*# Assignment #10: dp & bfs*

Updated 2 GMT+8 Nov 25, 2024

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

*## 1. 题目*

*### LuoguP1255 数楼梯*

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

思路：

DP经典题目，没有问题。

代码：

n=int(input())  
dp = [0] \* (n + 3)  
dp[1] = 1  
dp[2] = 2  
for i in range(3, n + 1):  
 dp[i]=dp[i-1]+dp[i-2]  
print(dp[n])

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/

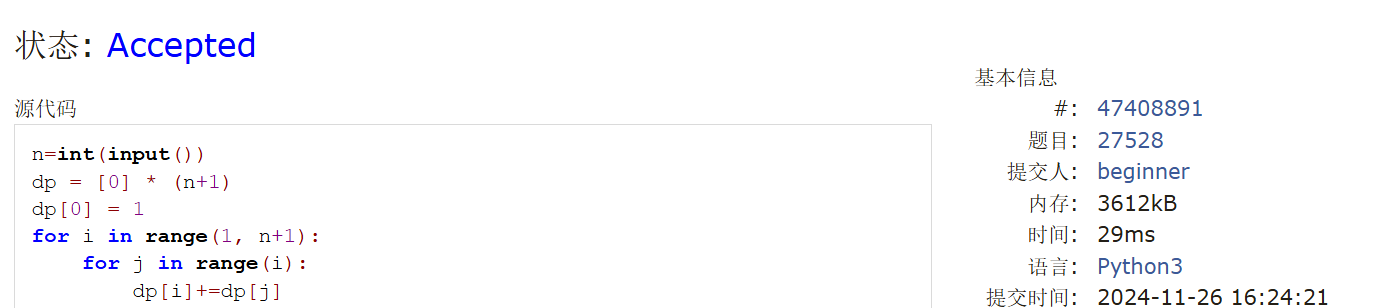
思路：

dp关系变为:dp[i] = dp[i-1] + dp[i-2] + ... + dp[0]，本质是一样的

代码：

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

代码运行截图 ==（至少包含有"Accepted"）==



*### 474D. Flowers*

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

思路：

K=2时，可以推出来n\*\*2//4+1；但是超时。dp关系找不出来，，看题解有点糊涂。

代码：

MOD = 1000000007  
def solve(t, k, queries):  
 max\_n = 100000  
 dp = [0] \* (max\_n + 1)  
 dp[0] = 1  
 for n in range(1, max\_n + 1):  
 dp[n] = dp[n - 1] # 如果直接吃红花的选择  
 if n >= k:  
 dp[n] = (dp[n] + dp[n - k]) % MOD # 加上成组的白花的选择  
 # 计算前缀和  
 prefix\_sum = [0] \* (max\_n + 1)  
 for i in range(1, max\_n + 1):  
 prefix\_sum[i] = (prefix\_sum[i - 1] + dp[i]) % MOD  
 # 返回查询结果  
 result = []  
 for a, b in queries:  
 if a > 0:  
 result.append((prefix\_sum[b] - prefix\_sum[a - 1]) % MOD)  
 else:  
 result.append(prefix\_sum[b] % MOD)  
 return result  
  
import sys  
input = sys.stdin.read  
data = input().splitlines()  
t, k = map(int, data[0].split())  
queries = [tuple(map(int, line.split())) for line in data[1:t + 1]]  
results = solve(t, k, queries)  
sys.stdout.write('\n'.join(map(str, results)) + '\n')

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

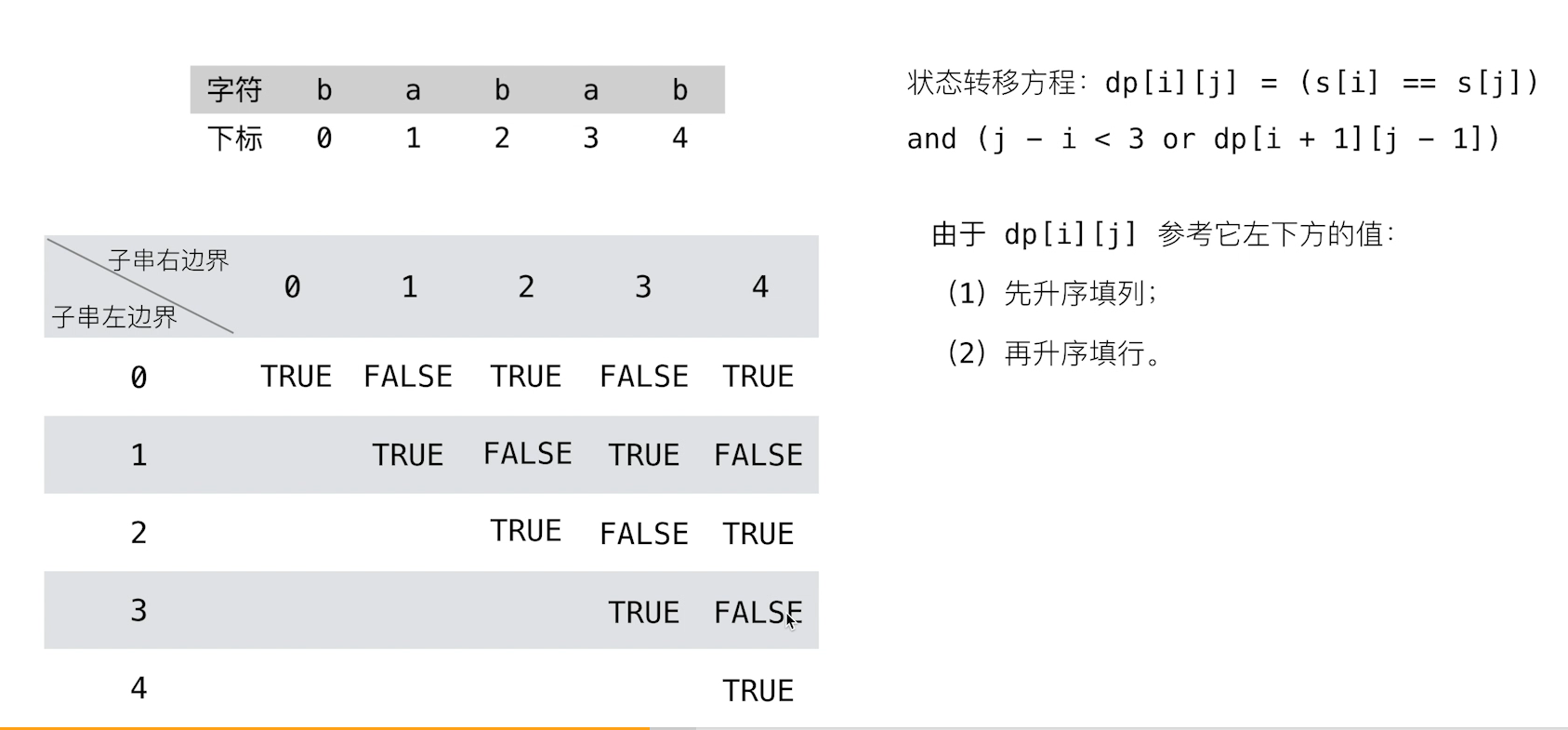


*### LeetCode5.最长回文子串*

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

思路：

看的题解，这张图很清楚！



代码：

class Solution:  
 def longestPalindrome(self, s: str) -> str:  
 n = len(s)  
 if n < 2:  
 return s  
 max\_len = 1  
 begin = 0  
 # dp[i][j] 表示 s[i..j] 是否是回文串  
 dp = [[False] \* n for \_ in range(n)]  
 for i in range(n):  
 dp[i][i] = True  
  
 # 递推开始  
 # 先枚举子串长度  
 for L in range(2, n + 1):  
 # 枚举左边界，左边界的上限设置可以宽松一些  
 for i in range(n):  
 # 由 L 和 i 可以确定右边界，即 j - i + 1 = L 得  
 j = L + i - 1  
 if j >= n:  
 break  
 if s[i] != s[j]:  
 dp[i][j] = False  
 else:  
 if j - i < 3:  
 dp[i][j] = True  
 else:  
 dp[i][j] = dp[i + 1][j - 1]  
  
 # 只要 dp[i][L] == true 成立，就表示子串 s[i..L] 是回文，此时记录回文长度和起始位置  
 if dp[i][j] and j - i + 1 > max\_len:  
 max\_len = j - i + 1  
 begin = i  
 return s[begin:begin + max\_len]

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### 12029: 水淹七军*

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

思路：

看题解的bfs实现法。

代码：

**from** collections **import** deque

**import** sys

input = sys.stdin.read

**def** **is\_valid**(x, y, m, n):

**return** 0 <= x < m **and** 0 <= y < n

**def** **bfs**(start\_x, start\_y, start\_height, m, n, h, water\_height):

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

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

q = **deque**([(start\_x, start\_y, start\_height)])

water\_height[start\_x][start\_y] = start\_height

**while** q:

x, y, height = q.**popleft**()

**for** i **in** **range**(4):

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

**if** **is\_valid**(nx, ny, m, n) **and** h[nx][ny] < height:

**if** water\_height[nx][ny] < height:

water\_height[nx][ny] = height

q.**append**((nx, ny, height))

**def** **main**():

data = **input**().**split**()

idx = 0

k = **int**(data[idx])

idx += 1

results = []

**for** \_ **in** **range**(k):

m, n = **map**(int, data[idx:idx + 2])

idx += 2

h = []

**for** i **in** **range**(m):

h.**append**(**list**(**map**(int, data[idx:idx + n])))

idx += n

water\_height = [[0] \* n **for** \_ **in** **range**(m)]

i, j = **map**(int, data[idx:idx + 2])

idx += 2

i, j = i - 1, j - 1

p = **int**(data[idx])

idx += 1

**for** \_ **in** **range**(p):

x, y = **map**(int, data[idx:idx + 2])

idx += 2

x, y = x - 1, y - 1

**if** h[x][y] <= h[i][j]:

**continue**

**bfs**(x, y, h[x][y], m, n, h, water\_height)

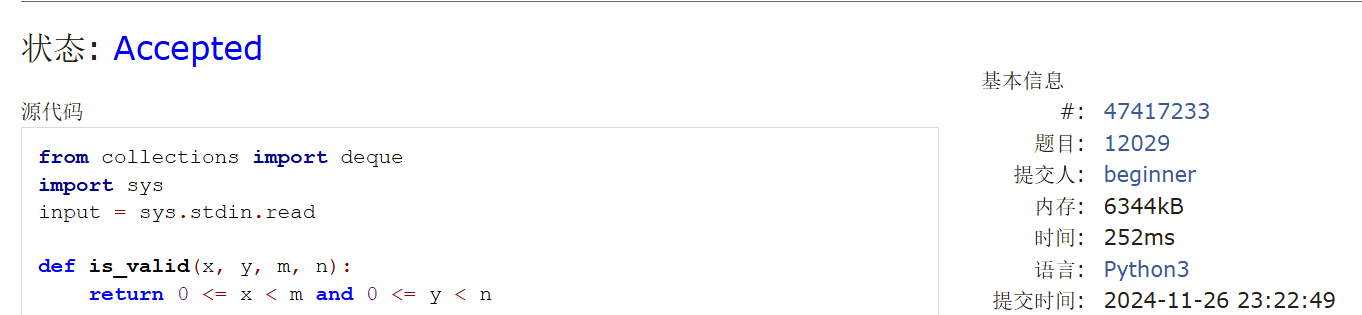
results.**append**("Yes" **if** water\_height[i][j] > 0 **else** "No")

sys.stdout.**write**("\n".**join**(results) + "\n")

**if** \_\_name\_\_ == "\_\_main\_\_":

**main**()

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/

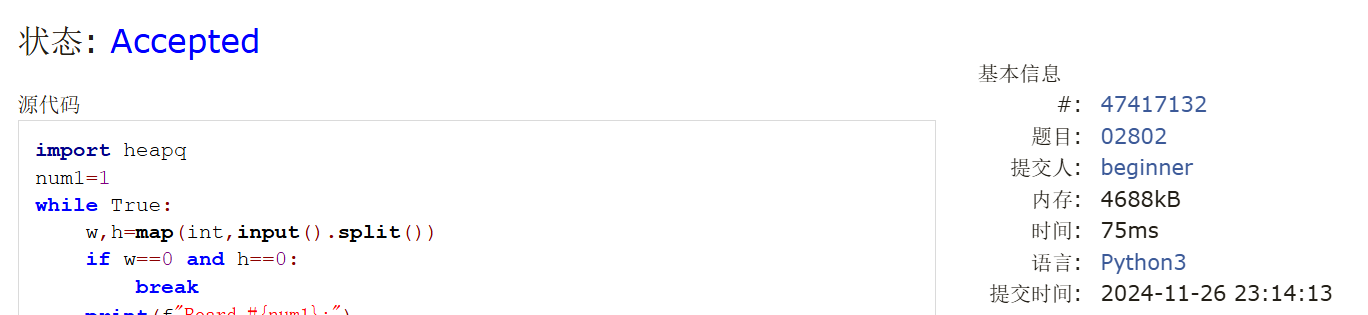
思路：

一眼做不出来，直接学习题解代码。

代码：

import heapq  
num1=1  
while True:  
 w,h=map(int,input().split())  
 if w==0 and h==0:  
 break  
 print(f"Board #{num1}:")  
 martix=[[" "]\*(w+2)]+[[" "]+list(input())+[" "] for \_ in range(h)]+[[" "]\*(w+2)]  
 dir=[(0,1),(0,-1),(1,0),(-1,0)]  
 num2=1  
 while True:  
 x1,y1,x2,y2=map(int,input().split())  
 if x1==0 and x2==0 and y1==0 and y2==0:  
 break  
 queue,flag=[],False  
 vis=set()  
 heapq.heappush(queue,(0,x1,y1,-1))  
 martix[y2][x2]=" "  
 vis.add((-1,x1,y1))  
 while queue:  
 step,x,y,dirs=heapq.heappop(queue)  
 if x==x2 and y==y2:  
 flag=True  
 break  
 for i,(dx,dy) in enumerate(dir):  
 px,py=x+dx,y+dy  
 if 0<=px<=w+1 and 0<=py<=h+1 and (i,px,py) not in vis and martix[py][px]!="X":  
 vis.add((i,px,py))  
 heapq.heappush(queue,(step+(dirs!=i),px,py,i))  
 if flag:  
 print(f"Pair {num2}: {step} segments.")  
 else:  
 print(f"Pair {num2}: impossible.")  
 martix[y2][x2]="X"  
 num2+=1  
 print()  
 num1+=1

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*## 2. 学习总结和收获*

我觉得要把学习策略转向了，很清楚一些难题自己很难理解，即使当时明白了考场上也很难做出来。不在难题上花费很多时间，大概能懂题解就行。省下来的时间开始做一些类似往年期末中等难度的题目和比较套路化、模板化的较难题目。