剑指60 n个骰子的点数

【二维动态规划】

设n个骰子的解(概率列表)为f(n), 其中某个点数和x的概率值为f(n,x)

递推公式:

$$f(n,x)=\sum_{i=1}^{6} f(n-1,x-i)\times\frac{1}{6}$$

n个骰子和为x的概率是: n-1个骰子和为x-1,x-2,x-3,x-4,x-5,x-6的概率分别相加乘1/6

```cpp
// 我的解法:  使用二维数组。
class Solution {
public:
    vector<double> dicesProbability(int n) {
        // 可能值: n,n+1....,6n
        // dp[n][x] 表示n个骰子, 掷到和为x的概率
        vector<vector<double>> dp(n + 1, vector<double>(6 * n + 1, 0));
        for (int i = 1; i <= 6; i++) {
            dp[1][i] = (double)1/6;
        }
        for (int i = 2; i <= n; i++) {
            for (int x = i; x <= 6*i; x++) {
                for (int up = 1; up <= 6 && x > up; up++) {
                    dp[i][x] += dp[i-1][x-up];
                }
                dp[i][x] = dp[i][x]/6;
            }
        }

        vector<double> outcome(dp[n].begin()+n, dp[n].end());
        return outcome;
    }
};
```

```cpp
// TODO:仔细看看！！妙啊~
// 因为dp[n]只和dp[n-1]有关, 所以可以优化为2个一维数组交替前进。
// k神题解:

class Solution {
public:
    vector<double> dicesProbability(int n) {
        vector<double> dp(6, 1.0 / 6.0);
        for (int i = 2; i <= n; i++) {
            vector<double> tmp(5 * i + 1, 0);
            for (int j = 0; j < dp.size(); j++) {
                for (int k = 0; k < 6; k++) {
```

```
                tmp[j + k] += dp[j] / 6.0;
            }
        }
        dp = tmp;
    }
    return dp;
  }
};

```