# Assignment #C: 五味杂陈

Updated 1148 GMT+8 Dec 10, 2024

2024 fall, Complied by <mark>付耀贤，信息管理系</mark>

## 1. 题目

### 1115. 取石子游戏

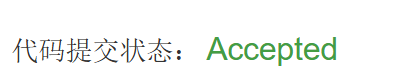
dfs, https://www.acwing.com/problem/content/description/1117/

思路：题目后面给了提示，这道题就显得不难了，用循环就能做。如果不给提示，恐怕我是怎么也想不出来。

代码：

while True:  
 a,b=sorted(map(int,input().split()))  
 if a+b==0:  
 break  
 c=0  
 while b//a<2:  
 if a==b:  
 break  
 b-=a  
 c+=1  
 a,b=min(a,b),max(a,b)  
 if c%2==0:  
 print("win")  
 else:  
 print("lose")

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 25570: 洋葱

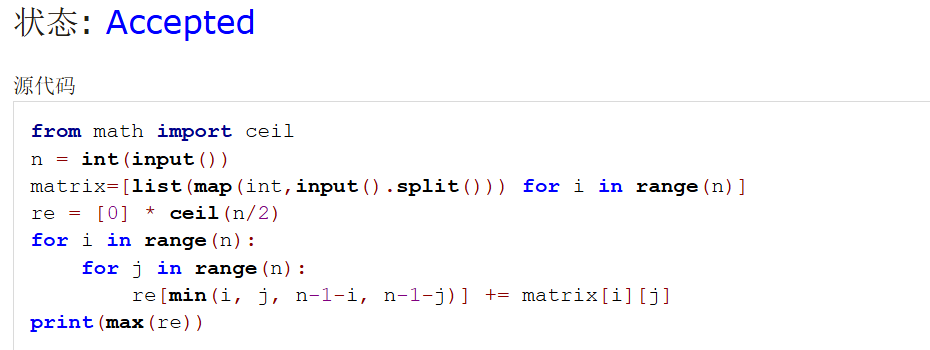
Matrices, http://cs101.openjudge.cn/practice/25570

思路：太绕了呜呜。自己写的时候心态已经崩溃了，看了题解更崩溃了，，这么简洁的代码怎么想到的。后来反思了一下，我是从圈出发，不断把每个数加进去；题解从数出发，考虑把数归类到圈里。这种思考视角的转变很有趣，很有用！

代码：

from math import ceil  
n = int(input())  
matrix=[list(map(int,input().split())) for i in range(n)]  
re = [0] \* ceil(n/2)  
for i in range(n):  
 for j in range(n):  
 re[min(i, j, n-1-i, n-1-j)] += matrix[i][j]  
print(max(re))

代码运行截图 ==（至少包含有"Accepted"）==



### 1526C1. Potions(Easy Version)

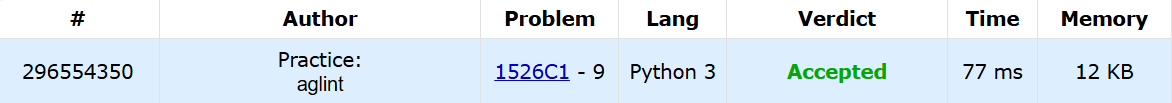
greedy, dp, data structures, brute force, \*1500, https://codeforces.com/problemset/problem/1526/C1

思路：没有用堆的解法也过了，就是将新数字与已经使用过的比较，不断更新所要使用的数字。

代码：

n = int(input())  
potions = list(map(int, input().split()))  
hp=0  
c=0  
used=[]  
for potion in potions:  
 if hp+potion>=0:  
 hp+=potion  
 used.append(potion)  
 c+=1  
 elif used and potion>min(used):  
 out=min(used)  
 used.remove(out)  
 hp-=out  
 hp+=potion  
 used.append(potion)  
print(c)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 22067: 快速堆猪

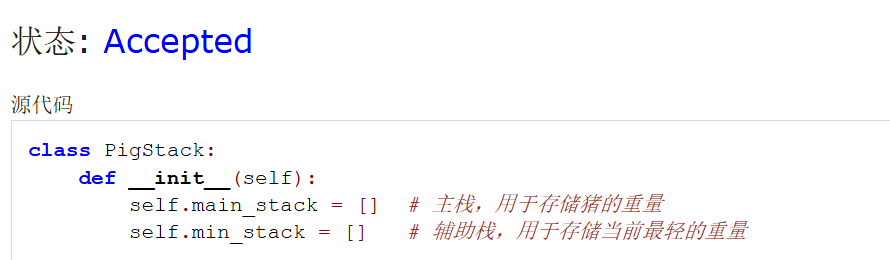
辅助栈，http://cs101.openjudge.cn/practice/22067/

思路：一步一步让AI拆解整个过程，通过这个题感受到了栈的具体应用

代码：

class PigStack:  
 def \_\_init\_\_(self):  
 self.main\_stack = [] # 主栈，用于存储猪的重量  
 self.min\_stack = [] # 辅助栈，用于存储当前最轻的重量  
  
 def push(self, n):  
 self.main\_stack.append(n)  
 # 如果辅助栈为空，或者新重量小于等于辅助栈顶部的重量，推入辅助栈  
 if not self.min\_stack or n <= self.min\_stack[-1]:  
 self.min\_stack.append(n)  
  
 def pop(self):  
 if self.main\_stack:  
 top = self.main\_stack.pop()  
 # 如果弹出的重量等于辅助栈栈顶，弹出辅助栈顶部  
 if self.min\_stack and top == self.min\_stack[-1]:  
 self.min\_stack.pop()  
  
 def get\_min(self):  
 if self.min\_stack:  
 return self.min\_stack[-1]  
 return None   
  
import sys  
pig\_stack = PigStack()  
for line in sys.stdin:  
 command = line.strip()  
 if command.startswith("push"):  
 \_, n = command.split()  
 pig\_stack.push(int(n))  
 elif command == "pop":  
 pig\_stack.pop()  
 elif command == "min":  
 min\_weight = pig\_stack.get\_min()  
 if min\_weight is not None:  
 print(min\_weight)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 20106: 走山路

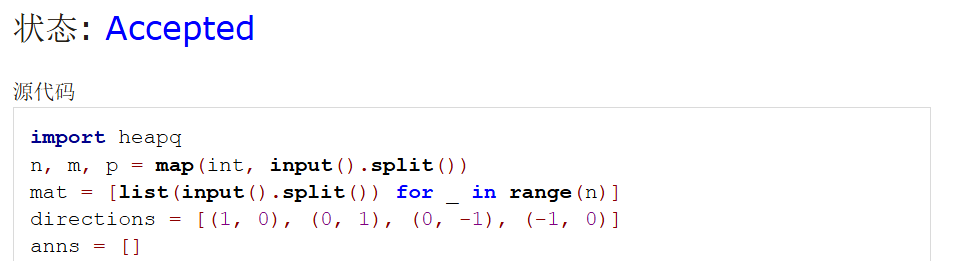
Dijkstra, http://cs101.openjudge.cn/practice/20106/

思路：真难啊。用堆来实现优先队列，快速找到当前消耗最小的节点。

代码：

import heapq   
n, m, p = map(int, input().split())  
mat = [list(input().split()) for \_ in range(n)]  
directions = [(1, 0), (0, 1), (0, -1), (-1, 0)]  
anns = []  
for \_ in range(p):  
 ans = 'NO'  
 x, y, xx, yy = map(int, input().split())  
 if mat[x][y] != '#' and mat[xx][yy] != '#':  
 dist = {(x, y): 0}  
 q = [(0, x, y)]   
 while q:  
 s, i, j = heapq.heappop(q)  
 if i == xx and j == yy:  
 ans = s  
 break  
 for a, b in directions:  
 ii, jj = i + a, j + b  
 if 0 <= ii < n and 0 <= jj < m and mat[ii][jj] != '#':  
 cost = s + abs(int(mat[ii][jj]) - int(mat[i][j]))  
 if (ii, jj) not in dist or cost < dist[(ii, jj)]:  
 dist[(ii, jj)] = cost  
 heapq.heappush(q, (cost, ii, jj))  
 anns.append(ans)  
for \_ in anns:  
 print(\_)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 04129: 变换的迷宫

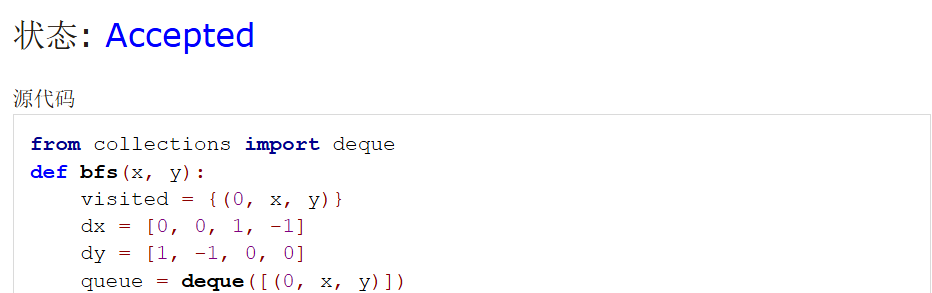
bfs, http://cs101.openjudge.cn/practice/04129/

思路：三维结构，取模，处理得很妙。

代码：

from collections import deque  
def bfs(x, y):  
 visited = {(0, x, y)}  
 dx = [0, 0, 1, -1]  
 dy = [1, -1, 0, 0]  
 queue = deque([(0, x, y)])  
 while queue:  
 time, x, y = queue.popleft()  
 for i in range(4):  
 nx, ny = x + dx[i], y + dy[i]  
 temp = (time + 1) % k  
 if 0 <= nx < r and 0 <= ny < c and (temp, nx, ny) not in visited:  
 cur = maze[nx][ny]  
 if cur == 'E':  
 return time + 1  
 elif cur != '#' or temp == 0:  
 queue.append((time + 1, nx, ny))  
 visited.add((temp, nx, ny))  
 return 'Oop!'  
t = int(input())  
for \_ in range(t):  
 r, c, k = map(int, input().split())  
 maze = [list(input()) for \_ in range(r)]  
 for i in range(r):  
 for j in range(c):  
 if maze[i][j] == 'S':  
 print(bfs(i, j))

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



## 2. 学习总结和收获

这周的正课闫老师讲了笔试部分，听起来没有写代码这么考验脑力（太好了是笔试，我们有救了），收获感满满。好害怕上机。好好复习，保持手感，放平心态，尽量多AC M难度的题目吧，尽力去做！