# Assignment #A: 矩阵和动态规划

Updated 1406 GMT+8 Nov 14, 2023

2023 fall, Complied by Xinjie Song, Phy

**说明：**

1）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, 学号），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。

2）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、作业评论有md或者doc。

3）如果不能在截止前提交作业，请写明原因。

**编程环境**

操作系统：Windows 11 22H2

Python编程环境：PyCharm 2023.2 (Community Edition)

C/C++编程环境：g++ (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0

## 1. 必做题目

### OJ12558: 岛屿周长

matices, http://cs101.openjudge.cn/practice/12558/

思路：类似dp，根据左侧和上方是否是陆地决定某个位置计入多少条边。

**代码**

```
n, m = map(int, input().split())
matrix = [[0 for i in range(m + 2)]]
matrix += [[0] + list(map(int, input().split())) + [0] for i in range(n)]
matrix += [matrix[0][:]]
c = 0

for i in range(1, n + 1):
    for j in range(1, m + 1):
        if matrix[i][j]:
            c += 4 - matrix[i - 1][j] - matrix[i][j - 1] - matrix[i + 1][j] -
matrix[i][j + 1]
print(c)
```

代码运行截图


# OJ02760: 数字三角形

dp, http://cs101.openjudge.cn/practice/02760/

思路：经典dp思路，从下向上遍历

### 代码

```
n = int(input())
nums = [list(map(int, input().split())) for _ in range(n)]
for i in range(1, n):
    for j in range(n - i):
        nums[-i - 1][j] += max(nums[-i][j], nums[-i][j + 1])
print(nums[0][0])
```

代码运行截图

# OJ02773: 采药

dp, http://cs101.openjudge.cn/practice/02773

思路：背包问题变式

### 代码

```
t, m = map(int, input().split())
data = [list(map(int, input().split())) for _ in range(m)]
dp = [[0]*(t + 1) for _ in range(m)]
for i in range(m):
    for j in range(1, t + 1):
        if j >= data[i][0]:
            if not i:
                dp[0][j] = data[i][1]
            else:
                dp[i][j] = max(data[i][1] + dp[i - 1][j - data[i][0]], dp[i - 1]
[j])
        else:
            dp[i][j] = dp[i - 1][j]
print(dp[-1][-1])
```

代码运行截图


## OJ18106: 螺旋矩阵

matrices, http://cs101.openjudge.cn/practice/18106/

这个题目技巧性较强，可以看题解记住。

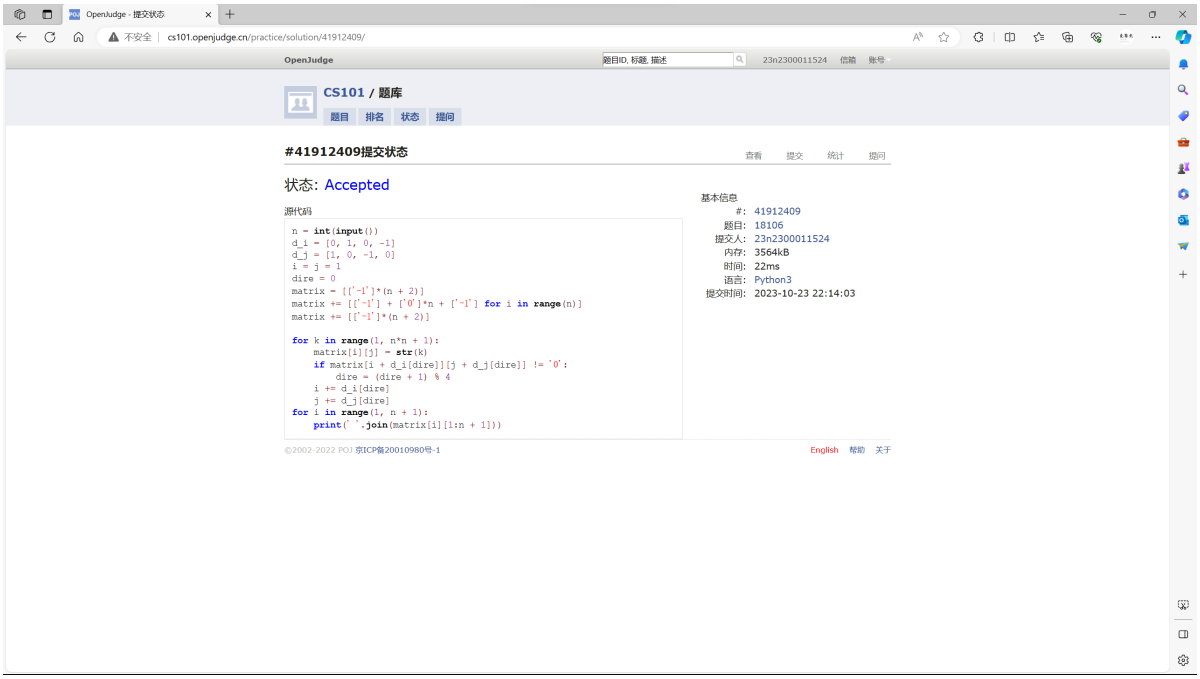
思路：模拟法，利用一个周期为4的变量记录方向，遇到边就更改方向。

**代码**

```
n = int(input())
d_i = [0, 1, 0, -1]
d_j = [1, 0, -1, 0]
i = j = 1
dire = 0
matrix = [['-1']*(n + 2)]
matrix += [['-1'] + ['0']*n + ['-1'] for i in range(n)]
matrix += [['-1']*(n + 2)]

for k in range(1, n*n + 1):
    matrix[i][j] = str(k)
    if matrix[i + d_i[dire]][j + d_j[dire]] != '0':
        dire = (dire + 1) % 4
    i += d_i[dire]
    j += d_j[dire]
for i in range(1, n + 1):
    print(' '.join(matrix[i][1:n + 1]))
```

代码运行截图




# 2. 选做题目

如果耗时太长，直接看解题思路，或者源码

## CF189A: Cut Ribbon

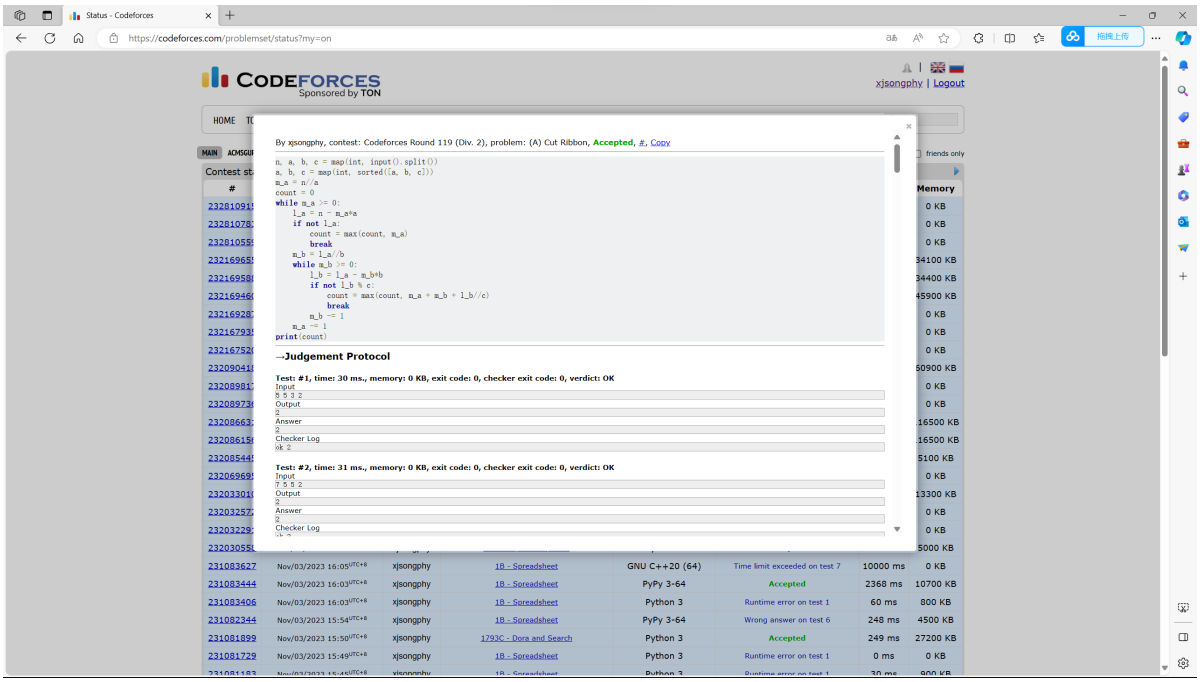
brute force/dp, 1300, https://codeforces.com/problemset/problem/189/A

思路：从最小的片段开始暴力求解

### 代码

```
n, a, b, c = map(int, input().split())
a, b, c = map(int, sorted([a, b, c]))
m_a = n//a
count = 0
while m_a >= 0:
    l_a = n - m_a*a
    if not l_a:
        count = max(count, m_a)
        break
    m_b = l_a//b
    while m_b >= 0:
        l_b = l_a - m_b*b
        if not l_b % c:
            count = max(count, m_a + m_b + l_b//c)
            break
        m_b -= 1
    m_a -= 1
print(count)
```

代码运行截图




## CF455A: Boredom

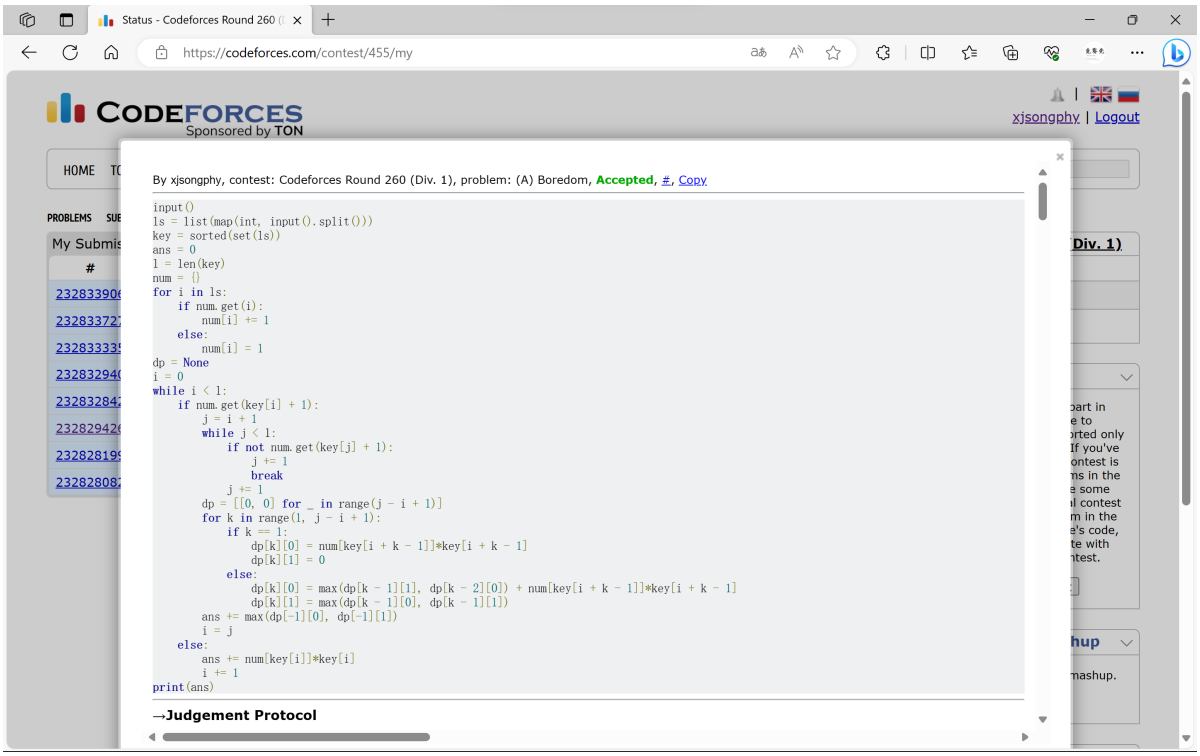
dp, 1500, https://codeforces.com/contest/455/problem/A

思路：若某个数右侧没有相邻数，计入得分；否则，确认连续数列收尾，利用动态规划确认某个数及以前在这个数取与不取的情况下的最大得分，得到这个连续数列数据可以达到的最高分，将索引移动到末尾，继续循环即可。

**代码**

```
input()
ls = list(map(int, input().split()))
key = sorted(set(ls))
ans = 0
l = len(key)
num = {}
for i in ls:
    if num.get(i):
        num[i] += 1
    else:
        num[i] = 1
dp = None
i = 0
while i < l:
    if num.get(key[i] + 1):
        j = i + 1
        while j < l:
            if not num.get(key[j] + 1):
                j += 1
                break
            j += 1
        dp = [[0, 0] for _ in range(j - i + 1)]
        for k in range(1, j - i + 1):
            if k == 1:
                dp[k][0] = num[key[i + k - 1]]*key[i + k - 1]
                dp[k][1] = 0
            else:
                dp[k][0] = max(dp[k - 1][1], dp[k - 2][0]) + num[key[i + k -
1]]*key[i + k - 1]
                dp[k][1] = max(dp[k - 1][0], dp[k - 1][1])
        ans += max(dp[-1][0], dp[-1][1])
        i = j
    else:
        ans += num[key[i]]*key[i]
        i += 1
print(ans)
```

代码运行截图




# 3. 学习总结和收获

如果时间充足的话，基础的动态规划题目已经可以解答了，感觉基础的动态规划题目没有以前想象的那么难，接下来就要继续练习挑战更难的题目以及贪心题目了。

截至2023年11月15日，OJ完成题目106道，CF完成题目46道