# Assignment #C: 五味杂陈

Updated 1148 GMT+8 Dec 10, 2024

2024 fall, Complied by <mark>蔡东辰 工学院</mark>

\*\*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 ），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"区有上传的 md 或者 doc 附件。

3）如果不能在截止前提交作业，请写明原因。

## 1. 题目

### 1115. 取石子游戏

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

思路：有了下面的提示就很好写代码了，但如果没有提示的话很难想到两倍以上必胜啊

代码：

```python

while True:

a,b=map(int,input().split())

if a==b==0:

break

can=False

round=0

def stone(a,b,round):

global can

round+=1

M,m=max(a,b),min(a,b)

if (M//m>=2 or M==m) and round%2==1:

can=True

return

if (M//m>=2 or M==m) and round%2==0:

return

M-=m

stone(M,m,round)

stone(a,b,round)

print('win' if can else '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：遍历每一层的和最后比较即可

代码：

```python

n=int(input())

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

layer=[]

for i in range(n//2):

num=0

num+=sum(onion[i][i:n-i])

num+=sum(onion[n-i-1][i:n-i])

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

num+=onion[j][i]

num+=onion[j][n-i-1]

layer.append(num)

if n%2==1:

layer.append(onion[n//2][n//2])

print(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

思路：只要非负就喝，如果小于0但喝完健康值还非负也喝，并记录在neg中，如果喝完了健康值小于0就把当前这杯和堆顶那杯中较小的扔掉，喝较大的

代码：

```python

import heapq

n=int(input())

potion=list(map(int,input().split()))

health=0

num=0

neg=[]

for i in range(n):

if potion[i]>=0:

health+=potion[i]

num+=1

elif potion[i]+health>=0:

health+=potion[i]

num+=1

heapq.heappush(neg,potion[i])

else:

if neg:

health+=potion[i]

heapq.heappush(neg,potion[i])

health-=heapq.heappop(neg)

print(num)

```

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：新进来的猪放进pig里，如果最轻则同时加入lt，不过读取输入的时候如果写a=input()然后在push的情况下转成列表取n=int(a[-1])会wa，不太清楚为啥，输入的数据在最后还会有多余的空格吗？

代码：

```python

pig=[]

lt=[]

while True:

try:

a=input().split()

if a[0]=='pop':

if pig:

b=pig.pop()

if b==lt[-1]:

lt.pop()

elif a[0]=='min':

if pig:

print(lt[-1])

else:

n=int(a[-1])

pig.append(n)

if not lt:

lt.append(n)

elif n<=lt[-1]:

lt.append(n)

except EOFError:

break

```

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：用dfs发现递归深度过大，用bfs发现爆内存，题解用heapq的巧妙之处在于始终选择的是当前体力消耗最少的路线，因此只要发现到达终点那就一定是答案

代码：

```python

import heapq

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

maze=[list(input().split()) for i in range(m)]

def mount(maze,x1,y1,x2,y2):

if maze[x1][y1]=='#' or maze[x2][y2]=='#':

return 'NO'

power=[[float('inf')]\*n for i in range(m)]

power[x1][y1]=0

q=[]

heapq.heappush(q,(0,x1,y1))

dx=[1,-1,0,0]

dy=[0,0,1,-1]

while q:

now\_power,x,y=heapq.heappop(q)

if x==x2 and y==y2:

return now\_power

for i in range(4):

nx,ny=x+dx[i],y+dy[i]

if 0<=nx<m and 0<=ny<n and maze[nx][ny]!='#':

if now\_power+abs(int(maze[x][y])-int(maze[nx][ny]))<power[nx][ny]:

power[nx][ny]=now\_power+abs(int(maze[x][y])-int(maze[nx][ny]))

heapq.heappush(q,(power[nx][ny],nx,ny))

return 'NO'

for i in range(p):

x1,y1,x2,y2=map(int,input().split())

print(mount(maze,x1,y1,x2,y2))

```

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：由于石头时有时无，所以应当允许重复到达相同位置（比如和终点之间只隔了个石头但是暂时过不去，在那里来回逛等到石头没了直接过去就可能比绕路省时），但是如果真的无解会导致q里一直有元素，程序结束不了，所以考虑用visit\_time[x][y]记录每次到达(x,y)时的时间，如果在K的整数倍之前到过这里，情况就与之前重复，所以不允许再去，这样就处理了无解的情况

代码：

```python

from collections import deque

T=int(input())

for i in range(T):

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

maze=[]

start=[]

x0=y0=0

for j in range(R):

a=list(input())

maze.append(a)

if 'S' in a:

x0=j

y0=a.index('S')

def bfs(maze):

q=deque()

visit\_time=[]

for \_ in range(R):

visit\_time.append([[] for l in range(C)])

q.append((x0,y0,0))

visit\_time[x0][y0].append(0)

dx=[0,0,1,-1]

dy=[1,-1,0,0]

while q:

x,y,t=q.popleft()

if maze[x][y]=='E':

return t

for k in range(4):

nx,ny=x+dx[k],y+dy[k]

if 0<=nx<R and 0<=ny<C:

if ((t+1)%K!=0 and maze[nx][ny]!='#') or (t+1)%K==0:

if not visit\_time[nx][ny]:

visit\_time[nx][ny].append(t+1)

q.append((nx,ny,t+1))

else:

visited=False

for \_ in range(len(visit\_time[nx][ny])):

if (t+1-visit\_time[nx][ny][\_])%K==0:

visited=True

if not visited:

visit\_time[nx][ny].append(t+1)

q.append((nx,ny,t+1))

return 'Oop!'

print(bfs(maze))

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

## 2. 学习总结和收获

<mark>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。</mark>

这次作业比前两次仁慈多了，不过还是有一定难度，但至少不像之前怎么都想不出来；主要提高了对heapq的熟练度，了解了dj的用法；变换迷宫从想思路到debug都独立完成，还挺有成就感的，对期末又产生了一丁点信心，周末再做些每日选做