1. 题目

02692: 假币问题

brute force, http://cs101.openjudge.cn/practice/02692

代码：

n = int(input())

def check(coins, case):

for item in case:

left, right, res = item.split()

left\_total = sum(coins[i] for i in left)

right\_total = sum(coins[i] for i in right)

if left\_total == right\_total and res != 'even':

return False

elif left\_total < right\_total and res != 'down':

return False

elif left\_total > right\_total and res != 'up':

return False

return True

for \_ in range(n):

case = [input().strip() for \_ in range(3)]

for counterfeit in 'ABCDEFGHIJKL':

found = False

for weight in [-1, 1]:

coins = {coin: 0 for coin in 'ABCDEFGHIJKL'}

coins[counterfeit] = weight

if check(coins, case):

found = True

tag = "light" if weight == -1 else "heavy"

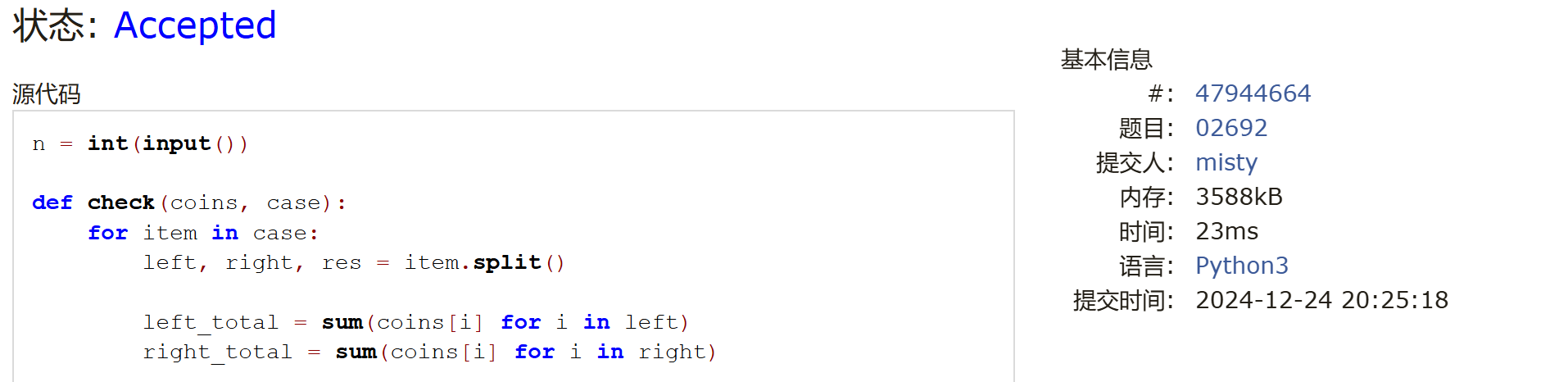
print(f'{counterfeit} is the counterfeit coin and it is {tag}.')

break

if found:

break

代码运行截图 （至少包含有"Accepted"）



01088: 滑雪

dp, dfs similar, http://cs101.openjudge.cn/practice/01088

代码：

import sys

sys.setrecursionlimit(1 << 30)

from functools import lru\_cache

@lru\_cache(maxsize=None)

def dfs(x, y):

if d[x][y] > 0: return d[x][y]

ans = 1

for nx, ny in directions:

tx, ty = x + nx, y + ny

if 0 <= tx < n and 0 <= ty < m and a[tx][ty] < a[x][y]:

ans = max(ans, dfs(tx, ty) + 1)

d[x][y] = ans

return ans

n, m = map(int, input().split())

a = [list(map(int, input().split())) for \_ in range(n)]

d = [[0] \* m for \_ in range(n)]

directions = [(-1, 0), (0, 1), (1, 0), (0, -1)]

ans = 1

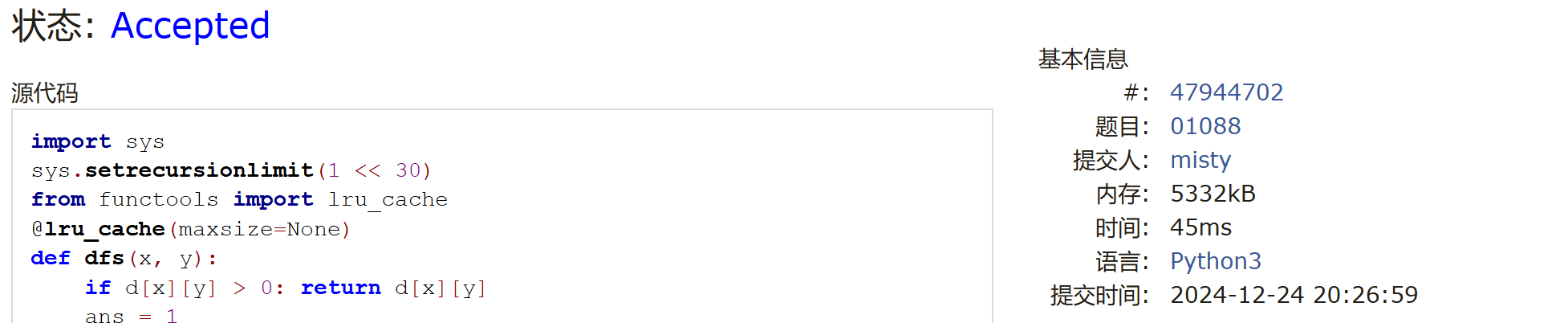
for i in range(n):

for j in range(m):

ans = max(ans, dfs(i, j))

print(ans)

代码运行截图 ==（至少包含有"Accepted"）==



25572: 螃蟹采蘑菇

bfs, dfs, http://cs101.openjudge.cn/practice/25572/

代码：

from collections import deque

# 定义四个方向：右、下、左、上

dire = [(0, 1), (1, 0), (0, -1), (-1, 0)]

def bfs(a, x1, y1, x2, y2):

visit = set() # 使用集合来避免重复访问

queue = deque([(x1, y1, x2, y2)])

visit.add((x1, y1, x2, y2)) # 初始点加入访问集合

while queue:

xa, ya, xb, yb = queue.popleft()

# 遍历四个方向

for xi, yi in dire:

# 计算新位置

nx1, ny1 = xa + xi, ya + yi

nx2, ny2 = xb + xi, yb + yi

# 判断新位置是否合法

if 0 <= nx1 < a and 0 <= ny1 < a and 0 <= nx2 < a and 0 <= ny2 < a:

if (nx1, ny1, nx2, ny2) not in visit and Matrix[nx1][ny1] != 1 and Matrix[nx2][ny2] != 1:

# 加入队列并标记访问

queue.append((nx1, ny1, nx2, ny2))

visit.add((nx1, ny1, nx2, ny2))

# 检查是否到达目标

if Matrix[nx1][ny1] == 9 or Matrix[nx2][ny2] == 9:

return True

return False

# 读取输入

a = int(input())

Matrix = [list(map(int, input().split())) for \_ in range(a)]

# 找到第一个和第二个 '5' 的位置

x1, y1, x2, y2 = -1, -1, -1, -1

found\_first = False

for i in range(a):

for j in range(a):

if Matrix[i][j] == 5:

if not found\_first:

x1, y1 = i, j

Matrix[i][j] = 0 # 标记为已访问

found\_first = True

else:

x2, y2 = i, j

Matrix[i][j] = 0 # 标记为已访问

break

if x2 != -1: # 如果第二个 5 已经找到

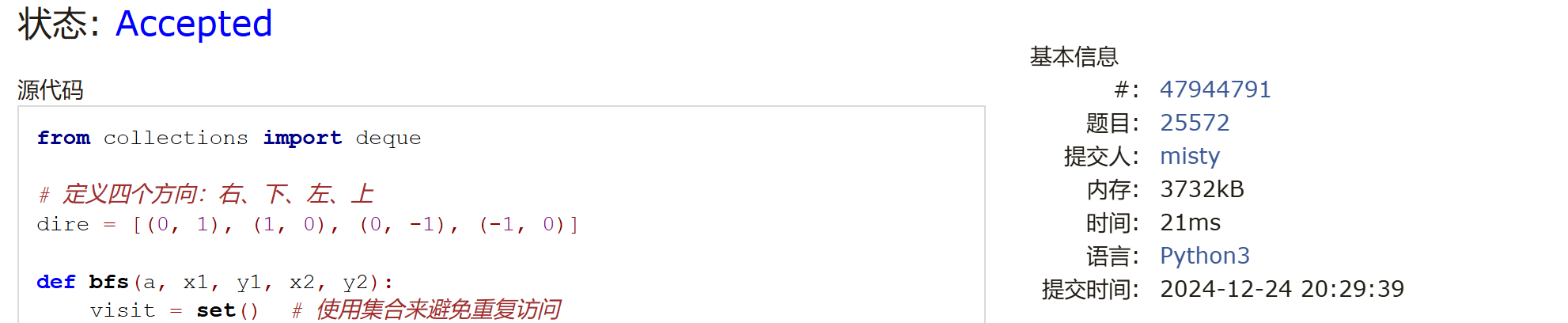
break

# 运行 BFS 检查是否可以从 (x1, y1) 到 (x2, y2)

check = bfs(a, x1, y1, x2, y2)

print('yes' if check else 'no')

代码运行截图 （至少包含有"Accepted"）



27373: 最大整数

dp, http://cs101.openjudge.cn/practice/27373/

代码：

def f(string):

if string=='':

return 0

else:

return int(string)

m=int(input())#最大位数

n=int(input())#正整数数量

l=input().split()

#冒泡排序

for i in range(n):

for j in range(n-1-i):

if l[j] + l[j+1] > l[j+1] + l[j]:

l[j],l[j+1] = l[j+1],l[j]

weight=[]#每个元素的位数

for num in l:

weight.append(len(num))

#dp[i][j]在前i数中选择，不超过j位，最大可能数值

dp=[['']\*(m+1) for \_ in range(n+1)]

for k in range(m+1):

dp[0][k]=''#无法组成整数

for q in range(n+1):

dp[q][0]=''#无法组成整数

for i in range(1,n+1):

for j in range(1,m+1):

if weight[i-1]>j:#不能选第i个，因为会超位数

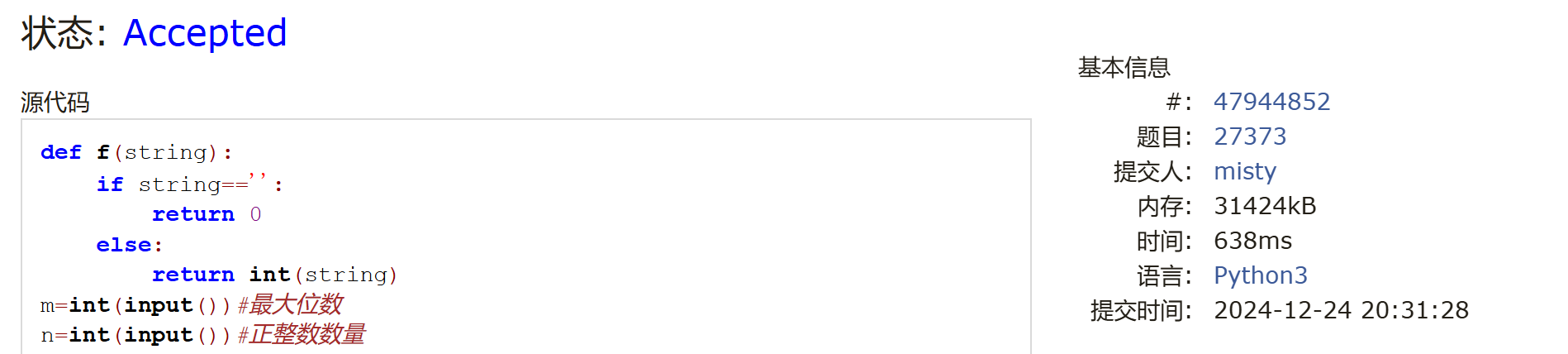
dp[i][j]=dp[i-1][j]

else:#可以选第i个也可以不选

dp[i][j]=str(max(f(dp[i-1][j]),int(l[i-1]+dp[i-1][j-weight[i-1]])))

print(dp[n][m])

代码运行截图 （至少包含有"Accepted"）



02811: 熄灯问题

brute force, http://cs101.openjudge.cn/practice/02811

代码：

X = [[0,0,0,0,0,0,0,0]]

Y = [[0,0,0,0,0,0,0,0]]

for \_ in range(5):

X.append([0] + [int(x) for x in input().split()] + [0])

Y.append([0 for x in range(8)])

X.append([0,0,0,0,0,0,0,0])

Y.append([0,0,0,0,0,0,0,0])

import copy

for a in range(2):

Y[1][1] = a

for b in range(2):

Y[1][2] = b

for c in range(2):

Y[1][3] = c

for d in range(2):

Y[1][4] = d

for e in range(2):

Y[1][5] = e

for f in range(2):

Y[1][6] = f

A = copy.deepcopy(X)

B = copy.deepcopy(Y)

for i in range(1, 7):

if B[1][i] == 1:

A[1][i] = abs(A[1][i] - 1)

A[1][i-1] = abs(A[1][i-1] - 1)

A[1][i+1] = abs(A[1][i+1] - 1)

A[2][i] = abs(A[2][i] - 1)

for i in range(2, 6):

for j in range(1, 7):

if A[i-1][j] == 1:

B[i][j] = 1

A[i][j] = abs(A[i][j] - 1)

A[i-1][j] = abs(A[i-1][j] - 1)

A[i+1][j] = abs(A[i+1][j] - 1)

A[i][j-1] = abs(A[i][j-1] - 1)

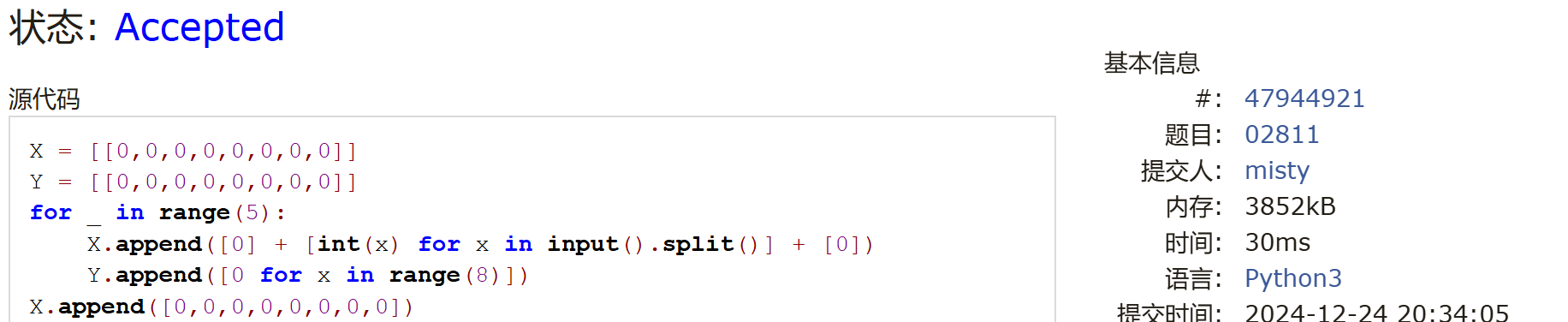
A[i][j+1] = abs(A[i][j+1] - 1)

if A[5][1]==0 and A[5][2]==0 and A[5][3]==0 and A[5][4]==0 and A[5][5]==0 and A[5][6]==0:

for i in range(1, 6):

print(" ".join(repr(y) for y in [B[i][1],B[i][2],B[i][3],B[i][4],B[i][5],B[i][6] ]))

代码运行截图 （至少包含有"Accepted"）



08210: 河中跳房子

binary search, greedy, http://cs101.openjudge.cn/practice/08210/

代码：

L,n,m = map(int,input().split())

rock = [0]

for i in range(n):

rock.append(int(input()))

rock.append(L)

def check(x):

num = 0

now = 0

for i in range(1, n+2):

if rock[i] - now < x:

num += 1

else:

now = rock[i]

if num > m:

return True

else:

return False

# https://github.com/python/cpython/blob/main/Lib/bisect.py

'''

2022fall-cs101，刘子鹏，元培。

源码的二分查找逻辑是给定一个可行的下界和不可行的上界，通过二分查找，将范围缩小同时保持下界可行而区间内上界不符合，

但这种最后print(lo-1)的写法的基础是最后夹出来一个不可行的上界，但其实L在这种情况下有可能是可行的

（考虑所有可以移除所有岩石的情况），所以我觉得应该将上界修改为不可能的 L+1 的逻辑才是正确。

例如：

25 5 5

1

2

3

4

5

应该输出 25

'''

# lo, hi = 0, L

lo, hi = 0, L+1

ans = -1

while lo < hi:

mid = (lo + hi) // 2

if check(mid):

hi = mid

else: # 返回False，有可能是num==m

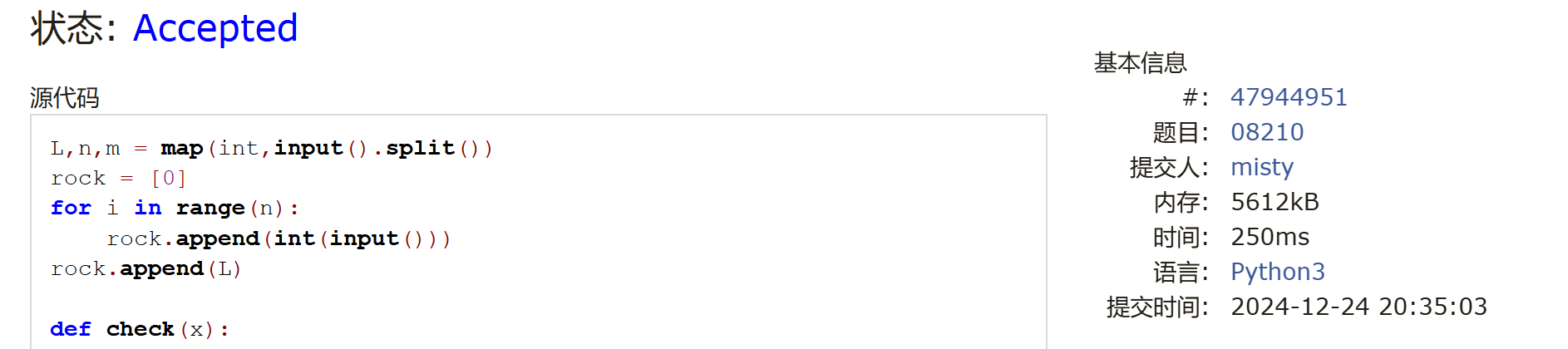
ans = mid # 如果num==m, mid可能是答案

lo = mid + 1

#print(lo-1)

print(ans)

代码运行截图 （至少包含有"Accepted"）



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。

1. 争取能有ac的题吧
2. 还得背背模板