Assignment #B: Dec Mock Exam大雪前一天

Updated 1649 GMT+8 Dec 5, 2024

2024 fall, Complied by 熊程宇 物理学院

说明：

1）月考： AC1

1. 题目

E22548: 机智的股民老张

http://cs101.openjudge.cn/practice/22548/

思路：

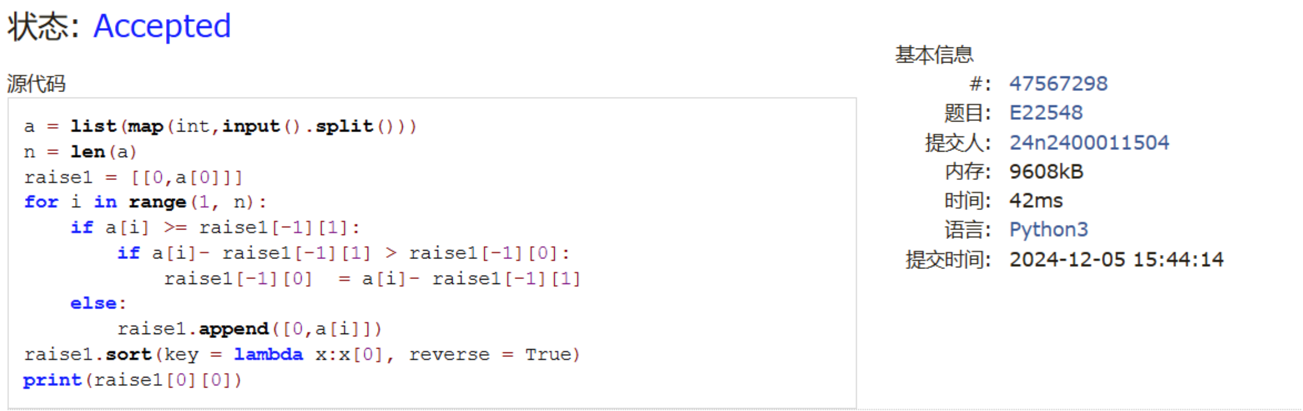
考试时思路如下：目标是找到最大的上升序列高度，于是用raise1记录这一段的高度和最低点. 若低于最低点，就是一个新的上升序列，若高于最高点，就更新当前的利润

问GPT明白了，没有必要存储无用的点，从前到后遍历，遇见最低价格就更新，然后更新当前卖出的利润即可

代码：

```
a = list(map(int,input().split()))
n = len(a)
raise1 = [[0,a[0]]]
for i in range(1, n):
    if a[i] >= raise1[-1][1]:
        if a[i]- raise1[-1][1] > raise1[-1][0]:
            raise1[-1][0]  = a[i]- raise1[-1][1]
    else:
        raise1.append([0,a[i]])
raise1.sort(key = lambda x:x[0], reverse = True)
print(raise1[0][0])
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```
a = list(map(int,input().split()))
n = len(a)
raise1 = [[0,a[0]]]
for i in range(1, n):
    if a[i] >= raise1[-1][1]:
        if a[i]- raise1[-1][1] > raise1[-1][0]:
            raise1[-1][0]  = a[i]- raise1[-1][1]
    else:
        raise1.append([0,a[i]])
raise1.sort(key = lambda x:x[0], reverse = True)
print(raise1[0][0])
```

基本信息

#:	47567298
题目:	E22548
提交人:	24n2400011504
内存:	9608kB
时间:	42ms
语言:	Python3
提交时间:	2024-12-05 15:44:14

M28701: 炸鸡排

greedy, http://cs101.openjudge.cn/practice/28701/

思路:

看完答案后意识到最关键是意识到第一句话:

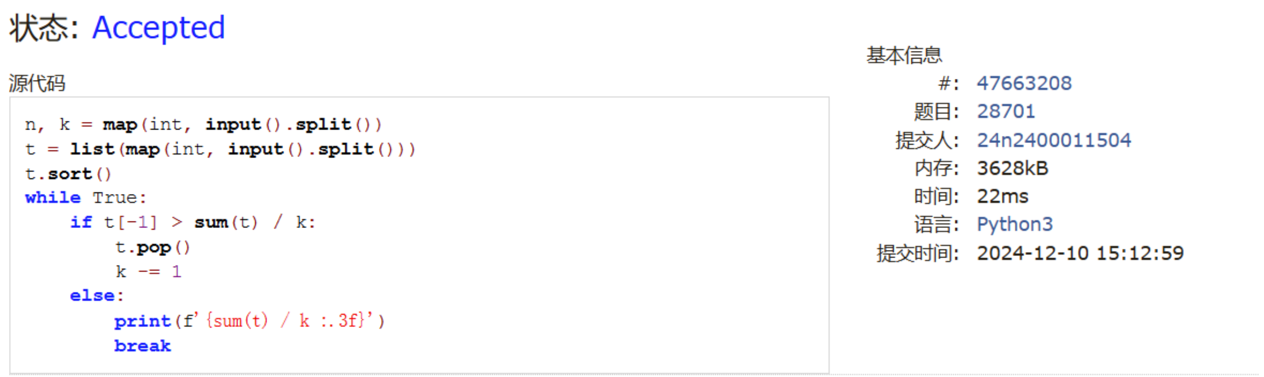
> 若每块鸡排都能炸完，用时就是所有鸡排平均值.

若炸不完，就放时间最长的那块一直炸，用剩下的平均即可

代码:

```python
n, k = map(int, input().split())
t = list(map(int, input().split()))
t.sort()
while True:
    if t[-1] > sum(t) / k:
        t.pop()
        k -= 1
    else:
        print(f'{sum(t) / k :.3f}')
        break
```

代码运行截图 ==（至少包含有"Accepted"）==



状态: Accepted

源代码

```
n, k = map(int, input().split())
t = list(map(int, input().split()))
t.sort()
while True:
    if t[-1] > sum(t) / k:
        t.pop()
        k -= 1
    else:
        print(f'{sum(t) / k :.3f}')
        break
```

基本信息

#:	47663208
题目:	28701
提交人:	24n2400011504
内存:	3628kB
时间:	22ms
语言:	Python3
提交时间:	2024-12-10 15:12:59

M20744: 土豪购物

dp, http://cs101.openjudge.cn/practice/20744/

思路：

一直当成贪心，现在才想起来连续最大子序列和是标准的动态规划问题.

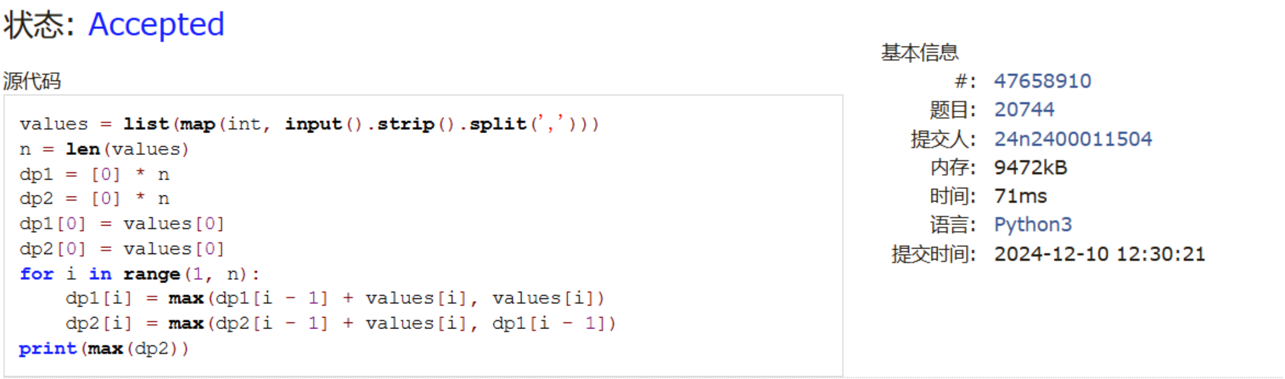
动态规划需要对某个量做递推，这里就是value的数量

难点在于双dp

代码：

```python
values = list(map(int, input().strip().split(',')))
n = len(values)
dp1 = [0] * n
dp2 = [0] * n
dp1[0] = values[0]
dp2[0] = values[0]
for i in range(1, n):
    dp1[i] = max(dp1[i - 1] + values[i], values[i])
    dp2[i] = max(dp2[i - 1] + values[i], dp1[i - 1])
print(max(dp2))
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```python
values = list(map(int, input().strip().split(',')))
n = len(values)
dp1 = [0] * n
dp2 = [0] * n
dp1[0] = values[0]
dp2[0] = values[0]
for i in range(1, n):
    dp1[i] = max(dp1[i - 1] + values[i], values[i])
    dp2[i] = max(dp2[i - 1] + values[i], dp1[i - 1])
print(max(dp2))
```

基本信息

#:	47658910
题目:	20744
提交人:	24n2400011504
内存:	9472kB
时间:	71ms
语言:	Python3
提交时间:	2024-12-10 12:30:21

T25561: 2022决战双十一

brute force, dfs, http://cs101.openjudge.cn/practice/25561/

思路：

代码：

代码运行截图 （至少包含有"Accepted"）

T20741: 两座孤岛最短距离

dfs, bfs, http://cs101.openjudge.cn/practice/20741/

思路：

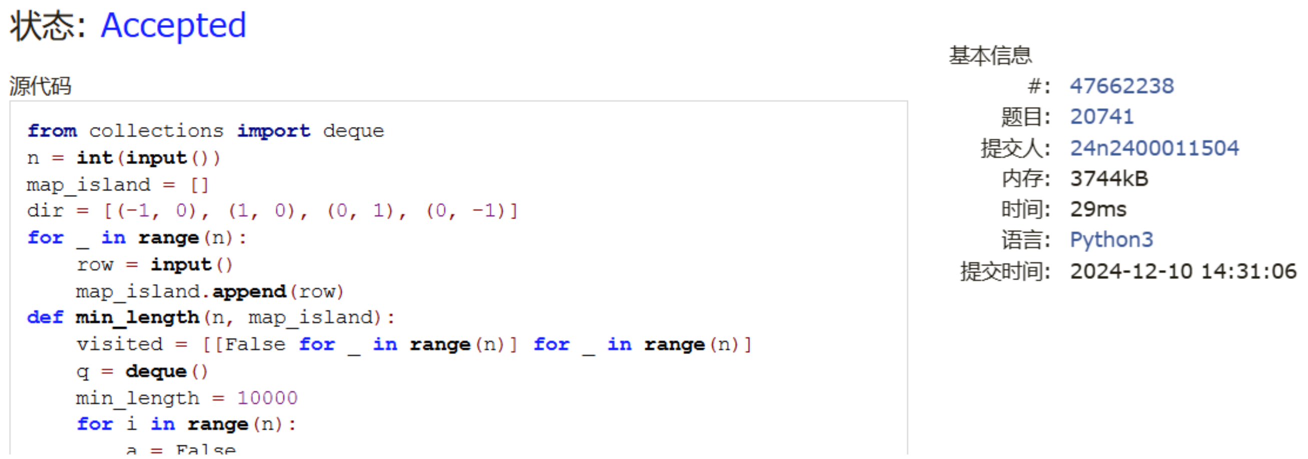
一开始只将孤岛的第一个点添加进了初始队列里，这就导致初始队列不完整，广度搜索就错了.

故初始队列就要搜索出第一个孤岛的所有点，再套用广度搜索模版即可.

代码：

```python
from collections import deque
n = int(input())
map_island = []
dir = [(-1, 0), (1, 0), (0, 1), (0, -1)]
for _ in range(n):
    row = input()
    map_island.append(row)
def min_length(n, map_island):
    visited = [[False for _ in range(n)] for _ in range(n)]
    q = deque()
    min_length = 10000
    for i in range(n):
        a = False
        for j in range(n):
            if map_island[i][j] == '1':
                stack = [(i, j)]
                q.append((i, j, 0))
                visited[i][j] = True
                a = True
                break
        if a:
            break
    while stack:
        x, y = stack.pop()
        if not visited[x][y] and map_island[x][y] == '1':
            visited[x][y] = True
            q.append((x, y, 0))
        for dx, dy in dir:
            cx, cy = x + dx, y + dy
            if 0 <= cx < n and 0 <= cy < n and map_island[cx][cy] == '1' and not
visited[cx][cy]:
                stack.append((cx, cy))
    while q:
        nx, ny, length = q.popleft()
        for dx, dy in dir:
            cx, cy = nx + dx, ny + dy
            if 0 <= cx < n and 0 <= cy < n and not visited[cx][cy]:
                if map_island[cx][cy] == '1':
                    min_length = length
                    return min_length
                else:
                    visited[cx][cy] = True
                    q.append((cx, cy, length + 1))
a = min_length(n, map_island)
print(a)
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```
from collections import deque
n = int(input())
map_island = []
dir = [(-1, 0), (1, 0), (0, 1), (0, -1)]
for _ in range(n):
    row = input()
    map_island.append(row)
def min_length(n, map_island):
    visited = [[False for _ in range(n)] for _ in range(n)]
    q = deque()
    min_length = 10000
    for i in range(n):
        a = False
```

基本信息

#: 47662238
题目: 20741
提交人: 24n2400011504
内存: 3744kB
时间: 29ms
语言: Python3
提交时间: 2024-12-10 14:31:06

T28776: 国王游戏

greedy, http://cs101.openjudge.cn/practice/28776

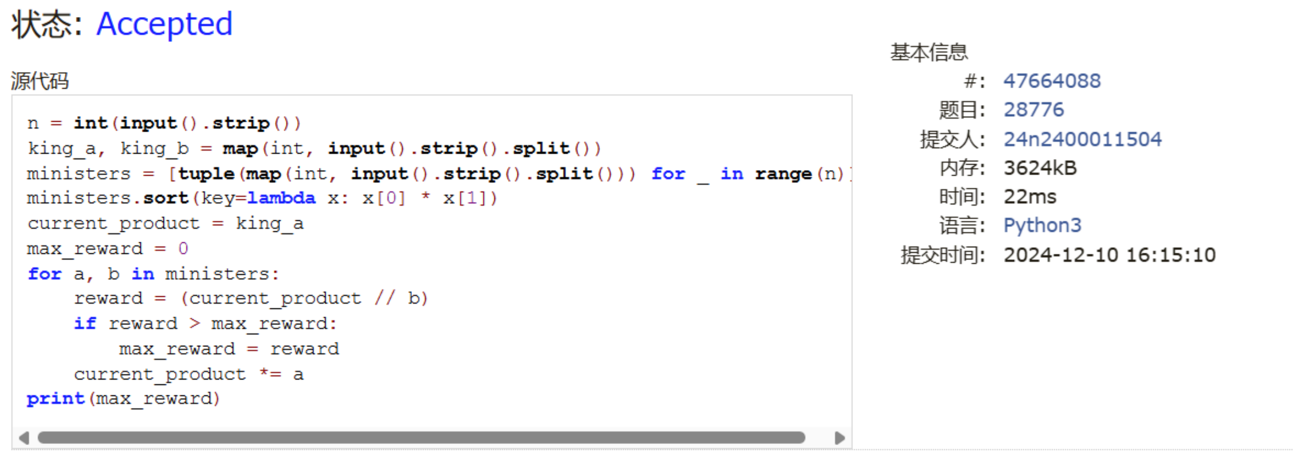
思路:

抄答案…

代码:

```
n = int(input().strip())
king_a, king_b = map(int, input().strip().split())
ministers = [tuple(map(int, input().strip().split())) for _ in range(n)]
ministers.sort(key=lambda x: x[0] * x[1])
current_product = king_a
max_reward = 0
for a, b in ministers:
    reward = (current_product // b)
    if reward > max_reward:
        max_reward = reward
    current_product *= a
print(max_reward)
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```
n = int(input().strip())
king_a, king_b = map(int, input().strip().split())
ministers = [tuple(map(int, input().strip().split())) for _ in range(n)]
ministers.sort(key=lambda x: x[0] * x[1])
current_product = king_a
max_reward = 0
for a, b in ministers:
    reward = (current_product // b)
    if reward > max_reward:
        max_reward = reward
    current_product *= a
print(max_reward)
```

基本信息

#: 47664088
题目: 28776
提交人: 24n2400011504
内存: 3624kB
时间: 22ms
语言: Python3
提交时间: 2024-12-10 16:15:10

2. 学习总结和收获

尝试了去年的考试题目，E和M还是比较容易的，感觉tough顶多试试搜索题目，贪心和动态规划的tough题目相对来说困难不少.