期中复习————动态规划

数字三角形 Number Triangles

题目描述

观察下面的数字金字塔。

写一个程序来查找从最高点到底部任意处结束的路径，使路径经过数字的和最大。每一步可以走到左下方的点也可以到达右下方的点。

```
        7
      3   8
    8   1   0
  2   7   4   4
4   5   2   6   5
```

在上面的样例中,从 $7 \to 3 \to 8 \to 7 \to 5$ 的路径产生了最大

输入格式

第一个行一个正整数 r ,表示行的数目。

后面每行为这个数字金字塔特定行包含的整数。

输出格式

单独的一行,包含那个可能得到的最大的和。

样例 #1

样例输入 #1

```
5
7
3 8
8 1 0
2 7 4 4
4 5 2 6 5
```

样例输出 #1

```
30
```

提示

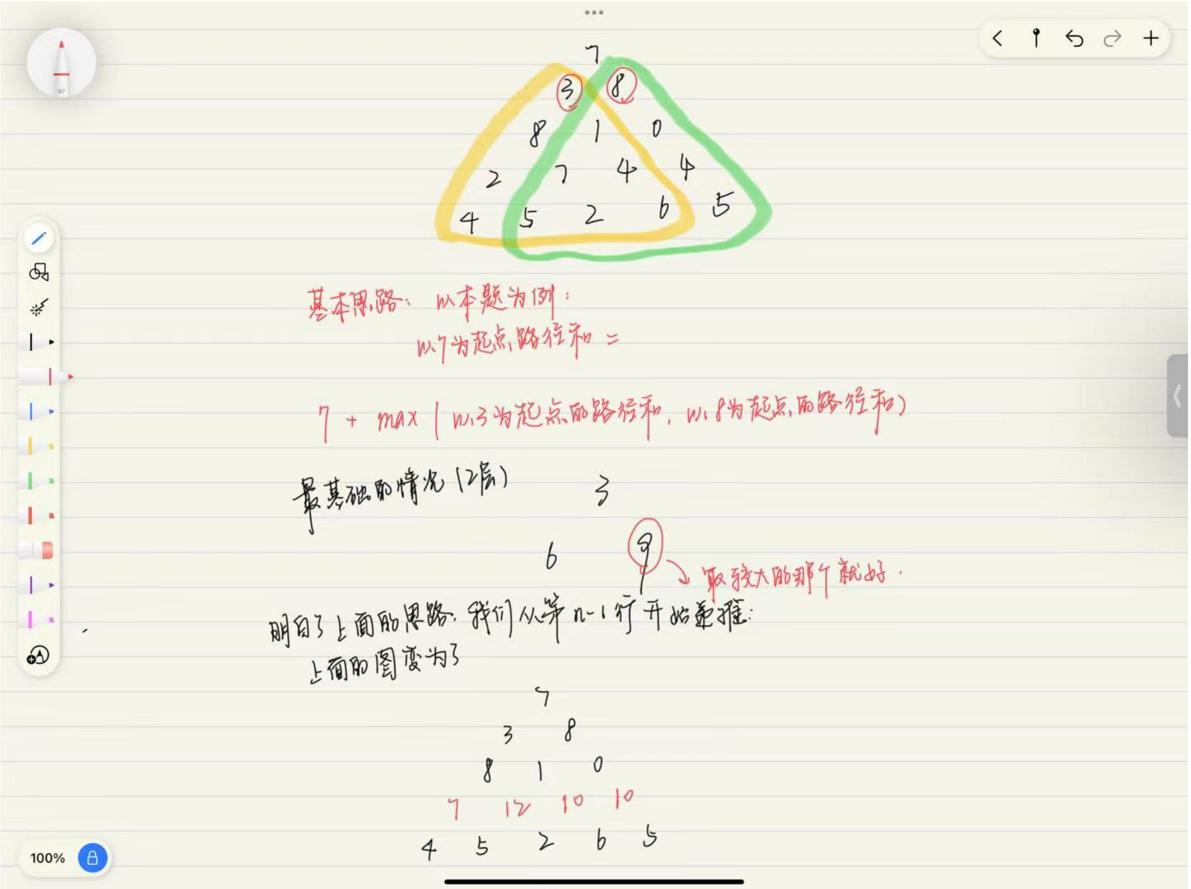
【数据范围】
对于 100% 的数据，$1 \le r \le 1000$，所有输入在 $[0, 100]$ 范围内。

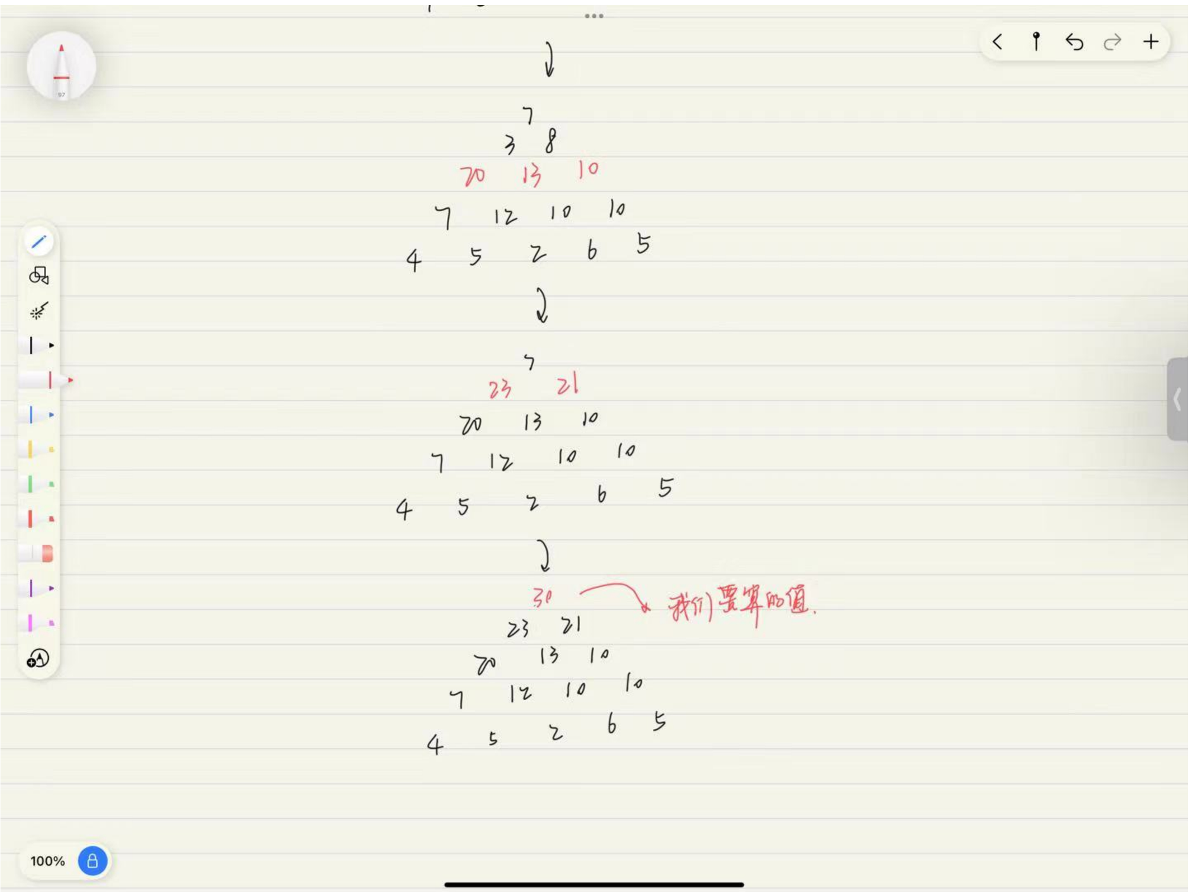
题目翻译来自NOCOW。

USACO Training Section 1.5

IOI1994 Day1T1

思路：





代码：

```c
#include <stdio.h>
int a[1005][1005], f[1005][1005];
int max(int a,int b);

int main()
{
    int n;
    int i,j;
    scanf("%d",&n);

    for(i=1;i<=n;i++){
        for(j=1;j<=i;j++){
            scanf("%d",&a[i][j]);
            f[i][j] = a[i][j];
        }
    }

    for(i=n-1;i>=1;i--){
        for(j=1;j<=i;j++){
            f[i][j] = max(f[i+1][j],f[i+1][j+1]) + a[i][j];
        }
    }

    printf("%d",f[1][1]);
    return 0;
}
int max(int a, int b){
    if(a > b)
        return a;
    return b;
}
```

介绍：动态规划算法:

动态规划的解题思路

动态规划的核心思想就是**拆分子问题，记住过往，减少重复计算。** 并且动态规划一般都是自底向上的，因此到这里，基于**青蛙跳阶**问题，我总结了一下我做动态规划的思路：

- 穷举分析
- 确定边界
- 找出规律，确定最优子结构
- 写出状态转移方程

1. 穷举分析

- 当台阶数是1的时候，有一种跳法，f（1） =1
- 当只有2级台阶时，有两种跳法，第一种是直接跳两级，第二种是先跳一级，然后再跳一级。即f(2) = 2;
- 当台阶是3级时，想跳到第3级台阶，要么是先跳到第2级，然后再跳1级台阶上去，要么是先跳到第 1级，然后一次迈 2 级台阶上去。所以f(3) = f(2) + f(1) =3
- 当台阶是4级时，想跳到第3级台阶，要么是先跳到第3级，然后再跳1级台阶上去，要么是先跳到第 2级，然后一次迈 2 级台阶上去。所以f(4) = f(3) + f(2) =5
- 当台阶是5级时......

自底向上

台阶数	1	2	3	4	5	6	...	10
子结构	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)	...	f(10)
跳法	1	2	f(1)+f(2)	f(2)+f(3)	f(3)+f(4)	f(4)+f(5)	...	f(8)+f(9)
替换变量	a=1	b=2	a=1+2=3	b=2+3=5	a=3+5=8	b=5+8	...	b=a+b

2. 确定边界

通过穷举分析，我们发现，当台阶数是1的时候或者2的时候，可以明确知道青蛙跳法。f（1） =1，f(2) = 2，当台阶n>=3时，已经呈现出规律f(3) = f(2) + f(1) =3，因此f（1） =1，f(2) = 2就是青蛙跳阶的边界。

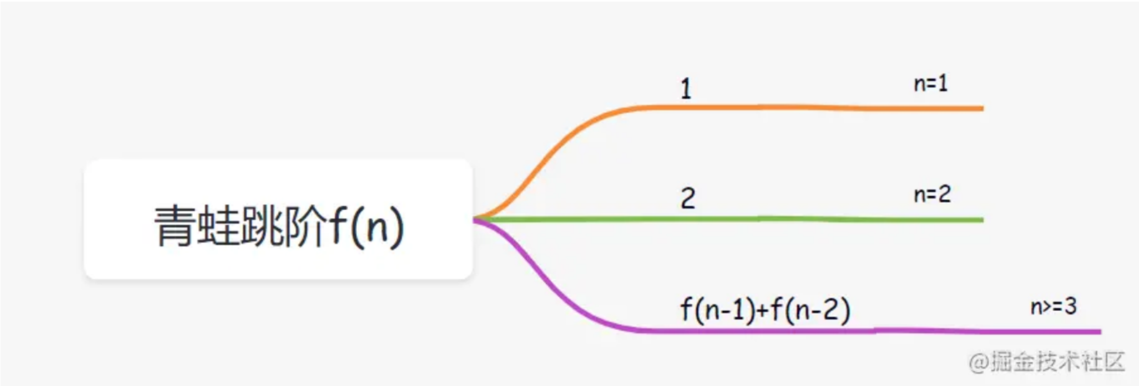
3. 找规律，确定最优子结构

n>=3时，已经呈现出规律 f(n) = f(n-1) + f(n-2)，因此，f(n-1)和f(n-2) 称为 f(n) 的最优子结构。什么是最优子结构？有这么一个解释：

> 一道动态规划问题，其实就是一个递推问题。假设当前决策结果是f(n),则最优子结构就是要让 f(n-k) 最优,最优子结构性质就是能让转移到n的状态是最优的,并且与后面的决策没有关系,即让后面的决策安心地使用前面的局部最优解的一种性质

4， 写出状态转移方程

通过前面3步，穷举分析，确定边界，最优子结构，我们就可以得出状态转移方程啦：



5. 代码实现

我们实现代码的时候，一般注意从底往上遍历哈，然后关注下边界情况，空间复杂度，也就差不多啦。动态规划有个框架的，大家实现的时候，可以考虑适当参考一下：

代码框架：

```
dp[0][0][...] = 边界值
for(状态1 ：所有状态1的值){
    for(状态2 ：所有状态2的值){
        for(...){
          //状态转移方程
          dp[状态1][状态2][...] = 求最值
        }
    }
}

```

ps：也可以算状态机？可以了解一下有限状态机~

参考链接：[(238条消息) 动态规划详解Meiko丶的博客-CSDN博客动态规划]