖50元的，不能让敌国武器比木叶多
```

全局题号 18211  
添加于 2025-03-13  
提交次数 316  
尝试人数 113  
通过人数 104

你的提交记录

#	结果	时间
1	Accepted	2025-10-15

## #50367032提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
p = int(input())
js = list(map(int, input().split()))
js.sort()
jsr = []
for _ in range(1, len(js)+1):
    jsr.append(js[-_])

lt = len(js)
if p < js[0]:
    print(0)
else:
    a = b = 0
    a += 1
    p -= js[0]
    dta = 1
    while a + b < lt:
        while p >= js[a]:
            p -= js[a]
            a += 1
            dta += 1
        if jsr[b] >= js[a]:
            p += jsr[b] - js[a]
            a += 1
            b += 1

    print(dta)
```

基本信息

#: 50367032  
题目: M18211  
提交人: 郭旭杰  
内存: 3620kB  
时间: 24ms  
语言: Python3  
提交时间: 2025-10-15 03:06:02

## M21554: 排队做实验

greedy, <http://cs101.openjudge.cn/pctbook/M21554/>

耗时: 1h左右才通过。

思路: 取材于经典数学问题("水桶打水最优策略"), 这种**先处理小问题**的策略在经济管理上有广泛应用。

需要灵活运用二维数组。考虑到可能有两个人做实验会用一样的时间(样例输入1), 直接排除Hash Table, 确定本题应使用二维数组存储数据并对其进行排序。

主要在细节方面, 如何正确使用函数以正确输出二维数组是一个难点。要使用`print("".join(map(str, js1)))`, 我先是没使用`str`, 后又弄丢了`map()`, 最后发现数字连在一起, 原来是我把" "打成了"", 最后终于正确输出了第一行。

计算平均等待时间就容易地多了, 只不过要注意`sgm`和`su`的前后顺序, 否则错都不知道怎么错的(这种小问题已经多次让我一个钟头调试代码百思不得其解了, 现在我有了一定的这个方面的经验, WA的时候记得检查代码顺序)。

代码

```
# Python
# math, greedy.
```

```

n = int(input())
js = list(map(int, input().split()))
js1 = []
sq = []
for i in range(n):
    sq.append([js[i],i])
sq.sort()

for i in range(n):
    js1.append(sq[i][1]+1)
print(" ".join(map(str,js1)))

sgm = 0
su = 0
js.sort()
for i in range(n):
    sgm += su
    su += js[i]
k_out = sgm / n
print("%.2f" % k_out)

```

代码运行截图 (至少包含有"Accepted")

OpenJudge
题目ID, 标题, 描述
郭旭杰 信箱 账号

**CS101 / 计算思维算法实践**

[题目](#)
[排名](#)
[状态](#)
[提问](#)

### M21554:排队做实验

查看
提交
统计
提问

总时间限制: 1000ms 内存限制: 65536kB

**描述**

某学院的学生都需要在某月的1号到2号期间完成课程实验，他们每个人的实验需要持续不同的时长。而实验室管理员要安排这些学生，按照一定顺序，在1号到2号期间全部完成实验。假设该学院的实验室同时只能容纳一名学生实验，而这些学生都非常积极，都希望被排在1号的第一个尽快完成实验。

假设该学院有n个学生，实验室管理员收到了n个学生每位需要占用实验室的时长T1,T2,...,Tn，由于学生发送预约邮件的时间比较接近，没办法完全按照先到先得的办法给学生分配实验时间（实验时间相同的话，先到先得）。管理员很犯愁，他希望能有一种能让所有学生平均等待时间尽可能小的顺序，来安排这n位同学的实验时间，请问你能帮他吗？

**输入**

输入为2行  
第一行为n，为学生人数(n<=1000)  
第二行分别表示第一位学生到第n位学生的实验时长T1，T2，...，Tn，每个数据间有一个空格

**输出**

输出为2行  
第一行为一种学生实验顺序，即1到n的一种排列  
第二行为这种方案下的平均等待时间（精确到小数点后两位）

**样例输入**

```

Sample Input:
10
81 365 72 99 22 7 444 203 1024 203

Sample Output:
6 5 3 1 4 8 10 2 7 9
431.90

```

**样例输出**

全局题号 **21554**  
添加于 **2025-03-13**  
提交次数 **169**  
尝试人数 **103**  
通过人数 **102**  
  
Other language versions  
English  
  
你的提交记录

#	结果	时间
2	Accepted	2025-10-16
1	Compile Error	2025-10-16



## #50387477提交状态

查看

提交

统计

提问

状态: Accepted

## 源代码

```
n = int(input())
js = list(map(int, input().split()))
js1 = []
sq = []
for i in range(n):
    sq.append([js[i], i])
sq.sort()

for i in range(n):
    js1.append(sq[i][1]+1)
print(" ".join(map(str, js1)))

sgm = 0
su = 0
js.sort()
for i in range(n):
    sgm += su
    su += js[i]
k_out = sgm / n
print("%.2f" % k_out)
```

## 基本信息

#: 50387477

题目: M21554

提交人: 郭旭杰

内存: 3616kB

时间: 21ms

语言: Python3

提交时间: 2025-10-16 00:14:23

## E23555: 节省存储的矩阵乘法

implementation, matrices, <http://cs101.openjudge.cn/pctbook/E23555>

思路: 较难, 需要强大的数学基础。

注意不要使用连等号。

个人认为应该归于M类题目。

代码

```
#Python
#implementation, matrices

n,m1,m2 = map(int,input().split())
A = [[0 for x1 in range(n)] for y1 in range(n)]
B = [[0 for x2 in range(n)] for y2 in range(n)]
C = [[0 for x3 in range(n)] for y3 in range(n)]
for i in range(m1):
    a,b,k = map(int,input().split())
    A[a][b] = k
for i in range(m2):
    a,b,k = map(int,input().split())
    B[a][b] = k
for i1 in range(n):
```

代码运行截图 (至少包含有"Accepted")

查看	提交	统计	提问			
全局题号	23555					
添加于	2025-10-22					
提交次数	252					
尝试人数	151					
通过人数	151					
Other language versions						
English						
你的提交记录						
#	结果	时间				
2	Accepted	2025-10-16				
	Time Limit					
1	Exceeded	2025-10-15				

## #50397301提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
n,m1,m2 = map(int,input().split())
A = [[0 for x1 in range(n)] for y1 in range(n)]
B = [[0 for x2 in range(n)] for y2 in range(n)]
C = [[0 for x3 in range(n)] for y3 in range(n)]
for i in range(m1):
    a,b,k = map(int,input().split())
    A[a][b] = k
for i in range(m2):
    a,b,k = map(int,input().split())
    B[a][b] = k
for i1 in range(n):
    for j1 in range(n):
        res = 0
        for t in range(n):
            res += A[i1][t] * B[t][j1]
        C[i1][j1] = res

for a1 in range(n):
    for b1 in range(n):
        if C[a1][b1] != 0:
            print(a1,b1,C[a1][b1])
```

基本信息

#: 50397301

题目: E23555

提交人: 25n2500011906

内存: 3736kB

时间: 27ms

语言: Python3

提交时间: 2025-10-16 16:56:49

## M12558: 岛屿周长

matrices, <http://cs101.openjudge.cn/pctbook/M12558>

耗时: 10min速通

思路: 这个题目思路非常简单, 小学的数学内容, 只需要基础的矩阵知识, 然后暴力运算求解, 只不过代码较长一些而已。

个人认为应该归于E类题目。

代码

```
# Python
# matrix, brute force.
n, m = map(int, input().split())
matrix = []
for i in range(n):
    row = list(map(int, input().split()))
    matrix.append(row)

C = 0
for i in range(n):
    for j in range(m):
        if matrix[i][j] == 1:
            C += 4
```

```
for i in range(n):
    for j in range(m-1):
        if matrix[i][j] + matrix[i][j+1] == 2:
            c -= 2

for j in range(m):
    for i in range(n-1):
        if matrix[i+1][j] == 2:
            c -= 2

print(c)
```

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号



CS101 / 计算思维算法实践

题目

排名

状态

提问

M12558:岛屿周长

查看

提交

统计

提问

总时间限制: 1000ms    内存限制: 65536kB

描述

用一个n\*m的二维数组表示地图，1表示陆地，0代表海水，每一格都表示一个1\*1的区域。地图中的格子只能横向或者纵向连接（不能对角连接），连接在一起的陆地称作岛屿，同时整个地图都被海水围绕。假设给出的地图中只会会有一个岛屿，并且岛屿中不会有湖（即不会有水被陆地包围的情况出现）。请判断所给定的二维地图中岛屿的周长。

输入

第一行为n和m，表示地图的大小(1<=n<=100, 1<=m<=100)。接下来n行，每行有m个数，分别描述每一格的数值。数值之间均用空格隔开。

输出

只有一行，即岛屿的周长（正整数）。

样例输入

```
3 4
1 1 1 0
0 1 0 0
1 1 0 0
```

样例输出

```
14
```

提示

matrix, dfs

来源

全局题号 12558

添加于 2025-03-13

提交次数 138

尝试人数 98

通过人数 98

你的提交记录

#	结果	时间
1	Accepted	2025-10-16

## #50387850提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

### 源代码

```
n, m = map(int, input().split())
matrix = []
for i in range(n):
    row = list(map(int, input().split()))
    matrix.append(row)

C = 0
for i in range(n):
    for j in range(m):
        if matrix[i][j] == 1:
            C += 4

for i in range(n):
    for j in range(m-1):
        if matrix[i][j] + matrix[i][j+1] == 2:
            C -= 2

for j in range(m):
    for i in range(n-1):
        if matrix[i][j] + matrix[i+1][j] == 2:
            C -= 2

print(C)
```

### 基本信息

#: 50387850  
题目: M12558  
提交人: 郭旭杰  
内存: 3692kB  
时间: 23ms  
语言: Python3  
提交时间: 2025-10-16 01:21:22

## M01328: Radar Installation

greedy, <http://cs101.openjudge.cn/practice/01328/>

耗时: 前后3天, 每天改一会儿代码。

思路: 一开始我想要按照y(岛离海岸线的距离)从大到小进行讨论(主要受先讨论y过大时输出-1的情况), 结果一个晚上毫无头绪; 后来又看到同学讨论使用x平移思路来解决问题, def了一个函数(递归的函数), 结果屡次爆栈(RE)。为解决爆栈尝试调高栈上限, 结果MLE; 递归换循环, 结果TLE; 中间还出现了几次CE和WA, 错误类型除了PE都经历了一遍, 还有两次使用AI做到AC的。最后老师建议我减少循环层数以降低时间复杂度, 我苦思冥想不得其解, 然后对着AI生成的几行代码看了有十几分钟才看出个所以然来, 当即改变代码整体思路, 改换了参考系又除去了一层循环, 终于成功。

```

32     count = 0
33     last_radar = -float('inf')
34     for l, r in intervals:
35         if l > last_radar:
36             count += 1
37             last_radar = r

```

雷达问题的代码，AI写的这几行简直绝妙👍，成功地减少了循环，我看了好久才看懂🤔。

这个使用了参考系变换，就是我们正常以海岸为参考系平移雷达，但是这个代码把每个岛分别作为参考系，这一点我一直没想到。

代码

```

#Python
#math, greedy
#我最终AC时用的代码

## 用null = input()实现空行
## 贪心
## 坐标系变换

from math import sqrt

def yuntin2x(x,y,r0):
    if y <= r0:
        return x - sqrt(r0 ** 2 - y ** 2), x + sqrt(r0 ** 2 - y ** 2)
    else:
        return ()

n = 1
d = 1
cnt = 0
blacklist = 0
res = 0

while blacklist == 0:
    cnt += 1
    n, d = map(int, input().split())
    if n == 0 and d == 0:
        blacklist = 1
        break
    ghets = []
    for _ in range(n):
        res = 0
        blacka = 0
        a, b = map(int, input().split())

```

```

        if b <= d:
            ghets.append(yuntin2x(a, b, d))
        else:
            blacka += 1
            res = -1
null = input()

rrest = -float('inf')
if res == -1:
    null = 0
else:
    ghets.sort(key=lambda x: x[1])
    for ip in ghets:
        if ip[0] > rrest:
            res += 1
            rrest = ip[1]

print("Case "+str(cnt)+" : "+str(res))

```

#=====

#AI(腾讯元宝)一开始生成的代码

```

import math

test_case = 0

while True:
    # 读取n和d, 跳过可能的空行
    line = input().strip()
    while line == '':
        line = input().strip()
    n, d = map(int, line.split())
    if n == 0 and d == 0:
        break
    test_case += 1

    impossible = False
    intervals = []
    for _ in range(n):
        # 读取岛屿坐标, 每个岛屿坐标占一行, 无空行
        x, y = map(int, input().split())
        if y > d:
            impossible = True
        dy = d * d - y * y
        if dy < 0:
            impossible = True
        else:
            sqrt_dy = math.sqrt(dy)
            left = x - sqrt_dy
            right = x + sqrt_dy
            intervals.append((left, right))

    if impossible:
        print(f"Test case {test_case}: -1")

```

```

        continue

# 按区间的右端点升序排序
intervals.sort(key=lambda x: x[1])

count = 0
last_radar = -float('inf')
for l, r in intervals:
    if l > last_radar:
        count += 1
        last_radar = r

print(f"Test case {test_case}: {count}")

#=====

#用递归导致爆栈的代码

#用null = input()实现空行
# 定义平移量yuntun
# 贪心+递归
from math import sqrt

def yuntun(x,y,r):
    return x + sqrt(r ** 2 - y ** 2)
def distance(x1,y1,x2,y2):
    return sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)

def radar(d0,pset0):
    blacklist0 = 0
    for point in pset0:
        if point[1] > d0:
            blacklist0 += 1
    if blacklist0 > 0:
        return -1
    else:
        yuntom = []
        npset0 = []
        for point in pset0:
            yuntom.append(yuntun(point[0],point[1],d0))
        myuntom = min(yuntom)
        for point in pset0:
            if distance(point[0],point[1],myuntom,0) > d0:
                npset0.append(point)
        if len(npset0) == 0:
            return 1
        else:
            return radar(d0,npset0) + 1

n = 1
d = 1
cnt = 0
blacklist = 0

```

```

while blacklist == 0:
    cnt += 1
    n, d = map(int, input().split())
    if n == 0 and d == 0:
        blacklist = 1
        break
    js = []
    for _ in range(n):
        a, b = map(int, input().split())
        js.append([a, b])
    null = input()
    res = str(radar(d, js))
    print("Case "+str(cnt)+": "+res)

```

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 / 题库 (包括计概、数算题目)**

[题目](#)
[排名](#)
[状态](#)
[提问](#)

### 01328:Radar Installation

[查看](#)
[提交](#)
[统计](#)
[提问](#)

总时间限制: 1000ms    内存限制: 65536kB

#### 描述

Assume the coasting is an infinite straight line. Land is in one side of coasting, sea in the other. Each small island is a point locating in the sea side. And any radar installation, locating on the coasting, can only cover d distance, so an island in the sea can be covered by a radius installation, if the distance between them is at most d.

We use Cartesian coordinate system, defining the coasting is the x-axis. The sea side is above x-axis, and the land side below. Given the position of each island in the sea, and given the distance of the coverage of the radar installation, your task is to write a program to find the minimal number of radar installations to cover all the islands. Note that the position of an island is represented by its x-y coordinates.

Figure A Sample Input of Radar Installations

#### 输入

The input consists of several test cases. The first line of each case contains two integers n ( $1 \leq n \leq 1000$ ) and d, where n is the number of islands in the sea and d is the distance of coverage of the radar installation. This is followed by n lines each containing two integers representing the coordinate of the position of each island. Then a blank line follows to separate the cases.

The input is terminated by a line containing pair of zeros

全局题号 **330**  
 添加于 **2020-12-15**  
 提交次数 **2471**  
 尝试人数 **470**  
 通过人数 **437**

#### 你的提交记录

#	结果	时间
62	Accepted	2025-10-18
61	Wrong Answer	2025-10-18
60	Wrong Answer	2025-10-18
59	Wrong Answer	2025-10-18
58	Wrong Answer	2025-10-18
57	Wrong Answer	2025-10-18
56	Runtime Error	2025-10-18
55	Time Limit Exceeded	2025-10-18
54	Accepted	2025-10-18
53	Time Limit Exceeded	2025-10-18
52	Time Limit Exceeded	2025-10-18
51	Time Limit Exceeded	2025-10-18
50	Compile Error	2025-10-18
49	Accepted	2025-10-18
48	Compile Error	2025-10-18
47	Wrong Answer	2025-10-18
46	Time Limit Exceeded	2025-10-18

#50440129提交状态

查看

提交

统计

提问

状态: **Accepted**

源代码

```
# 用null = input() 实现空行
# 贪心
# 坐标系变换

from math import sqrt

def yuntin2x(x,y,r0):
    if y <= r0:
        return x - sqrt(r0 ** 2 - y ** 2), x + sqrt(r0 ** 2 - y ** 2)
    else:
        return ()

n = 1
d = 1
cnt = 0
blacklist = 0
res = 0

while blacklist == 0:
    cnt += 1
    n, d = map(int, input().split())
    if n == 0 and d == 0:
        blacklist = 1
        break
    ghets = []
    for _ in range(n):
        res = 0
        blacka = 0
        a, b = map(int, input().split())
        if b <= d:
            ghets.append(yuntin2x(a, b, d))
        else:
            blacka += 1
            res = -1
    null = input()

    rrest = -float('inf')
```

基本信息

#: 50440129

题目: 01328

提交人: 25n2500011906

内存: 32408kB

时间: 165ms

语言: PyPy3

提交时间: 2025-10-18 22:36:27

源代码

```
# 用null = input()实现空行
# 贪心
# 坐标系变换

from math import sqrt

def yuntin2x(x,y,r0):
    if y <= r0:
        return x - sqrt(r0 ** 2 - y ** 2), x + sqrt(r0 ** 2 - y ** 2)
    else:
        return ()

n = 1
d = 1
cnt = 0
blacklist = 0
res = 0

while blacklist == 0:
    cnt += 1
    n, d = map(int, input().split())
    if n == 0 and d == 0:
        blacklist = 1
        break
    ghets = []
    for _ in range(n):
        res = 0
        blacka = 0
        a, b = map(int, input().split())
        if b <= d:
            ghets.append(yuntin2x(a, b, d))
        else:
            blacka += 1
            res = -1
    null = input()

    rrest = -float('inf')
    if res == -1:
        null = 0
    else:
        ghets.sort(key=lambda x: x[1])
        for ip in ghets:
            if ip[0] > rrest:
                res += 1
                rrest = ip[1]

    print("Case "+str(cnt)+" : "+str(res))
```

#: 50440129

题目: 01328

提交人: 25n2500011906

内存: 32408kB

时间: 165ms

语言: PyPy3

提交时间: 2025-10-18 22:36:27

©2002-2022 POJ 京ICP备20010980号-1

English 帮助

## 545C. Woodcutters

dp, greedy, 1500, <https://codeforces.com/problemset/problem/545/C>

耗时: 前后3天, 每天改一会儿代码。

思路: 简单但很重要的贪心算法, 要想砍树最多, 最左边和最右边的树无论如何都必须倒下, 中间的树如果可以倒下只有向左和向右两种可能。从最左边的树开始, 后续的树尽可能向左边倒, 如果左边不能倒就向右边倒, 不必考虑右边的树(因为这个时候即使右面的树受其影响倒不了, 这棵树的倒下也能抵消其影响), 右边也不能倒就只能立着, 如是进行, 即可得到最终答案。此外, 这一题也是上面M01328: Radar Installation题目的底层逻辑。

屡次提交代码没有通过竟然是因为中间一行少打了一个等于号, 最终发现了这种情况, 也是十分令人警醒。

代码

```

#Python
#greedy
n = int(input())
trees = []
for i in range(n):
    th = list(map(int, input().split()))
    trees.append(th)

if n <= 2:
    cut = n
else:
    cut = 2
    left = trees[0][0]
    for i in range(1, n - 1):
        if trees[i][0] - trees[i][1] > left:
            cut += 1
            left = trees[i][0]
        elif trees[i][0] - trees[i][1] <= left and trees[i][0] + trees[i][1] < trees[i + 1][0]:
            cut += 1
            left = trees[i][0] + trees[i][1]
        else:
            cut += 0
            left = trees[i][0]

print(cut)

```

代码运行截图 (至少包含有"Accepted")

PROBLEMS SUBMIT CODE MY SUBMISSIONS STATUS HACKS ROOM STANDINGS CUSTOM INVOCATION

General									
#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged	
344521871	Practice: LittleBeetroot	545C - 14	PyPy 3-64	Accepted	390 ms	17100 KB	2025-10-18 18:09:13	2025-10-18 18:09:13	★ Compare

→ Source

Copy

```

n = int(input())
trees = []
for i in range(n):
    th = list(map(int, input().split()))
    trees.append(th)

if n <= 2:
    cut = n
else:
    cut = 2
    left = trees[0][0]
    for i in range(1, n - 1):
        if trees[i][0] - trees[i][1] > left:
            cut += 1
            left = trees[i][0]
        elif trees[i][0] - trees[i][1] <= left and trees[i][0] + trees[i][1] < trees[i + 1][0]:
            cut += 1
            left = trees[i][0] + trees[i][1]
        else:
            cut += 0
            left = trees[i][0]

print(cut)

```

[Click](#) to see test details

#	When	Who	Problem	Lang	Verdict	Time	Memory
344521871	Oct/18/2025 23:09UTC+8	LittleBeetroot	C - Woodcutters	PyPy 3-64	Accepted	390 ms	17100 KB
344519270	Oct/18/2025 22:54UTC+8	LittleBeetroot	C - Woodcutters	PyPy 3-64	Wrong answer on test 30	312 ms	17000 KB
344518522	Oct/18/2025 22:49UTC+8	LittleBeetroot	C - Woodcutters	PyPy 3-64	Wrong answer on test 30	375 ms	17000 KB
344518296	Oct/18/2025 22:48UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 30	421 ms	20000 KB
344081533	Oct/17/2025 02:41UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 4	390 ms	20500 KB
344081316	Oct/17/2025 02:39UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 45	343 ms	20600 KB
344081125	Oct/17/2025 02:38UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 44	359 ms	20500 KB
344080663	Oct/17/2025 02:34UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 43	311 ms	20400 KB
344080547	Oct/17/2025 02:33UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 43	359 ms	20600 KB
344080370	Oct/17/2025 02:32UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 43	296 ms	20500 KB
344079628	Oct/17/2025 02:27UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 42	327 ms	20600 KB
344078553	Oct/17/2025 02:26UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 41	374 ms	20700 KB
344078347	Oct/17/2025 02:19UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 40	328 ms	20600 KB
344078107	Oct/17/2025 02:18UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 40	343 ms	20400 KB
344077878	Oct/17/2025 02:16UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 39	311 ms	20400 KB
344077754	Oct/17/2025 02:16UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 40	343 ms	20400 KB
344077593	Oct/17/2025 02:15UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 40	312 ms	20500 KB
344076896	Oct/17/2025 02:10UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 39	343 ms	20400 KB
344076702	Oct/17/2025 02:09UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 38	296 ms	20300 KB
344076566	Oct/17/2025 02:08UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 37	374 ms	20400 KB
344076461	Oct/17/2025 02:07UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 36	281 ms	20400 KB
344076312	Oct/17/2025 02:06UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 35	312 ms	20400 KB
344076181	Oct/17/2025 02:06UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 34	327 ms	20300 KB
344076023	Oct/17/2025 02:05UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 33	327 ms	20400 KB
344075762	Oct/17/2025 02:03UTC+8	LittleBeetroot	C - Woodcutters	Python 3	Wrong answer on test 32	327 ms	20400 KB

Codeforces Round 303 (Div. 2)

Finished

Practice



Virtual participation

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

Start virtual contest

Clone Contest to Mashup

You can clone this contest to a mashup.

Clone Contest

Contest materials

- Announcement (en)
- Tutorial (en)

Accepted

By LittleBeetroot, contest: Codeforces Round 303 (Div. 2), problem: (C) Woodcutters, Accepted, #, Copy

```
n = int(input())
trees = []
for i in range(n):
    th = list(map(int, input().split()))
    trees.append(th)

if n <= 2:
    cut = n
else:
    cut = 2
    left = trees[0][0]
    for i in range(1, n - 1):
        if trees[i][0] - trees[i][1] > left:
            cut += 1
            left = trees[i][0]
        elif trees[i][0] - trees[i][1] <= left and trees[i][0] + trees[i][1] < trees[i + 1][0]:
            cut += 1
            left = trees[i][0] + trees[i][1]
    else:
        cut += 0
        left = trees[i][0]

print(cut)
```

## 2. 学习总结和收获

这周重点学习的内容是矩阵(二维数组), 掌握矩阵的输入、输出、变换、排序等(18161: 矩阵运算(先乘后加),E23555: 节省存储的矩阵乘法,M12558: 岛屿周长,545A.Toy Cars), 兼顾双指针(M18211: 军备竞赛)和贪心算法(M18211: 军备竞赛,M21554: 排队做实验,545C. Woodcutters,M01328: Radar Installation)。

利用二维数组解其他问题(M21554: 排队做实验)

矩阵的优化(E23555: 节省存储的矩阵乘法)

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。

要想完成E23555，必须先完成18161：矩阵运算（先乘后加）

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号



CS101 / 题库（包括计概、数算题目）

题目

排名

状态

提问

18161:矩阵运算(先乘再加)

查看

提交

统计

提问

总时间限制: 1000ms 内存限制: 65536kB

描述

现有三个矩阵A, B, C, 要求矩阵运算A·B+C并输出结果

矩阵运算介绍：  
矩阵乘法运算必须要前一个矩阵的列数与后一个矩阵的行数相同，  
如m行n列的矩阵A与n行p列的矩阵B相乘，可以得到m行p列的矩阵C，  
矩阵C的每个元素都由A的对应行中的元素与B的对应列中的元素——相乘并求和得到，  
即 $C[i][j] = A[i][0]*B[0][j] + A[i][1]*B[1][j] + ..... + A[i][n-1]*B[n-1][j]$   
( $C[i][j]$ 表示C矩阵中第i行第j列元素)。  
  
矩阵的加法必须在两个矩阵行数列数均相等的情况下进行，  
如m行n列的矩阵A与m行n列的矩阵B相加，可以得到m行n列的矩阵C，  
矩阵C的每个元素都由A与B对应位置的元素相加得到，  
即 $C[i][j] = A[i][j] + B[i][j]$

输入

输入分为三部分，分别是A,B,C三个矩阵的内容。  
每一部分的第一行为两个整数，代表矩阵的行数row和列数col  
接下来row行，每行有col个整数，代表该矩阵这一行的每个元素

输出

如果可以完成矩阵计算，输出计算结果，与输入格式类似，不需要输出行数和列数信息。  
如果不能完成矩阵计算，输出"Error!"

样例输入

Sample Input1:

3 1  
0  
1

全局题号 18161

添加于 2018-12-26

提交次数 1570

尝试人数 764

通过人数 737

你的提交记录

#	结果	时间
4	Accepted	2025-10-15
3	Wrong Answer	2025-10-15
2	Wrong Answer	2025-10-15
1	Wrong Answer	2025-10-15

## #50367019提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

### 源代码

```
m1, n1 = map(int, input().split())
A = []
for i1 in range(m1):
    row1 = list(map(int, input().split()))
    A.append(row1)

n2, p2 = map(int, input().split())
B = []
for i2 in range(n2):
    row2 = list(map(int, input().split()))
    B.append(row2)

m3, p3 = map(int, input().split())
C = []
for i3 in range(m3):
    row3 = list(map(int, input().split()))
    C.append(row3)

if n1 == n2 and p2 == p3 and m1 == m3:
    D = []
    m4 = m3
    p4 = p3
    for i4 in range(m4):
        D.append([])
    for i in range(m3):
        for j in range(p3):
            re = C[i][j]
            for t in range(n2):
                re += A[i][t] * B[t][j]
            D[i].append(re)
        for new_row in D:
            print(' '.join(map(str, new_row)))
    else:
        print("Error!")
else:
    print("Error!")
```

### 基本信息

#: 50367019  
题目: 18161  
提交人: 郭旭杰  
内存: 4048kB  
时间: 93ms  
语言: Python3  
提交时间: 2025-10-15 02:15:56

Codeforces参加Div3的比赛，发现很多题有思路就特别轻松，但是问题在于我自己做题的时候没有思路，因此成绩很不理想。(OJ上面的题目经常有思路但语法上略有欠缺，CF上面的题目更多是语法会但没有思路)。

最后其他同学的思路令我大吃一惊。

Contests

Only rated ▾

# ▴	Contest	Start time	Rank ▴	Solved ▴	Rating change ▴	New rating ▴	
4	<a href="#">Codeforces Round 1059 (Div. 3)</a>	<a href="#">Oct/17/2025 22:35UTC+8</a>	<a href="#">15224</a>	<a href="#">2</a>	<b>+82</b>	982	
3	<a href="#">Codeforces Round 1057 (Div. 2)</a>	<a href="#">Oct/10/2025 22:35UTC+8</a>	<a href="#">9164</a>	<a href="#">2</a>	<b>+220</b>	900	
2	<a href="#">Codeforces Round 1056 (Div. 2)</a>	<a href="#">Oct/06/2025 00:35UTC+8</a>	<a href="#">8628</a>	<a href="#">1</a>	<b>+277</b>	680	
1	<a href="#">Squarepoint Challenge (Codeforces Round 1055, Div. 1 + Div. 2)</a>	<a href="#">Oct/03/2025 22:35UTC+8</a>	<a href="#">10709</a>	<a href="#">1</a>	<b>+403</b>	403	LittleBeetroot → LittleBeetroot

做了Codeforces上面的545A.Toy Cars这个矩阵有关的题目。

#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged		
344523799	Practice: LittleBeetroot	<a href="#">545A</a> - 8	PyPy 3-64	Accepted	108 ms	1632 KB	2025-10-18 18:21:12	2025-10-18 18:21:12	★	<button>Compare</button>

→ Source Copy

```
n = int(input())
matrix = []
car = [_ for _ in range(n)]
loss = []
for c in range(n):
    row = list(map(int, input().split()))
    matrix.append(row)

for i in range(n):
    for j in range(n):
        if matrix[i][j] == 1:
            loss.append(i)
        elif matrix[i][j] == 2:
            loss.append(j)
        elif matrix[i][j] == 3:
            loss.append(i)
            loss.append(j)
        else:
            null = 0

goodcar = []
cnt = 0
for i in range(n):
    if i in car and i not in loss:
        goodcar.append(i+1)
        cnt += 1

print(len(goodcar))
print(" ".join(map(str, goodcar)))
```

[Click](#) to see test details

我用矩阵给自己出了一道题("小行星"), Codeforce风格, 类似于小游戏的那种, 可以用来巩固基础的矩阵知识(很多题目都是由这个结合其他知识构成的)。就是用矩阵实现上下连通和左右连通。

#在一个小行星上(抽象为一个m行n列的矩阵)有若干棵树, 矩阵中每个数字对应相应位置树的个数。由于小行星是球体, 矩阵上下连通, 左右连通(就是如果从矩阵最上面一行向上会走到最下面一行, 依此类推)。初始位置在第a行, 第b列(从0开始算起)。要求用数字键盘输入一个数可以执行相应操作, 除了5可以输出当前的位置和矩阵的情况, 其他数字分别代表向相应方向移动(例如8: 向上移动; 1: 向左下移动。输入+表示在当前位置种一棵树, 输入-表示在当前位置砍一棵树, 如果没有树就输出“No trees”。输入0结束, 输出最终矩阵的情况。输入其他字符在原地不动, 输出“You are staying.”要求每次输入之后都要输出当前的位置和此位置树的棵数。最终结束时输出"End"。

代码如下:

```
# Python
# matrices
# "小行星"模型(常见类似问题)
```

```

planet = []
n,m = map(int,input().split())
for i in range(m):
    row = list(map(int,input().split()))
    planet.append(row)
a,b = list(map(int,input().split()))
blacklist = 0
while blacklist == 0:
    stt = input().strip()
    if stt == "1":
        if a <= n - 2:
            a += 1
        else:
            a = 0
        if b >= 1:
            b -= 1
        else:
            b = m - 1
    elif stt == "2":
        if a <= n - 2:
            a += 1
        else:
            a = 0
    elif stt == "3":
        if a <= n - 2:
            a += 1
        else:
            a = 0
        if b <= m - 2:
            b += 1
        else:
            b = 0
    elif stt == "4":
        if b >= 1:
            b -= 1
        else:
            b = m - 1
    elif stt == "5":
        for new_row in planet:
            print(' '.join(map(str, new_row)))
    elif stt == "6":
        if b <= m - 2:
            b += 1
        else:
            b = 0
    elif stt == "7":
        if a >= 1:
            a -= 1
        else:
            a = n - 1
        if b >= 1:
            b -= 1
        else:

```

```

        b = m - 1
elif stt == "8":
    if a >= 1:
        a -= 1
    else:
        a = n - 1
elif stt == "9":
    if a >= 1:
        a -= 1
    else:
        a = n - 1
    if b <= m - 2:
        b += 1
    else:
        b = 0
elif stt == "-":
    if planet[a][b] >= 1:
        planet[a][b] = planet[a][b] - 1
    else:
        print("No trees")
elif stt == "+":
    planet[a][b] = planet[a][b] + 1
elif stt == "0":
    blacklist = 1
    for new_row in planet:
        print(' '.join(map(str, new_row)))
else:
    print("You are staying.")
print(a,b,planet[a][b])
if stt == "0":
    print("End")

```