HW#11

经济学院 刘安澜 1700015495

1759: 最长上升子序列

dp, <http://cs101.openjudge.cn/practice/1759>

解题思路:参考最大上升子序列和的思路，简单改写，将最大子序列和的列表改为最长子序列长度的列表即可。

﻿﻿n=int(input())

l=[int(x) for x in input().split()]

s=[1]\*n

for i in range(1,n):

for j in range(i):

if l[i]>l[j]:

s[i]=max(s[j]+1,s[i])

print(max(s))

1700:八皇后问题解输出  
dfs, http://cs101.openjudge.cn/practice/1700/

解题思路：注意到是按照列的顺序输出，所以批量更改输出的矩阵；且前面需要加上格式

﻿def queen(A,cur=0):

if cur==len(A):

l.append(A[:])

return 0

for i in range(len(A)):

A[cur],flag=i,True

for j in range(cur):

if A[j]==A[cur] or abs(A[j]-A[cur])==cur-j:

flag=False

break

if flag:

queen(A,cur+1)

l=[]

queen([None]\*8)

for i in range(92):

print('No.',i+1)

m=[[0]\*8 for i in range(8)]

for j in range(8):

m[l[i][j]][j]=1

for j in range(8):

print(' '.join([str(x) for x in m[j]]))

368B. Sereja and Suffixes  
data structures/dp, 1100, https://codeforces.com/problemset/problem/368/B

解题思路：仍然是考虑构造一个答案列表，所以先将原列表倒置，运用集合统计不相同元素的个数，最后反向提取列表答案即可。

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

l=[int(x) for x in input().split()]

l.reverse()

s={l[0]}

l2=[1]\*n

for i in range(n-1):

if l[i+1] not in s:

s.add(l[i+1])

l2[i+1]=l2[i]+1

else:

l2[i+1]=l2[i]

for i in range(m):

print(l2[n-int(input())])

313B. Ilya and Queries  
dp, 1300, https://codeforces.com/contest/313/problem/B

解题思路：dp的思路仍然是构造答案列表，l[i]是截止到第i个元素的相同相邻元素个数，最后答案输出只需要将对应区间首尾的答案列表相减即可。但是写的过程当中需要额外注意列表的下标。

s=input()

n=len(s)

l1=[0]\*n

for i in range(n-1):

if s[i]==s[i+1]:

l1[i+1]+=1

for i in range(n-2):

l1[i+2]+=l1[i+1]

m=int(input())

for i in range(m):

l,r=map(int,input().split())

print(l1[r-1]-l1[l-1])

//选做：

18108:池塘数目（cs101-2016 期末机考备选）  
matrix,dfs, http://cs101.openjudge.cn/practice/18108/

解题思路：本题是这次作业最难的一道题，主要还是对dfs的思路和构建相应的函数形式不熟悉。思路是先找到一个W，然后以这个点为出发点将所有关联的W找到并做标记；然后再遍历列表寻找其他的W并继续回溯。难点主要在于定义回溯的函数：需要先确立终止条件，即如果检索的元素不是W就可以终止，return终止；如果是W，要先将这个点做好标记，然后再对周围的元素进行上述回溯。

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

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

def dfs(i,j):

if l[i][j]!='W':

return

l[i][j]='M'

for s in range(8):

dfs(i+dx[s],j+dy[s])

T=int(input())

for i in range(T):

N,M=map(int,input().split())

l=[[0]\*(M+2)]+[[0]+list(input())+[0] for i in range(N)]+[[0]\*(M+2)]

p=[[0]\*(M+2) for i in range(N+2)]

ans=0

for i in range(1,N+1):

for j in range(1,M+1):

if l[i][j]=='W':

ans+=1

dfs(i,j)

print(ans)

1762:数字三角形  
dp, http://cs101.openjudge.cn/practice/1762/

解题思路：最开始学习dp思路的时候在网上看了一个案例分析就是数字三角形，所以按照dp的思路很快写出来了代码。对于这种有明确回溯对象的题目来说还是比较容易的：在三角形中，构建最大序列和的列表，选取元素上方两侧最大的序列和加上元素的值即可得到新的列表。

解题代码：

﻿n=int(input())

l1=[int(x) for x in input().split()]

for i in range(n-1):

l2=[int(x) for x in input().split()]

l3=[l2[0]+l1[0]]+[max(l1[i],l1[i+1])+l2[i+1] for i in range(len(l2)-2)]+[l2[-1]+l1[-1]]

l1=l3

print(max(l1))

90:滑雪  
dp/dfs, http://cs101.openjudge.cn/practice/90/

解题思路：

滑雪这道dp题目的思路其实还算比较容易，但是处理上面细节比较多，而且也反映出我对dp的函数写法上还是不熟练。滑雪不同于数字三角形这种有明确的回溯方向的dp，滑雪需要充分遍历每一个路径，所以需要在回溯的函数中再嵌套回溯的函数，保证遍历的充分性。但是，如果不加回溯函数的停止条件，本题会对每一条路径都进行回溯，就会非常超时；所以对于已经回溯完成的元素，就可以直接返回。

我对老师的代码进行了一个改进，因为本题说高度最大值不超过10000，所以保护圈元素取10001即可满足不超过边界，这样函数更加简洁。

解题代码：

﻿r,c=map(int,input().split())

l=[[10001]\*(c+2)]+[[10001]+[int(x) for x in input().split()]+[10001] for i in range(r)]+[[10001]\*(c+2)]

output=[[0]\*(c+2) for i in range(r+2)]

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

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

def dp(i,j):

if output[i][j]>0:

return output[i][j]

for s in range(4):

if l[i][j]>l[i+dx[s]][j+dy[s]]:

output[i][j]=max(output[i][j],dp(i+dx[s],j+dy[s])+1)

return output[i][j]

ans=0

for i in range(1,r+1):

for j in range(1,c+1):

ans=max(ans,dp(i,j))

print(ans+1)