

Assignment #6: 矩阵、贪心

Updated 1432 GMT+8 Oct 14, 2025

2025 fall, Compiled by 韩旭 元培学院

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>

用时: 20min

思路

这个题有两个方面的贪心，因此需要用到双指针：对于自己而言，优先制作成本低的武器；对于对方而言，则优先卖出价格高的武器。这里面需要特别注意卖出的条件：首先，卖出后对方不能比自己多（即 `ours > oppo`）；其次，卖出后自己还有剩余武器可以买（即 `left < right`）；最后，卖出后赚的钱足够至少买一个武器（由排序保证）。这样，卖出后我方与对方的武器差至少不会减小，从而卖出是有利可图的。当所有武器都被制作或卖出，或者没钱制作也没必要卖出的时候，停止循环。

代码

```
money=int(input())
maps=list(map(int,input().split()))
maps.sort()
```

```

left=0
right=len(maps)-1
ours=0 #Number of our weapon
oppo=0 #Number of opponent's weapon
while left<=right and left<len(maps) and abs(right)>=0:
    if maps[left] <= money: #Make
        money-=maps[left]
        ours+=1
        left+=1
    else:
        if ours>oppo and left<right: #Sell
            money+=maps[right]
            oppo+=1
            right-=1
        else:
            break
print(ours-oppo)

```

代码运行截图

#50433868提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

money=int(input())
maps=list(map(int,input().split()))
maps.sort()
left=0
right=len(maps)-1
ours=0 #Number of our weapon
oppo=0 #Number of opponent's weapon
while left<=right and left<len(maps) and abs(right)>=0:
    if maps[left] <= money: #Make
        money-=maps[left]
        ours+=1
        left+=1
    else:
        if ours>oppo and left<right: #Sell
            money+=maps[right]
            oppo+=1
            right-=1
        else:
            break
print(ours-oppo)

```

基本信息

#: 50433868
 题目: M18211
 提交人: 22n2200017737
 内存: 3620kB
 时间: 27ms
 语言: Python3
 提交时间: 2025-10-18 17:08:41

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

M21554: 排队做实验

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

用时: 10min

思路

这道题的思路就是直接按时间排序，因为越靠前的学生，其作用于平均等待时间的乘数就越大。难点在于排序的时候要连着index一块排序，所以需要先使用 `enumerate()` 函数生成index，然后设置 `key=lambda x:x[1]` 来带着index一块排序，最后再分别生成列表即可。要注意 `.sort()` 函数本身就是**保值排序**，可以满足相同时间先到先得的原则。另外，`.join()` 函数必须是字符串列表才能拼接。

代码

```
N=int(input())
times=list(map(int,input().split()))
id_times=list(enumerate(times))
id_times.sort(key=lambda x:x[1])
ids=[str(i[0]+1) for i in id_times]
times=[i[1] for i in id_times]
print(" ".join(ids))
avg=0
for i,t in enumerate(times):
    avg+=(N-i-1)*t
avg/=N
print(f"{avg:.2f}")
```

代码运行截图

#50434325提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
N=int(input())
times=list(map(int,input().split()))
id_times=list(enumerate(times))
id_times.sort(key=lambda x:x[1])
ids=[str(i[0]+1) for i in id_times]
times=[i[1] for i in id_times]
print(" ".join(ids))
avg=0
for i,t in enumerate(times):
    avg+=(N-i-1)*t
avg/=N
print(f"{avg:.2f}")
```

基本信息

#: 50434325
题目: M21554
提交人: 22n2200017737
内存: 3632kB
时间: 20ms
语言: Python3
提交时间: 2025-10-18 17:29:13

E23555: 节省存储的矩阵乘法

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

用时: 10min

思路

将两个矩阵中的非零元素按其所在行列存储在字典中。然后按标准的矩阵乘法进行计算，在需要元素值的时候使用 `.get()` 函数在字典中检索，并且将default值设置为零即可。最后输出结果中的非零元素及其所在行列。

代码

```
n,m1,m2=map(int,input().split())
A1={}
A2={}
for _ in range(m1):
    r,c,a=map(int,input().split())
    A1[(r,c)]=a
for _ in range(m2):
    r,c,a=map(int,input().split())
    A2[(r,c)]=a
for i in range(n): #row
    for j in range(n): #column
        ans=0
        for k in range(n):
            ans+=A1.get((i,k),0)*A2.get((k,j),0)
        if ans:
            print(f"{i} {j} {ans}")
```

代码运行截图

#50450555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())
A1={}
A2={}
for _ in range(m1):
    r,c,a=map(int,input().split())
    A1[(r,c)]=a
for _ in range(m2):
    r,c,a=map(int,input().split())
    A2[(r,c)]=a
for i in range(n): #row
    for j in range(n): #column
        ans=0
        for k in range(n):
            ans+=A1.get((i,k),0)*A2.get((k,j),0)
        if ans:
            print(f"{i} {j} {ans}")
```

基本信息

#: 50450555
题目: E23555
提交人: 22n2200017737
内存: 3644kB
时间: 30ms
语言: Python3
提交时间: 2025-10-19 15:25:45

M12558: 岛屿周长

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

用时: 10min

思路

每个陆地区域贡献的周长初始值为4，且其上下左右的陆地区域贡献的周长减1。最后将所有区域贡献的周长相加即可。

代码

```
n,m=map(int,input().split())
Map=[]
for _ in range(n):
    row=list(map(int,input().split()))
    Map.append(row)
ans=[[4]*m for _ in range(n)]
for i in range(n):
    for j in range(m):
        if Map[i][j]==0:
            ans[i][j]=0
        else:
            if i-1>=0 and ans[i-1][j]>0:
                ans[i-1][j]-=1
            if i+1<n and ans[i+1][j]>0:
                ans[i+1][j]-=1
            if j-1>=0 and ans[i][j-1]>0:
                ans[i][j-1]-=1
            if j+1<m and ans[i][j+1]>0:
                ans[i][j+1]-=1
print(sum(map(sum,ans)))
```

代码运行截图

状态: **Accepted**

源代码

```
n,m=map(int,input().split())
Map=[]
for _ in range(n):
    row=list(map(int,input().split()))
    Map.append(row)
ans=[[4]*m for _ in range(n)]
for i in range(n):
    for j in range(m):
        if Map[i][j]==0:
            ans[i][j]=0
        else:
            if i-1>=0 and ans[i-1][j]>0:
                ans[i-1][j]-=1
            if i+1<n and ans[i+1][j]>0:
                ans[i+1][j]-=1
            if j-1>=0 and ans[i][j-1]>0:
                ans[i][j-1]-=1
            if j+1<m and ans[i][j+1]>0:
                ans[i][j+1]-=1
print(sum(map(sum,ans)))
```

基本信息

#: 50454326
题目: M12558
提交人: 22n2200017737
内存: 3680kB
时间: 23ms
语言: Python3
提交时间: 2025-10-19 18:02:03

M01328: Radar Installation

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

用时: 30min

思路

这道题需要逆向思维，对于每个岛而言，计算能覆盖到它的雷达在x轴上的区间。然后用贪心考虑这些区间最少可以用几个雷达来覆盖（思路有点像月考校门外的树）：首先按区间左侧排序，然后看第二个区间是不是和第一个有交集，如果有则求出其交集，然后考虑第三个区间。由于已经排序，在取交集的时候只需要更新区间右侧的值即可。如果新的区间和原来的区间交集没有新的交集，则以这个区间为起点重新考虑，同时需要的雷达数目+1。需要注意的是，如果某个岛的y坐标大于d，则这个题无解。这时候不能直接把输入流 `break` 掉，不然会出错，而是应该先标记 `ans=-1`，在输入都完成后再检查即可。

代码

```
import math
count=0
while True:
    n,d=map(int,input().split())
    if n==0 and d==0:
        break
    else:
        count+=1
        ans=0
```

```

ranges=[]
for _ in range(n):
    x,y=map(int,input().split())
    if y>d: #No solution
        ans=-1
    else:
        left=x-math.sqrt(d**2-y**2)
        right=x+math.sqrt(d**2-y**2)
        ranges.append((left,right))
if ans>=0:
    ranges.sort(key=lambda x:x[0])
    Right=ranges[0][0]-1
    for k in ranges:
        if k[0]>Right:
            ans+=1
            Right=k[1]
        else:
            Right=min(Right,k[1])
print(f"Case {count}: {ans}")
input()

```

代码运行截图

#50454934提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

import math
count=0
while True:
    n,d=map(int,input().split())
    if n==0 and d==0:
        break
    else:
        count+=1
        ans=0
        ranges=[]
        for _ in range(n):
            x,y=map(int,input().split())
            if y>d: #No solution
                ans=-1
            else:
                left=x-math.sqrt(d**2-y**2)
                right=x+math.sqrt(d**2-y**2)
                ranges.append((left,right))
        if ans>=0:
            ranges.sort(key=lambda x:x[0])
            Right=ranges[0][0]-1
            for k in ranges:
                if k[0]>Right:
                    ans+=1
                    Right=k[1]
                else:
                    Right=min(Right,k[1])
        print(f"Case {count}: {ans}")
    input()

```

基本信息

#: 50454934
 题目: 01328
 提交人: 22n2200017737
 内存: 3736kB
 时间: 54ms
 语言: Python3
 提交时间: 2025-10-19 18:43:21

545C. Woodcutters

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

用时：55min

思路

这道题正确思路的逻辑并不复杂，但是可能的方向比较多，可能不容易直接想到正确思路。从左到右贪心：第0棵树一定可以往左倒，后面的树能往左则优先往左，如果不能往左，则考虑往右倒。可行的原因在于，往右倒唯一可能的隐患就是使得下一棵树没法往左倒，但如果下一棵树真的往左倒了，这一棵树本身也就没法倒了。这两种情况中，倒下的树总数是一致的。这道题我一开始想的是先统一往左倒，然后再循环一轮往右倒，但这个方法其实会导致有一些树的方向不是最优。后来，又想能不能按照高度进行贪心，即优先砍倒比较矮的树。但是比较矮的树未必代表周围空间就更宽裕，况且最左侧（和最右侧）一定是往左（右）倒，这个起始点其实是比较明确的，应该在第一个思路WA之后寻找**更贪心**的方法，即一次性处理完左边的树，然后再处理右边的树。

代码

```
N=int(input())
trees=[]
for _ in range(N):
    x,h=map(int,input().split())
    trees.append((x,h))
ans=[0]*N
ans[0]=-1
for i in range(1,N-1):
    x=trees[i][0]
    h=trees[i][1]
    if x-h>trees[i-1][0]+max(0,ans[i-1]*trees[i-1][1]):
        ans[i]=-1
    elif x+h<trees[i+1][0]:
        ans[i]=1
ans[N-1]=1
print(sum(map(abs,ans)))
```

代码运行截图

#	When	Who	Problem	Lang	Verdict	Time	Memory
344634964	Oct/19/2025 19:53UTC+8	xuhanecon	C - Woodcutters	Python 3	Accepted	359 ms	15400 KB

2. 学习总结和收获

在这次作业中，我对贪心的理解更为深入，也体会到正确思路与边界细节对算法正确性的决定作用。

- “排队做实验”与“军备竞赛”这两个题的思路在做完上一次作业和月考后已经比较熟悉了，核心在于**排序与双指**

针。它们帮助我复习了“按最关键变量排序后，线性解决问题的思维方式。

- “Radar Installation”这题我很快就想到了逆向思维的解法（因为正向完全没法做），而每个岛对应一个区间的形式又特别像月考里面**校门外的树**的区间合并思路，所以又采取了指针的方式解决了问题。
- “Woodcutters”这题的难点是虽然知道要用贪心，但具体按什么贪心，贪心的时候怎么操作，仍然需要比较好的逻辑性和经验。这道题既可以往左也可以往右的灵活性尤其增加了难度。这时候我们要做**真正的贪心**，也就是完全优先处理较左侧的树，处理完左边再处理右边，在这个基本算法的基础上，再去考虑向左向右的灵活性，即优先向左、不行向右的策略。如果不放心可以对这个逻辑进行考察（见解题思路），甚至先提交代码试验。
- 此外，“节省存储的矩阵乘法”这题我使用了字典与稀疏矩阵的结合方式。通过使用 `.get()` 函数和键值索引，我能在不浪费空间的情况下实现矩阵运算，也加深了对时间复杂度与空间效率的平衡理解。