如何判断一个点在一条有向边的左边还是右边?这个利用几何上向量积(叉积)的求解公式即可。如果有向边 1->2 叉乘有向边 1->3 的结果为正,说明 2 在有向边 1->3 的左边,比如图 9-4;如果有向边 1->2 叉乘有向边 1->3 的结果为负,说明 2 在有向边 1->3 的右边,比如图 9-5。

具体过程请参看如下代码中的 crossProduct 方法,该方法描述了向量(x1,y1)叉乘向量(x2,y2),两个向量的开始点都是原点。

```
public double crossProduct(double x1, double y1, double x2, double y2) {
    return x1 * y2 - x2 * y1;
}
```

至此,我们已经解释了解法二的所有细节,全部过程请参看如下代码中的 isInside2 方法。

```
public boolean isInside2(double x1, double y1, double x2, double y2,
              double x3, double y3, double x, double y) {
       // 如果三角形的点不是逆时针输入, 改变一下顺序
       if (crossProduct(x3 - x1, y3 - y1, x2 - x1, y2 - y1) >= 0) {
              double tmpx = x2;
              double tmpy = v2;
              x2 = x3;
              y2 = y3;
              x3 = tmpx;
              y3 = tmpy;
       if (crossProduct(x2 - x1, y2 - y1, x - x1, y - y1) < 0) {
              return false;
       if (crossProduct(x3 - x2, y3 - y2, x - x2, y - y2) < 0) {
              return false:
       if (crossProduct(x1 - x3, y1 - y3, x - x3, y - y3) < 0) {
              return false;
       return true;
```

# 折纸问题

### 【题目】

请把一段纸条竖着放在桌子上,然后从纸条的下边向上方对折1次,压出折痕后展开。此时折痕是凹下去的,即折痕突起的方向指向纸条的背面。如果从纸条的下边向上方连续

对折 2 次,压出折痕后展开,此时有三条折痕,从上到下依次是下折痕、下折痕和上折痕。 给定一个输入参数 N,代表纸条都从下边向上方连续对折 N 次,请从上到下打印所有折痕 的方向。

```
例如: N=1 时,打印:
down
N=2 时,打印:
down
down
up
```

#### 【难度】

尉★★☆☆

## 【解答】

如上图关系可以总结出:

- 产生第 i+1 次折痕的过程,就是在对折 i 次产生的每一条折痕的左右两侧,依次插入上折痕和下折痕的过程。
- 所有折痕的结构是一棵满二叉树,在这棵满二叉树中,头节点为下折痕,每一棵 在子树的头节点为上折痕,每一棵右子树的头节点为下折痕。
- 从上到下打印所有折痕方向的过程,就是二叉树的先右、再中、最后左的中序遍历。

具体过程请参看如下代码中的 printAllFolds 方法。

```
public void printAllFolds(int N) {
         printProcess(1, N, true);
}

public void printProcess(int i, int N, boolean down) {
         if (i > N) {
```

```
return;
}
printProcess(i + 1, N, true);
System.out.println(down ? "down " : "up ");
printProcess(i + 1, N, false);
}
```

纸条连续对折 n 次之后一定产生  $2^{n-1}$  条折痕,所以要打印所有的节点,不管用什么方法,其时间复杂度肯定都是  $O(2^n)$ ,因为解的空间本身就有这么大,但是本书提供的方法的额外空间复杂度为 O(n),也就是这棵满二叉树的高度,额外空间主要用来维持递归函数的运行,也就是函数栈的大小。

# 蓄水池算法

#### 【题目】

有一个机器按自然数序列的方式吐出球(1号球,2号球,3号球,……),你有一个袋子,袋子最多只能装下K个球,并且除袋子以外,你没有更多的空间。设计一种选择方式,使得当机器吐出第N号球的时候(N>K),你袋子中的球数是K个,同时可以保证从1号球到N号球中的每一个,被选进袋子的概率都是K/N。举一个更具体的例子,有一个只能装下 10个球的袋子,当吐出 100个球时,袋子里有 10个球,并且  $1\sim100$ 号中的每一个球被选中的概率都是 10/100。然后继续吐球,当吐出 1000个球时,袋子里有 10个球,,当吐出 1个球时,袋子里有 10个球,并且  $1\sim1000$ 号中的每一个球被选中的概率都是 10/1000。继续吐球,当吐出 1个球时,袋子里有 10个球,并且  $1\sim1000$ 号中的每一个球被选中的概率都是 10/1000。继续吐球,当吐出 10分球时,袋子里有 100个球,并且  $1\sim1000$ 0号中的每一个球被选中的概率都是 10/10000。继续吐球,当吐出 10/10000,即吐球的同时,已经吐出的球被选中的概率也动态地变化。

## 【难度】

尉★★☆☆

### 【解答】

这道题的核心解法就是蓄水池算法,我们先说这个算法的过程,然后再证明。

- 1. 处理 1~k 号球时,直接放进袋子里。
- 2. 处理第 i 号球时(i > k),以 k/i 的概率决定是否将第 i 号球放进袋子。如果不决定将第 i 号球放进袋子,直接扔掉第 i 号球。如果决定将第 i 号球放进袋子,那么就从袋子里的 k