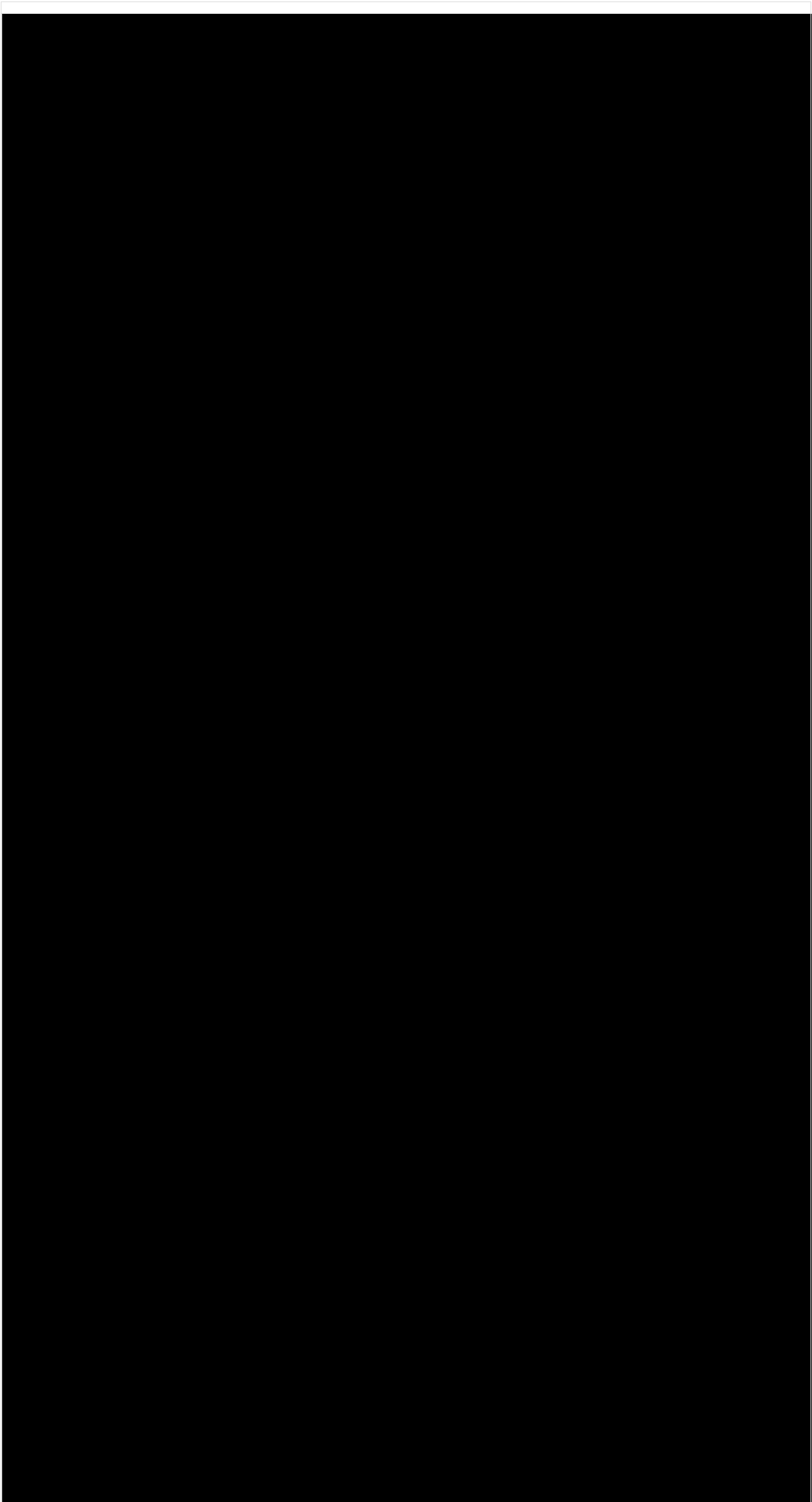


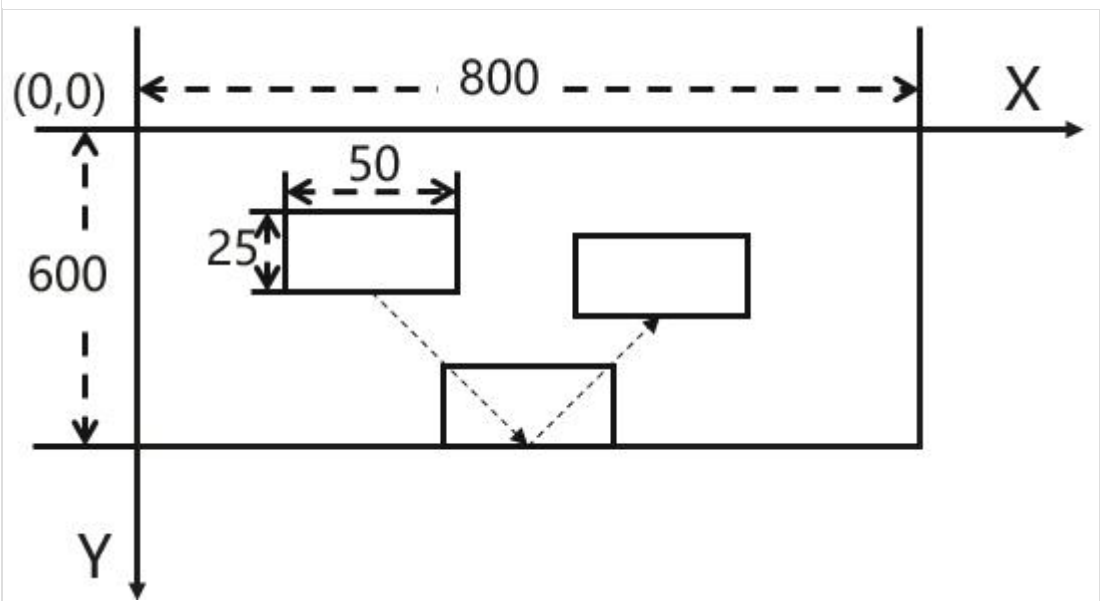
题目描述：

DVD 机在视频输出时，为了保护电视显像管，在待机状态会显示“屏保动画”，如下图所示，
DVD Logo 在屏幕内来回运动，碰到边缘会反弹：



请根据如下要求，实现屏保 *Logo* 坐标的计算算法。

- 1、屏幕是一个 800×600 像素的矩形，规定屏幕的左上角点坐标原点，沿横边向右方向为 X 轴，沿竖边向下方向为 Y 轴；
- 2、*Logo* 是一个 50×25 像素的矩形，初始状态下，左上角点坐标记做 (x, y) ，它在 X 和 Y 方向上均以 1 像素/秒的速度开始运动；
- 3、遇到屏幕四个边缘后，会发生镜面反弹，即以 45° 碰撞边缘，再改变方向以 45° 弹出；
- 4、当 *Logo* 和四个角碰撞时，两个边缘同时反弹的效果是 *Logo* 会原路返回。



请编码实现， t 秒后 *Logo* 左上角点的坐标。

输入描述：

输入 3 个数字，以空格分隔：

$x\ y\ t$

第一个数字表示 *Logo* 左上角点的初始 X 坐标；

第二个数字表示 *Logo* 左上角点的初始 Y 坐标；

第三个数字表示时间 t ，题目要求即求 t 秒后 *Logo* 左上角点的位置。

输出描述：

输出 2 个数字，以空格分隔：

$x\ y$

第一个数字表示 t 秒后，*Logo* 左上角点的 X 坐标

第二个数字表示 t 秒后，*Logo* 左上角点的 Y 坐标

补充说明：

所有用例均保证：

- 1、输入的 x 和 y 坐标会保证整个 *Logo* 都在屏幕范围内，*Logo* 不会出画；
- 2、所有输入数据都是合法的数值，且不会出现负数；
- 3、 t 的最大值为 100000。

示例 1

输入：

0 0 10

输出：

10 10

说明：

输入样例表示 *Logo* 初始位置在屏幕的左上角点，10s 后，*Logo* 在 X 和 Y 方向都移动了 10 像素，因此输出 10 10。

示例 2

输入：

500 570 10

输出：

510 570

说明：

输入样例表示初始状态下，*Logo* 的下边缘再有 5 像素就碰到屏幕下边缘了，5s 后，会与屏幕碰撞，碰撞后，斜向 45° 弹出，又经过 5s 后，*Logo* 与起始位置相比，水平移动了 10 像素，垂直方向回到了原来的高度。

// 本题为考试单行多行输入输出规范示例，无需提交，不计分。

```
import java.util.Scanner;
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        Scanner in = new Scanner(System.in);
```

```
        int x = 0;
```

```
        int y = 0;
```

```
        int t = 0;
```

```
        while (in.hasNextInt()) { // 注意，如果输入是多个测试用例，请通过 while 循环处理多个测试用例
```

```
            x = in.nextInt();
```

```
            y = in.nextInt();
```

```
            t = in.nextInt();
```

```
        }
```

```
        int step = 0;
```

```
        boolean large = true;
```

```
        while (step < t) {
```

```
            if (x + 50 == 800) {
```

```
                large = false;
```

```
            }
```

```
            if (x == 0) {
```

```
                large = true;
```

```
            }
```

```
            if (large) {
```

```
                x++;
```

```
            } else {
```

```
                x--;
```

```
            }
```

```
            step++;
```

```
        }
```

```
step = 0;
large = true;
while (step < t) {
    if (y + 25 == 600) {
        large = false;
    }
    if (y == 0) {
        large = true;
    }
    if (large) {
        y++;
    } else {
        y--;
    }
    step++;
}
//x += t;
// y += t;
System.out.println(x + " " + y);
}
}
```