剑指 Offer 60. n个骰子的点数

Sunday, June 19, 2022 11:31 AM

https://leetcode.cn/problems/nge-tou-zi-de-dian-shu-lcof/
中等

把**n个骰子**扔在地上，所有骰子**朝上一面的点数之和**为s。
输入n，**打印**出s的所有可能的值出现的**概率**。
你需要**用一个浮点数数组返回答案**，其中第 i 个元素代表这 n 个骰子所能掷出的点数集合中第 i 小的那个的概率。

示例 1:

输入: 1
输出: [0.16667,0.16667,0.16667,0.16667,0.16667,0.16667]

我的思路：(昨天做了，dp数组含义想错了，于是,, 算了，放假了 >_ <)

~~dp[i][j]:第i个筛子和为j的概率。~~
~~dp[i][j]~~
~~初始化,dp[0][j]=0 dp[i][0]=0~~
~~dp[1][j]=1/6~~
~~dp[i][j]=sum(dp[i-1][j-1]+dp[i-1][j-2]+dp[i-1][j-3]+dp[i-1][j-4]+dp[i-1][j-5]+dp[i-1][j-6]);~~

今天做出来了，虽然借鉴了一下别人的思路，但依旧是自己的写法。accepted!yeah!
dp[i][j]含义：当前**i个骰子**数**之和j**的 **总骰子组合数**
递推：当前数字可由少摇一个骰子的点数累加得到，依旧是跳阶梯思路，6种跳法
dp[i][j]=dp[i-1][j-1]+dp[i-1][j-2]+dp[i-1][j-3]+...+dp[i-1][j-6]
初始化：
因为数组定义出来默认全为0；故只初始化一个骰子的结果
dp[1][1]..dp[1][6]均初始化为1。
最后将结果除以n个骰子的总组合数即可。

code：

```
class Solution {
    public double[] dicesProbability(int n) {
        // dp[i][j]:当前i个骰子数之和j的 总骰子组合数
        // 行长为n+1个，列长6*n +1，ij与现实对应，第一行为0个骰子
        int[][] dp=new int[n+1][6*n+1];
        // res:结果数组
        // 数组大小推算：1个骰子，结果1-6 6个；2个骰子，2-12 11个；3个骰子，3-18，16个
        // 故： 5*n + 1
        double[] res=new double[5*n+1];
        // 初始化dp数组，默认全0，一个骰子：dp[1][0<j<7] :1
        for(int j=1;j<7;j++){
            dp[1][j]=1;
```

```
        }
        // 递推：dp[i][j]=dp[i-1][j-1]+dp[i-1][j-2]+...+dp[i-1][j-6]
        // dp[i][j]为 i个骰子点数之和j 的可能出现次数。
        for(int i=2;i<=n;i++){ // 一个骰子初始化过了，从两个开始
            for(int j=i;j<i*6+1;j++){ // i个骰子的点数之和 j  不可能小于i，也不可能大于i*
6+1
                for(int k=1;k<7;k++){ // 递推简写 dp[i][j]+=dp[i-1][j-k]
                    if(j-k>=0){ // 避免 j很小时 j-k越界。如：dp[2][2]=dp[1][1]+...+dp[1]
[-3]
                        dp[i][j] += dp[i-1][j-k];
                    }
                }
            }
        }
        double all=Math.pow(6.0,n); // n个骰子所有的组合数
        for(int m=0;m<5*n+1;m++){
            res[m]=dp[n][n+m]/all; // 取n个骰子的每一个可能数的可能性存到res
        }
        return res;
    }
}
```