题目描述：
停车场有一横排车位，0 代表没有停车，1 代表有车。至少停了一辆车在车位上，也至少有一个空位没有停车。

为了防剐蹭，需为停车人找到一个车位，使得距停车人的车最近的车辆的距离是最大的，返回此时的最大距离。
输入描述：
1、一个用半角逗号分割的停车标识字符串，停车标识为 0 或 1，0 为空位，1 为已停车。

2、停车位最多 100 个。
输出描述：
输出一个整数记录最大距离。
补充说明：
示例 1
输入：
1,0,0,0,0,1,0,0,1,0,1
输出：
2
说明：
当车停在第 3 个位置上时，离其最近的的车距离为 2（1 到 3）。
当车停在第 4 个位置上时，离其最近的的车距离为 2（4 到 6）。
其他位置距离为 1。
因此最大距离为 2。

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String[] parkingLot = scanner.nextLine().split(",");
        int count = 0;
        int countMax = 0;

        // 有三种情况
        // 如果前边都是 0：0,0,0,0,1...类型
        // 中间处理：1,0,...,0,1 类型
        // 如果后边都是 0：...,1,0,0,0,0 类型

        boolean flag = false;
        for (String item : parkingLot) {
            // 如果前边都是 0：0,0,0,0,1...类型
            if (item.equals("0") && !flag) {
                count += 2;
            }
            // 中间处理：1,0,...,0,1 类型
```

```
                else if (item.equals("1")) {
                    flag = true;        // 结束‘前边都是 0’
                    int temp1 = (count + 1) / 2;
                    if (temp1 > countMax) {
                        countMax = temp1;
                    }
                    count = 0;
                } else if (item.equals("0") && flag) {
                    count++;
                }

            }
            // 如果后边都是 0：...,1,0,0,0,0 类型
            int temp = count;
            if (temp > countMax) {
                countMax = temp;
            }
            System.out.println(countMax);
        }
    }
```