Assignment #B: Dec Mock Exam大雪前一天

Updated 1649 GMT+8 Dec 5, 2024

2024 fall, Complied by 同学的姓名、院系

说明：

1）月考： AC6（请改为同学的通过数）。考试题目都在"题库（包括计概、数算题目）"里面，按照数字题号能找到，可以重新提交。作业中提交自己最满意版本的代码和截图。

2）请把每个题目解题思路（可选），源码Python, 或者C++（已经在Codeforces/Openjudge上AC），截图（包含Accepted），填写到下面作业模版中（推荐使用 typora https://typoraio.cn，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

3）提交时候先提交pdf文件，再把md或者doc文件上传到右侧"作业评论"。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。

4）如果不能在截止前提交作业，请写明原因。

1. 题目

E22548: 机智的股民老张

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

思路：

不断更新最小数和zui'da

代码：

```
stock = list(map(int, input().split()))
n = len(stock)
dp = [0]*n
minstock = float('inf')
for i in range(1, n):
    minstock = min(minstock, stock[i])
    dp[i] = max(dp[i - 1], stock[i] - minstock)
print(dp[-1])
```

代码运行截图（至少包含有"Accepted"）

#47572011提交状态

状态: Accepted

源代码

```
stock = list(map(int, input().split()))
n = len(stock)
dp = [0]*n
minstock = float('inf')
for i in range(1, n):
    minstock = min(minstock, stock[i])
    dp[i] = max(dp[i - 1], stock[i] - minstock)
print(dp[-1])
```


M28701: 炸鸡排

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

思路:

（摘自群内发言）

首先，所有鸡肉的总时长是固定的sum(t)，每次炸k个，最多sum(t)/k秒就一定会结束，无论什么方案。此时就会出现两种情况，第一种情况是所有t[i]<sum(t)/k，这种情况可以直接给构造，具体构造类似于题目里给出的1 1 1的例子。第二种情况是存在一个大于sum(t)/k，那么意味着就算炸了理论最长时间都不能炸熟它，那么就直接把它丢进锅里不管了，考虑一个k-1的问题即可

代码:

```
n, k = map(int, input().split())
chicks = list(map(int, input().split()))
chicks.sort()
s = sum(chicks)
for i in range(n - 1, -1, -1):
    if chicks[i] > s / k:
        s -= chicks[i]
        k -= 1
print('%.3f' %(s / k))
```

代码运行截图 ==（至少包含有"Accepted"）==

状态: Accepted

源代码

```
n, k = map(int, input().split())
chicks = list(map(int, input().split()))
chicks.sort()
s = sum(chicks)
for i in range(n - 1, -1, -1):
    if chicks[i] > s / k:
        s -= chicks[i]
        k -= 1
print('%.3f' %(s / k))
```


M20744: 土豪购物

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

思路:

这道题总有种故人相识的感觉，但是具体状态转移方程还是想了很久

建造一个二维dp数组，决定参数分别是连续数组结束的位置和是否要抛弃某一个数，最后输出最大的'可抛弃'状态。

代码:

```
goods = list(map(int, input().split(',')))
n = len(goods)
dp = [[0,0] for _ in range(n)]
dp[0] = [goods[0]]*2
for i in range(1, n):
    dp[i][0] = max(dp[i - 1][0], 0) + goods[i]
    dp[i][1] = max(dp[i - 1][0], goods[i], dp[i - 1][1] + goods[i])
print(max(dp[i][1] for i in range(n)))
```

代码运行截图 （至少包含有"Accepted"）

状态：Accepted

源代码

```
goods = list(map(int, input().split(',')))
n = len(goods)
dp = [[0,0] for _ in range(n)]
dp[0] = [goods[0]]*2
for i in range(1, n):
    dp[i][0] = max(dp[i - 1][0], 0) + goods[i]
    dp[i][1] = max(dp[i - 1][0], goods[i], dp[i - 1][1] + goods[i])
print(max(dp[i][1] for i in range(n)))
```

T25561: 2022决战双十一

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

思路：

主打一个自己一个字写不出来遂认真学习标答

代码：

```
def plans(n, price, count, all_plans, plan):  # 递归列出所有购买方案
    if count == n + 1:
        all_plans.append(plan[:])
        return
    for i in price[count].keys():
        plan.append(i)
        plans(n, price, count + 1, all_plans, plan)
        plan.pop()
    return


def buy(n, m, price, coupon):
    all_plans = list()  # 列出所有购买方案
    plans(n, price, 1, all_plans, [])
    # for i in all_plans:
    #     print(i)
    final_price = list()  # 每个方案的最终价格
    for plan in all_plans:  # 对每个购买方案
        totals_rsp = list()  # 每个店铺的总价
        prices = [price[i][plan[i - 1]] for i in range(1, n + 1)]  # 每个商品的价格
        total = sum(prices)  # 所有商品的总价
        total -= total // 300 * 50  # 跨店满减
        for i in range(1, m + 1):  # 对每个店铺
            prices_rsp = [price[j + 1][plan[j]]
                          for j in range(n) if plan[j] == i]  # 每个商品在该店铺的价
格
            totals_rsp.append(sum(prices_rsp))  # 该店铺的总价
        store = 0
        for total_rsp in totals_rsp:
            store += 1
            discount = 0
```

```
            for j in coupon[store]:
                if total_rsp >= j[0]:
                    discount = max(j[1], discount)
            total -= discount
        final_price.append(total)
    # print(final_price)
    return min(final_price)


n, m = map(int, input().split())
price = dict()
coupon = dict()
for i in range(n):
    price_i = dict()
    price_raw = list(input().split())
    for j in price_raw:
        price_i[int(list(j.split(':'))[0])] = int(list(j.split(':'))[1])
    price[i + 1] = price_i
for i in range(m):
    coupon_i = list()
    coupon_raw = list(input().split())
    for j in range(len(coupon_raw)):
        coupon_i.append(tuple(map(int, coupon_raw[j].split('-'))))
    coupon[i + 1] = coupon_i
# print(n, m, price, coupon)
print(buy(n, m, price, coupon))
```

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：

一开始想怎么解决不会自产自销的问题，后来拍脑袋发现可以先用dfs把一个孤岛变成以2标记的数据组

然后就是bfs找到最短路径，但是因为还是不太熟悉bfs，dfs花了很多时间调试，打算多练习+把模板加到cheatsheet里

以及这题输入乍一看确实骗到我了（

代码：

```
from collections import deque

D = [(1, 0), (-1, 0), (0, 1), (0, -1)]
#dfs find a connected island
def dfs(x, y, n, maps, queue):
    maps[x][y] = '2'
    queue.append((x, y))
```

```
    for dx, dy in D:
        nx, ny = x + dx, y + dy
        if 0 <= nx <= n - 1 and 0 <= ny <= n - 1 and maps[nx][ny] == '1':
            dfs(nx, ny, n, maps, queue)

def bfs(queue, maps, n):
    steps = 0
    while queue:
        for _ in range(len(queue)):
            x, y = queue.popleft()
            for dx, dy in D:
                nx, ny = x + dx, y + dy
                if 0 <= nx <= n - 1 and 0 <= ny <= n - 1:
                    if maps[nx][ny] == '1':
                        return steps
                    elif maps[nx][ny] == '0':
                        maps[nx][ny] = '2'
                        queue.append((nx, ny))
        steps += 1
    return steps

def main():
    n = int(input())
    maps = []
    for _ in range(n):
        maps.append(list(input()))
    queue = deque()

    for i in range(n):
        for j in range(n):
            if maps[i][j] == '1':
                dfs(i, j, n, maps, queue)
                return bfs(queue, maps, n)

if __name__ == "__main__":
    print(main())
```

代码运行截图（至少包含有"Accepted"）

状态: Accepted

源代码

```
from collections import deque

D = [(1, 0), (-1, 0), (0, 1), (0, -1)]
#dfs find a connected island
def dfs(x, y, n, maps, queue):
    maps[x][y] = '2'
    queue.append((x, y))
    for dx, dy in D:
        nx, ny = x + dx, y + dy
        if 0 <= nx <= n - 1 and 0 <= ny <= n - 1 and maps[nx][ny] == '1':
            dfs(nx, ny, n, maps, queue)

def bfs(queue, maps, n):
    steps = 0
    while queue:
        for _ in range(len(queue)):
            x, y = queue.popleft()
            for dx, dy in D:
                nx, ny = x + dx, y + dy
                if 0 <= nx <= n - 1 and 0 <= ny <= n - 1:
                    if maps[nx][ny] == '1':
                        return steps
```

T28776: 国王游戏

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

思路:

对于某一大臣（第i位），设其前一个人（第i-1位）处理后的结果（未取整版）为ANS，则他能获得的钱为:

$$\text{money} = \text{ANS} \times \frac{\text{left}_{i-1} \cdot \text{right}_{i-1}}{\text{right}_i}$$

因而猜测是根据左手右手乘积排序

代码:

```
n = int(input())
kingl, kingr = map(int, input().split())
data = []
for i in range(n):
    left, right = map(int, input().split())
    data.append([left, right])
data.sort(key = lambda x:x[0]*x[1])
count = kingl
money = [count//data[0][1]]
for i in range(1, n):
    count *= data[i - 1][0]
    money.append(count//data[i][1])
print(max(money))
```

代码运行截图（至少包含有"Accepted"）

状态：Accepted

源代码

```
n = int(input())
kingl, kingr = map(int, input().split())
data = []
for i in range(n):
    left, right = map(int, input().split())
    data.append([left, right])
data.sort(key = lambda x:x[0]*x[1])
count = kingl
money = [count//data[0][1]]
for i in range(1, n):
    count *= data[i - 1][0]
    money.append(count//data[i][1])
print(max(money))
```

2. 学习总结和收获

如果作业题目简单，有否额外练习题目，比如：OJ“计概2024fall每日选做”、CF、LeetCode、洛谷等网站题目。

若有时间限制，预计AC2；无时间限制AC3-4；累了，短时间内不想碰键盘，但是人还得学习，悲伤（