# Assignment #A: dp & bfs

Updated 2 GMT+8 Nov 25, 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

### LuoguP1255 数楼梯

dp, bfs, https://www.luogu.com.cn/problem/P1255

思路：建立n+1长度的dp列表，当n=0或n=1时只有一种上台阶方法。n>=2时，当前阶的方法数等于前一阶的方法数加上前两阶的方法数之和。

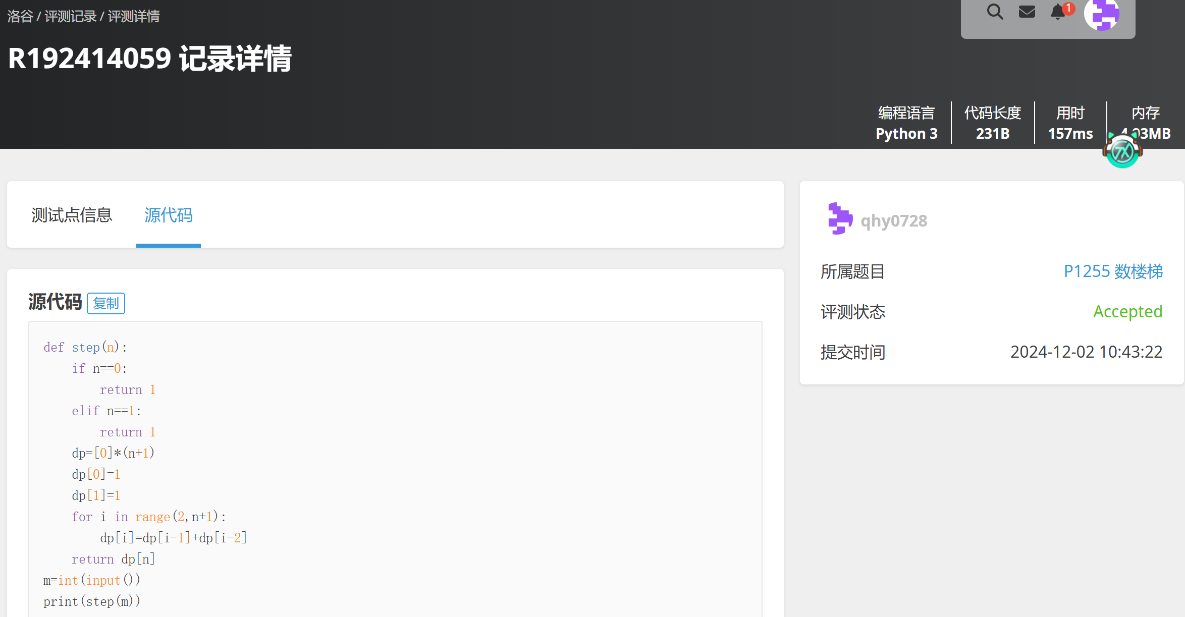
代码：

```python

def step(n):  
 if n==0:  
 return 1  
 elif n==1:  
 return 1  
 dp=[0]\*(n+1)  
 dp[0]=1  
 dp[1]=1  
 for i in range(2,n+1):  
 dp[i]=dp[i-1]+dp[i-2]  
 return dp[n]  
m=int(input())  
print(step(m))

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 27528: 跳台阶

dp, http://cs101.openjudge.cn/practice/27528/

思路：dp数组，到达第n台阶的方法数等于到达n-1级台阶的方法数加上到达n-2级台阶的方法数，依此类推，直到到达地面（即第0级台阶）。

代码：

```python

def step(n):  
 if n==0:  
 return 1  
 dp=[0]\*(n+1)  
 dp[0]=1  
 for i in range(1,n+1):  
 for j in range(i):  
 dp[i]+=dp[j]  
 return dp[n]  
m=int(input())  
print(step(m))

```

代码运行截图 ==（至少包含有"Accepted"）==

图形用户界面, 应用程序

描述已自动生成

### 474D. Flowers

dp, https://codeforces.com/problemset/problem/474/D

思路：

代码：

```python

MAX = 1000000007  
t, k = map(int, input().split())  
MOD = int(1e9+7)  
MAXN = 100001  
dp = [0]\*MAXN  
s = [0]\*MAXN  
dp[0] = 1  
s[0] = 1  
for i in range(1, MAXN):  
 if i >= k:  
 dp[i] = (dp[i-1]+dp[i-k]) % MOD  
 else:  
 dp[i] = dp[i-1] % MOD  
 s[i] = (s[i-1]+dp[i]) % MOD  
  
for \_ in range(t):  
 a, b = map(int, input().split())  
 print((s[b]-s[a-1]+MOD) % MOD)

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### LeetCode5.最长回文子串

dp, two pointers, string, https://leetcode.cn/problems/longest-palindromic-substring/

思路：在处理回文子串问题时，需要考虑奇数和偶数两种情况。首先设定一个空字符串作为最大回文子串res = ''，然后逐步更新。遍历数组，使用指针i向前寻找长度大于当前res的回文子串，注意定位到开头的前一位start = max(0, i-len(s)-1)。根据索引范围切片判断是否为回文子串，分为奇数和偶数两种情况。命中条件则更新res，最终得到最长回文子串。

代码：

```python

class Solution(object):  
 def longestPalindrome(self, s):  
 res = ''  
 for i in range(len(s)):  
 start = max(i - len(res) -1, 0)  
 temp = s[start: i+1]  
 if temp == temp[::-1]:  
 res = temp  
 else:  
 temp = temp[1:]  
 if temp == temp[::-1]:  
 res = temp  
 return res

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

电脑萤幕的截图

描述已自动生成

### 12029: 水淹七军

bfs, dfs, http://cs101.openjudge.cn/practice/12029/

思路：

代码：

```python

from collections import deque  
import sys  
MAXN=1111  
def bfs(x,y):  
 global slb,mat  
 q=deque([(x,y)])  
 d=[(-1,0),(0,1),(1,0),(0,-1)]  
 while q:  
 x,y=q.popleft()  
 for dx,dy in d:  
 nx,ny=x+dx,y+dy  
 if 1<=nx<=m and 1<=ny<=n \  
 and mat[nx][ny]<mat[x][y]:  
 q.append((nx,ny))  
 mat[nx][ny]=mat[x][y]  
 if (nx,ny)==slb:  
 return('Yes')  
 return ('No')  
lines=list(sys.stdin.read().split())  
  
t=int(lines[0])  
id=1  
for \_ in range(t):  
 m,n=map(int,(lines[id],lines[id+1]))  
 mat=[[MAXN]\*(n+2)]  
 id+=2  
 for i in range(m):  
 mat.append([MAXN]+list(map(int,lines[id:id+n]))+[MAXN])  
 id+=n  
 mat.append([MAXN]\*(n+2))  
 slb=(int(lines[id]),int(lines[id+1]))  
 id+=2  
 k=int(lines[id])  
 id+=1  
 H2O=[]  
 for i in range(k):  
 H2O.append((int(lines[id]),int(lines[id+1])))  
 id+=2  
 for h20 in H2O:  
 ans=bfs(h20[0],h20[1])  
 if ans=='Yes':  
 print(ans)  
 break  
 else:print('No')

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



### 02802: 小游戏

bfs, http://cs101.openjudge.cn/practice/02802/

思路：可以用递归方法求解通过穷举方式找到从起点到终点的路径，朝一个方向走下去，如果走不通，则换一个方向走;四个方向都走不通，则回到上一步的地方，再换一个方向走;依次走下去，直到走到终点。该问题的路径只包含水平或者竖直的直线段，所以需要记录每一步走的方向。如果上一步走的方向和这一步走的方向相同，递归搜索时路径数不变，否则路径数加1。

另外根据题意，路径只包含水平或者竖直的直线段，路径不能穿过别的游戏卡片，但是允许路径临时离开矩形板，所以可以在矩形板最外层增加一圈格子，路径可以通过这些新增加的格子可以设置一个二维数组来表示迷宫(空格表示无游戏卡片，X表示有游戏卡片)，用另一个二维数组来标记某个位置的格子是否已经被走过了。

代码：

```python

from collections import deque  
  
  
def bfs(start, end, grid, h, w):  
 queue = deque([start])  
 visited = set()  
 dirs = [(0, -1), (-1, 0), (0, 1), (1, 0)]  
  
 ans = []  
 while queue:  
 x, y, d\_i\_r, seg = queue.popleft()  
 # print(x,y,end)  
 if (x, y) == end:  
 # return seg  
 ans.append(seg)  
 break  
  
 for i, (dx, dy) in enumerate(dirs):  
 nx, ny = x + dx, y + dy  
  
 if 0 <= nx < h + 2 and 0 <= ny < w + 2 and ((nx, ny, i) not in visited):  
 new\_dir = i  
 new\_seg = seg if new\_dir == d\_i\_r else seg + 1  
 if (nx, ny) == end:  
 # return new\_seg  
 ans.append(new\_seg)  
 continue  
  
 if grid[nx][ny] != 'X':  
 visited.add((nx, ny, i))  
 queue.append((nx, ny, new\_dir, new\_seg))  
  
 if len(ans) == 0:  
 return -1  
 else:  
 return min(ans)  
  
  
board\_num = 1  
while True:  
 w, h = map(int, input().split())  
 if w == h == 0:  
 break  
  
 # grid = [[' '] \* (w + 2)] + \  
 # [[' '] + list(input()) + [' '] for \_ in range(h)] + \  
 # [[' '] \* (w + 2)]  
 grid = [' ' \* (w + 2)] + [' ' + input() + ' ' for \_ in range(h)] + [' ' \* (w + 2)]  
 print(f"Board #{board\_num}:")  
 pair\_num = 1  
 while True:  
 y1, x1, y2, x2 = map(int, input().split())  
 if x1 == y1 == x2 == y2 == 0:  
 break  
  
 start = (x1, y1, -1, 0)  
 end = (x2, y2)  
  
 seg = bfs(start, end, grid, h, w)  
 if seg == -1:  
 print(f"Pair {pair\_num}: impossible.")  
 else:  
 print(f"Pair {pair\_num}: {seg} segments.")  
 pair\_num += 1  
  
 print()  
 board\_num += 1

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



## 2. 学习总结和收获

这次做下来依然要感叹好难好难好难！打算好好啃一下dp的easy和middle题目，熟悉模板，期末遇到觉得难的题目（特别是像最后两题，做作业的时候看完ai和题解还是不懂……）会直接放弃，显然需要再多做点题，并训练一下数学思维。日常希望能过！