```
工厂   >>>>>>>>>>>>>>>>>    仓库   >>>>>>>>>>>>>>>>    消费者
           0.4              12              0.6
```

```
    private int randomMove(int x, double PROB_MOVEUP) {
        // by PROB_MOVEUP probability, x increases, otherwise x decreases.
        // If x is smaller than 0, reset x to 0
        final int STEP_SIZE = 1;
        return Math.random() < PROB_MOVEUP ? x+STEP_SIZE: Math.max(0,x-STEP_SIZE);
    }

    @Test
    public void testRandomMove() {
        final double PROB_MOVEUP = 0.4;
        final int RETRY = 40;
        final int START_VALUE = 0;
        int value = START_VALUE;
        for (int i = 0 ;i < RETRY; i++) {
            value = randomMove(value, PROB_MOVEUP);
            System.out.print(value + ", ");
        }
        // What's the average expectation of value after infinite number of RETRY?
    }
```

An example sequence

```
0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 2, 1, 0, 0, 0, 0,
1, 0, 0, 0, 1, 2, 1, 0, 1, 2, 3, 4, 4
```

```
0 ================
1 ===========
2 ========
3 ===
...
```

已知二阶递推公式，以及边界条件

$p_{n,k+1} = Pp_{n-1,k} + (1-P)p_{n+1,k}$

$p_{0,k+1} = Pp_{0,k} + (1-P)p_{1,k}$

其中P 是已知的增加的概率，是常量。假设能够稳定状态存在，那么

$p_n = Pp_{n-1} + (1-P)p_{n+1}$

构造以下一阶递推公式

$p_n = Kp_{n-1}$

其中K 是常数。那么

$p_n = Pp_n/K + (1-P)Kp_n$

得到

$K = P/(1-P)$

进一步验证边界条件也符合，所以得到表达式，是指数函数

$p_n = (P/(1-P))^n p_0$

因为总体概率为1，所以

$\sum_{n=0}^{\infty} p_n = \sum_{n=0}^{\infty} K^n p_0 = 1$

因此得到

$p_0 = (1-2P)/(1-P)$

我们想求平均有多少库存

$E = \sum_{n=0}^{\infty} np_n = \sum_{n=0}^{\infty} nK^n p_0$

用错项相消的方式求解

$E/K = \sum_{n=0}^{\infty} nK^{n-1} p_0 = \sum_{n=1}^{\infty} nK^{n-1} p_0 = \sum_{n=0}^{\infty}(n+1)K^n p_0$

$E/K - E = \sum_{n=0}^{\infty} K^n p_0 = 1$

$E = K/(1-K) = P/(1-2P)$

可见，$P < 0.5$

比方说，P是0.49的时候，E是24.50。P是0.45的时候，E是4.50