找车位题目描述：

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;

/**
 *
 * @since 2023/07/09 09:21
 * @author Myo
 */
public class Main {

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

```
        Scanner sc = new Scanner(System.in);
        while (sc.hasNext()) {
            String strs = sc.nextLine();
            if (strs == null || "".equals(strs)) {
                System.out.println();
                continue;
            }

            String[] nums = strs.split(",");
            int[] num = new int[nums.length];
            for (int i = 0; i < nums.length; i++) {
                num[i] = Integer.parseInt(nums[i]);
            }

            int start = 0;
            while (start < num.length && num[start] == 0) {
                start ++ ;
            }

            int end = num.length - 1;
            while (end >= 0 && num[end] == 0) {
                end -- ;
            }

            int maxLen = 0;
            int idx = start;

            while (idx < end) {
                int tmp = 0;
                while (idx < end && num[idx] == 0) {
                    tmp++;
                    idx++;
                }

                maxLen = Math.max(tmp, maxLen);
                idx++;
            }

            int res = (maxLen + 1) / 2;
            res = Math.max(res, Math.max(start, (num.length - 1 - end)));
            System.out.println(res);
        }
    }
}
```