# Assignment #D: 十全十美

Updated 1254 GMT+8 Dec 17, 2024

2024 fall, Complied by <mark>祁黄奕 城市与环境学院</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

### 02692: 假币问题

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

思路：创建存储状态信息的 dic 字典和包含所有硬币编号的 total 集合。locate 函数通过 split 来判断硬币位置。遍历称量结果，将平衡时的硬币放入 normal 集。计算出可能为假币的 uncertain 集。对于 uncertain 中的硬币，遍历称量结果，若非平衡，通过 locate 找位置并计算状态，根据状态首次赋值或判断一致性，不一致则标记，未标记的为假币。

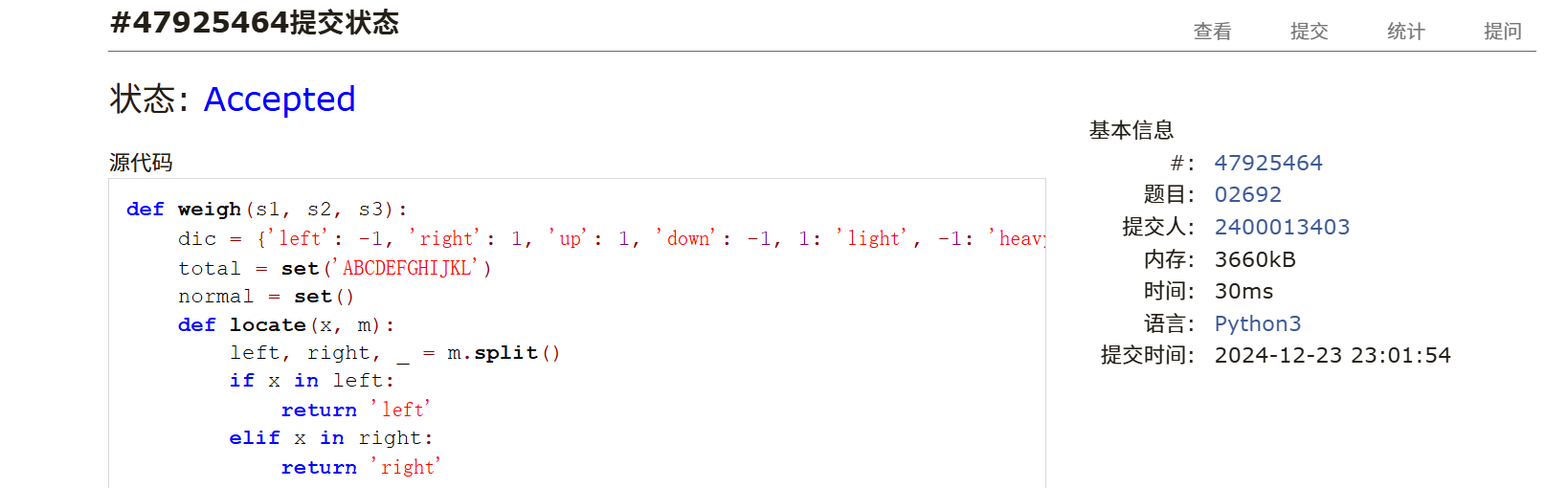
代码：

```python

def weigh(s1, s2, s3):  
 dic = {'left': -1, 'right': 1, 'up': 1, 'down': -1, 1: 'light', -1: 'heavy'}  
 total = set('ABCDEFGHIJKL')  
 normal = set()  
 def locate(x, m):  
 left, right, \_ = m.split()  
 if x in left:  
 return 'left'  
 elif x in right:  
 return 'right'  
 else:  
 return 0  
 for i in (s1, s2, s3):  
 if i.split()[-1] == 'even':  
 for j in i:  
 if 'A' <= j <= 'L':  
 normal.add(j)  
 uncertain = total - normal  
 for i in uncertain:  
 heavy\_light = None  
 break\_flag = False  
 for j in (s1, s2, s3):  
 left, right, result = j.split()  
 if result == 'even':  
 continue  
 location = locate(i, j)  
 if location == 0:  
 break\_flag = True  
 break  
 res = dic[location] \* dic[result]  
 if heavy\_light is None:  
 heavy\_light = dic[res]  
 elif heavy\_light!= dic[res]:  
 break\_flag = True  
 break  
 if not break\_flag:  
 return f'{i} is the counterfeit coin and it is {heavy\_light}.'  
  
n = int(input())  
ans = []  
for \_ in range(n):  
 a = str(input()).rstrip()  
 b = str(input()).rstrip()  
 c = str(input()).rstrip()  
 ans.append(weigh(a, b, c))  
print(\*ans, sep='\n')

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 01088: 滑雪

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

思路：用dfs搜索找矩阵中最长递增路径，采用记忆化搜索，在dfs里先判断是否已算过。

代码：

```python

r, c = map(int, input().split())  
matrix = [list(map(int, input().split())) for \_ in range(r)]  
dp = [[0 for \_ in range(c)] for \_ in range(r)]  
  
def dfs(x, y):  
 if dp[x][y]!= 0:  
 return dp[x][y]  
 directions = [(0, 1), (0, -1), (1, 0), (-1, 0)]  
 max\_path = 1  
 for dx, dy in directions:  
 new\_x, new\_y = x + dx, y + dy  
 if 0 <= new\_x < r and 0 <= new\_y < c and matrix[x][y] > matrix[new\_x][new\_y]:  
 path\_length = dfs(new\_x, new\_y) + 1  
 max\_path = max(max\_path, path\_length)  
 dp[x][y] = max\_path  
 return max\_path  
  
ans = 0  
for row in range(r):  
 for col in range(c):  
 ans = max(ans, dfs(row, col))  
print(ans)

```

代码运行截图 ==（至少包含有"Accepted"）==

文本

中度可信度描述已自动生成

### 25572: 螃蟹采蘑菇

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

思路：先读取迷宫信息及小呆起始坐标，计算其两部分坐标差值。通过定义合法位置判断函数，利用栈模拟小呆移动探索，按四个方向尝试。

代码：

```python

n = int(input())  
maze = [list(map(int, input().split())) for \_ in range(n)]  
start = []  
  
for i in range(n):  
 for j in range(n):  
 if maze[i][j] == 5:  
 start.append((i, j))  
  
delta\_x = start[1][0] - start[0][0]  
delta\_y = start[1][1] - start[0][1]  
visited = set()  
visited.add((start[0][0], start[0][1]))  
  
def is\_valid(r, c):  
 return (0 <= r < n and 0 <= c < n and (r, c) not in visited   
 and maze[r][c]!= 1 and 0 <= r + delta\_x < n and 0 <= c + delta\_y < n   
 and maze[r + delta\_x][c + delta\_y]!= 1)  
  
dx = [0, 0, 1, -1]  
dy = [1, -1, 0, 0]  
stack = [(start[0][0], start[0][1])]  
while stack:  
 front\_x, front\_y = stack.pop()  
  
 if maze[front\_x][front\_y] == 9 or maze[front\_x + delta\_x][front\_y + delta\_y] == 9:  
 print('yes')  
 break  
 for i in range(4):  
 new\_x, new\_y = front\_x + dx[i], front\_y + dy[i]  
 if is\_valid(new\_x, new\_y):  
 visited.add((new\_x, new\_y))  
 stack.append((new\_x, new\_y))  
else:  
 print('no')

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

图片包含 文本

描述已自动生成

### 27373: 最大整数

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

思路：先对数据进行预处理，再进行dp，由于一个数只能用一次，所以添加了一个temp来缓存。

代码：

```python

limit=int(input())  
n=int(input())  
lst=sorted(list(map(str,input().split())),reverse=True)  
  
def sorting(a,b):  
 if int(a+b)>int(b+a):  
 return True  
 else:  
 return False  
for i in range(n):  
 for j in range(i,n):  
 if not sorting(lst[i],lst[j]):  
 lst[i],lst[j]=lst[j],lst[i]  
  
dp=[0]\*(limit+1)  
temp=[0]\*(limit+1)  
for j in lst:  
 for i in range(len(j),limit+1):  
 dp[i]=max(dp[i],int(str(temp[i-len(j)])+j))  
 temp[:]=dp  
print(dp[limit])

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

图片包含 图形用户界面

描述已自动生成

### 02811: 熄灯问题

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

思路：遍历第一行所有可能，再根据提示解决二到五行。

代码：

```python

from copy import deepcopy  
  
matrix=[list(map(int,input().split())) for \_ in range(5)]  
target=[[0 for \_ in range(6)] for \_ in range(5)]  
  
def change(x,y):  
 if cnt[x][y]==1:  
 cnt[x][y]=0  
 else:  
 cnt[x][y]=1  
  
def turn(x,y):  
 dx=[-1,1,0,0,0]  
 dy=[0,0,-1,1,0]  
 for o in range(5):  
 nx,ny=dx[o]+x,dy[o]+y  
 if 0<=nx<5 and 0<=ny<6:  
 change(nx,ny)  
  
t=[0,1]  
for a in t:  
 for b in t:  
 for c in t:  
 for d in t:  
 for e in t:  
 for f in t:  
 ans=deepcopy(target)  
 cnt=deepcopy(matrix)  
 ans[0][:]=[a,b,c,d,e,f]  
 for j in range(6):  
 if ans[0][j]==1:  
 turn(0,j)  
 for i in range(4):  
 for j in range(6):  
 if cnt[i][j]==1:  
 ans[i+1][j]=1  
 turn(i+1,j)  
 if cnt==target:  
 for i in range(5):  
 print(\*ans[i])

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

图片包含 文本

描述已自动生成

### 08210: 河中跳房子

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

思路：算出相邻点间距存于delta。chec函数判断给定距离下分割数是否超限制。接着用二分查找，不断调整查找区间，最后依据check的结果输出，找到满足分割限制的最大距离值。

代码：

```python

end,n,limit=map(int,input().split())  
pre=0  
delta=[]  
for \_ in range(n):  
 x=int(input())  
 delta.append(x-pre)  
 pre=x  
delta.append(end-pre)  
  
def check(distance):  
 num=0  
 cumulative=0  
 for i in range(n+1):  
 if delta[i]+cumulative>=distance:  
 cumulative=0  
 continue  
 cumulative+=delta[i]  
 num+=1  
 if num<=limit:  
 return True  
 else:  
 return False  
  
low=0  
high=end  
mid=0  
while low<high-1:  
 mid=(low+high)//2  
 if check(mid):  
 low=mid  
 else:  
 high=mid  
if check(mid):  
 print(mid)  
else:  
 print(mid-1)

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

图形用户界面, 应用程序

中度可信度描述已自动生成

## 2. 学习总结和收获

这次作业每一题都难啊难难难！马上就上机考试了真的求善良求放过呜呜……且笔试要记的内容也好多呀……依然在做最后的努力（垂死挣扎），疑似失去所有力气和手段。