HW#9

经济学院 刘安澜 1700015495

12560: 生存游戏

matrix, <http://cs101.openjudge.cn/practice/12560/>, cs10116 final exam

解题思路：首先先写出生存规则对应的函数，然后对数据构造一个保护圈，建立新的答案列表对列表遍历即可。小tips：我在建立列表时一开始多加了一个括号，造成程序无法运行，对这种多个数组嵌套的语句要格外注意括号的个数。另外本题提供了二维数组的矩阵输出格式。

﻿﻿def game(board,x,y):

c=board[x-1][y-1]+board[x-1][y]+board[x-1][y+1]+board[x][y-1]+board[x][y+1]+board[x+1][y-1]+board[x+1][y]+board[x+1][y+1]

if board[x][y]==1:

if c<2 or c>3:

return 0

else:

return 1

else:

if c==3:

return 1

else:

return 0

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

board=[[0]\*(m+2)]+[[0]+[int(x) for x in input().split()]+[0] for j in range(n)]+[[0]\*(m+2)]

output=[[0]\*m for x in range(n)]

for i in range(n):

for j in range(m):

output[i][j]=game(board,i+1,j+1)

for y in range(n):

print(' '.join([str(x) for x in output[y]]))

18182: 打怪兽

sorting/math, <http://cs101.openjudge.cn/practice/18182/>

解题思路：先按照时刻的升序和技能的降序排序；遍历的过程中对于同一个时刻的技能计数，到m次为止；特别需要注意的是，在这个循环中如果恰好在最后一步前后b变为负数，需要判断是否是在最后一次攻击后b为负来确定攻击结束的时间。在这个循环中，是先循环数+1再判断b的正负，所以如果恰好在倒数第二步攻击结束，此时j就已经变为n-1，最初的写法是将最后一步判断写在循环外，这样就导致了WA。所以以后对于这种边界的特殊情况一定要格外注意！

﻿cases=int(input())

for i in range(cases):

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

l=[]

for j in range(n):

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

l.sort(key=lambda x:(x[0],-x[1]))

k=1

b-=l[0][1]

for j in range(1,n):

if b<=0:

break

if l[j][0]==l[j-1][0]:

k+=1

else:

k=1

if k<=m:

b-=l[j][1]

if j==n-1 and b<=0:

j+=1

print(l[j-1][0] if b<=0 else 'alive')

//选做：

189A. Cut Ribbon

brute force/dp, 1300, <https://codeforces.com/problemset/problem/189/A>

解题思路：

一看上去这个题和老师上课讲的dp是一个镜像关系的题目，所以就按照找硬币的dp思路写了最初始的一版。但是不同于找硬币必有一个解，剪丝带对于某些丝带长度是无法分割的，所以这也就造成了在某些不能分割的长度上输出错误的结果。所以对于这种情况需要我们将除0外所有的初始值都赋为一个很大的负数，这样就能很好避免不能分割的长度对于其它可分割长度答案的扰动。

解题代码：

﻿def dpcut(lengthlist,l,maxcuts):

for i in range(l+1):

for j in [c for c in lengthlist if c<=i]:

if maxcuts[i-j]+1>maxcuts[i]:

maxcuts[i]=maxcuts[i-j]+1

return maxcuts[l]

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

lengthlist=list({a,b,c})

maxcuts=[0]+[-1000000]\*(l)

print(dpcut(lengthlist,l,maxcuts))

455A. Boredom

dp, 1500, <https://codeforces.com/contest/455/problem/A>

解题思路：

这道题目答案的思路太巧妙了！我之前思考的dp是考虑最后一步剩什么单一数字再一步步往前推，但是这个并不好操作。但是答案给的思路是按照数的大小一步步往后推，并且+1和-1体现在如果选择下一步的数字是n，那么只保留n-2以前的分数和n所得的分数（-1消除条件）；如果下一步选择的数字是n-1，那么n也被消除，所以分数仍然是n-1的分数（+1消除条件）。如果说找硬币是一种“逆向”的dp思路的话，其实本题更类似于一种正向的dp思路：从n-1的情况出发去考虑n的最优解。

解题代码：

﻿n=int(input())

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

l2=[0]\*(max(l1)+1)

for i in l1:

l2[i]+=1

f=[0]\*(max(l1)+1)

for i in range(max(l1)+1):

f[i]=max(f[i-1],f[i-2]+i\*l2[i])

print(f[max(l1)])

706B. Interesting drink

binary search/dp/implementation, 1100, <https://codeforces.com/problemset/problem/706/B>

解题思路：

本题的思路比较简单，仍然是构造一个列表方便查询。但是我自己尝试的时候因为又写了一个O（n2）的循环所以超时；后来参考答案中的方便写法，直接对列表遍历相加，大大缩短了复杂度。另外，我之前习惯于对一些简单变量不存储，每一步都重新调用，但对于max（列表）这种函数，每次调用都非常耗时，所以需要将它存储起来，就可以完全避免超时的问题。

解题代码：

﻿n=int(input())

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

maxl=max(l)

l2=[0]\*(maxl+1)

for i in l:

l2[i]+=1

for j in range(2,maxl+1):

l2[j]+=l2[j-1]

q=int(input())

output=[]

for i in range(q):

m=int(input())

if m>=maxl:

output.append(str(n))

else:

output.append(str(l2[m]))

print('\n'.join(output))