1. 题目

1115. 取石子游戏

dfs, <https://www.acwing.com/problem/content/description/1117/>

代码：

#include<iostream>

#include<algorithm>//算法库,swap

using namespace std;

bool pd(long a,long b)//判断是否满足提示中条件

{

if(a/b>=2 || a==b) return true;//相等先拿的一定赢

else return !pd(b,a-b);//如果小于两倍，则先手取完之后，第二堆数目必定比第一堆多

}

int main()

{

long long a,b;

while(cin>>a>>b)

{

if(a==0||b==0)//结束进程

return 0;

if(b>a)

swap(a,b);

if(pd(a,b))

cout<<"win"<<endl;

else

cout<<"lose"<<endl;

}

return 0;

}

代码运行截图 （至少包含有"Accepted"）



25570: 洋葱

Matrices, <http://cs101.openjudge.cn/practice/25570>

代码：

DIRECTIONS = ((0, 1), (1, 0), (0, -1), (-1, 0))n = int(input())N = 0onion = [[-1e9 for i in range(n + 2)]] + [[-1e9] + list(map(int, input().split())) + [-1e9] for i in range(n)] + [[-1e9 for i in range(n + 2)]]dx, dy = DIRECTIONS[0]x, y = 1, 0layer = [0 for i in range(n // 2 + 1)]for i in range(1, 1 + n \* n):

if onion[x + dx][y + dy] == -1e9:

N += 1

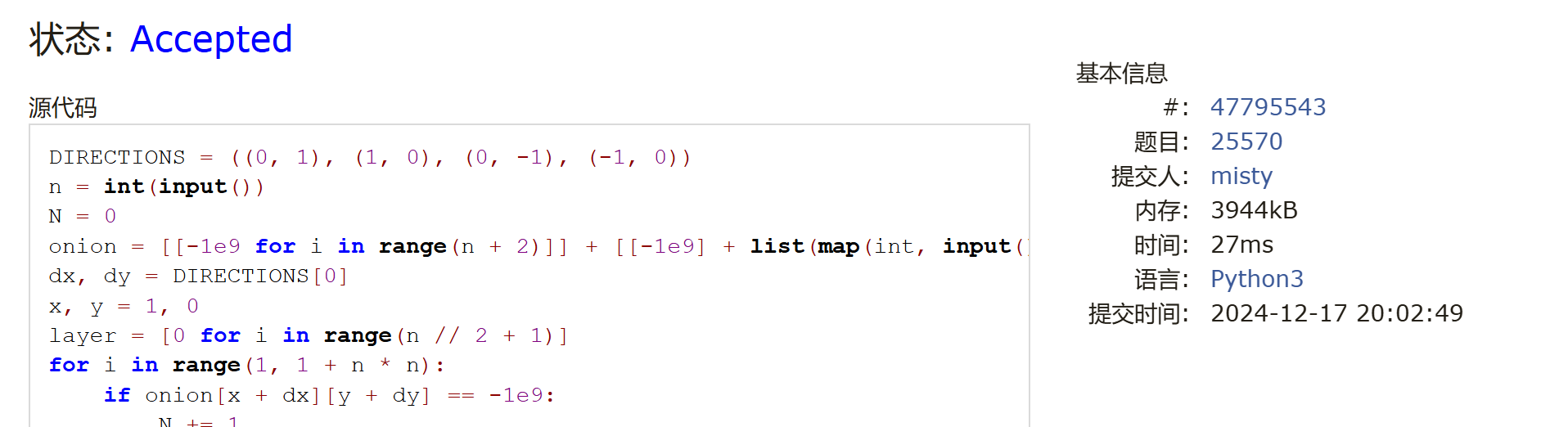
dx, dy = DIRECTIONS[N % 4]

x, y = x + dx, y + dy

layer[N // 4] += onion[x][y]

onion[x][y] = -1e9print(max(layer))

代码运行截图 ==（至少包含有"Accepted"）==



1526C1. Potions(Easy Version)

greedy, dp, data structures, brute force, \*1500, <https://codeforces.com/problemset/problem/1526/C1>

代码：

a=int(input())potions=list(map(int,input().split()))dp=[-float('inf')]\*(1+a)dp[0]=0for i in range(1+a):

for j in range(i,0,-1):

temp=max(dp[j],dp[j-1]+potions[i-1])

if temp>=0:

dp[j]=tempfor i in range(a,-1,-1):

if dp[i]>=0:

print(i)

break

代码运行截图 （至少包含有"Accepted"）



22067: 快速堆猪

辅助栈，<http://cs101.openjudge.cn/practice/22067/>

代码：

class MinStack:

def \_\_init\_\_(self):

self.stack = []

self.min\_stack = []

def push(self, x):

self.stack.append(x)

if not self.min\_stack or x <= self.min\_stack[-1]:

self.min\_stack.append(x)

def pop(self):

if self.stack:   
 top = self.stack.pop()

if top == self.min\_stack[-1]:

self.min\_stack.pop()

def min(self):

if self.min\_stack:

return self.min\_stack[-1]

else:

return None

while True:

try:

command = input().strip()

if command.startswith('push'):

value = int(command.split()[1])

min\_stack.push(value)

elif command.startswith('pop'):

min\_stack.pop()

elif command.startswith('min'):

min\_value = min\_stack.min()

if min\_value is not None:

print(min\_value)

except EOFError:

break

代码运行截图 （至少包含有"Accepted"）



20106: 走山路

Dijkstra, <http://cs101.openjudge.cn/practice/20106/>

代码：import heapq #260ms

def find\_min\_cost\_path(n, m, mat, queries):

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

results = []

for x, y, xx, yy in queries:

if mat[x][y] == '#' or mat[xx][yy] == '#':

results.append("NO")

continue

dist = {(x, y): 0} # Distance dictionary to keep track of minimum cost to each node

heap = [(0, x, y)] # Priority queue: (cost, row, col)

found = False

while heap:

cost, i, j = heapq.heappop(heap)

# If the target is reached, record the result and exit the loop

if (i, j) == (xx, yy):

results.append(cost)

found = True

break

# Explore all possible moves

for di, dj in directions:

ni, nj = i + di, j + dj

if 0 <= ni < n and 0 <= nj < m and mat[ni][nj] != '#':

new\_cost = cost + abs(int(mat[ni][nj]) - int(mat[i][j]))

# Update the cost if it's lower than any previously recorded cost

if (ni, nj) not in dist or new\_cost < dist[(ni, nj)]:

dist[(ni, nj)] = new\_cost

heapq.heappush(heap, (new\_cost, ni, nj))

if not found:

results.append("NO")

return results

# Input processing

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

mat = [input().split() for \_ in range(n)]

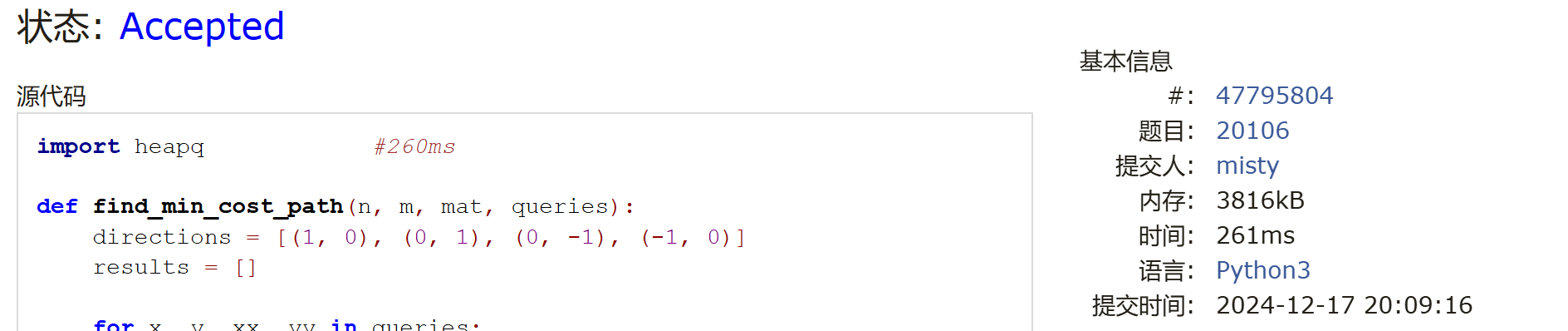
queries = [tuple(map(int, input().split())) for \_ in range(p)]

# Solve the problem and output results

answers = find\_min\_cost\_path(n, m, mat, queries)

print("\n".join(map(str, answers)))

代码运行截图 （至少包含有"Accepted"）



04129: 变换的迷宫

bfs, <http://cs101.openjudge.cn/practice/04129/>

代码：

from collections import deque

def bfs(matrix, n, m, k, start, end):

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

queue = deque([(start[0], start[1], 0)])

visited = [[[-1] \* k for \_ in range(m)] for \_ in range(n)]

visited[start[0]][start[1]][0] = 0

while queue:

x, y, time = queue.popleft()

if (x, y) == end:

return time

for dx, dy in directions:

cx, cy = x + dx, y + dy

if 0 <= cx < n and 0 <= cy < m and visited[cx][cy][(time + 1) % k] == -1:

if (time + 1) % k == 0: # 注意⚠️这里是time+1

visited[cx][cy][(time + 1) % k] = 0

queue.append((cx, cy, time + 1))

else:

# 注意⚠️此处如果漏了可以=‘S’的情况会报错

if matrix[cx][cy] != '#':

visited[cx][cy][(time + 1) % k] = 0

queue.append((cx, cy, time + 1))

return 'Oop!'

for \_ in range(int(input())):

R, C, K = map(int, input().split())

board = [input() for \_ in range(R)]

start = None

end = None

for i in range(R):

for j in range(C):

if board[i][j] == 'S':

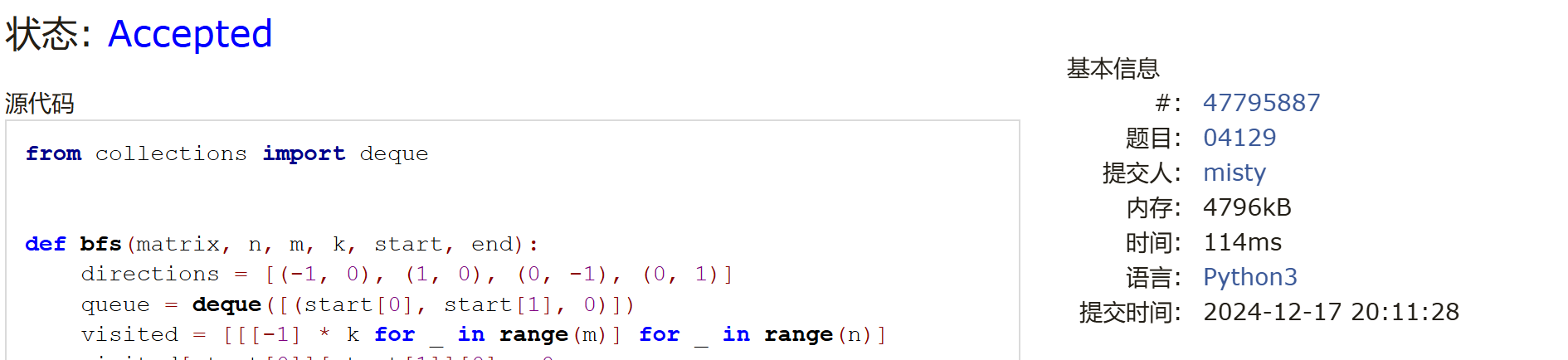
start = (i, j)

if board[i][j] == 'E':

end = (i, j)

print(bfs(board, R, C, K, start, end))

代码运行截图 （至少包含有"Accepted"）



2. 学习总结和收获

(1)题好难啊，不知道机考能做对几个

(2)这几天把之前的作业再看一遍