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。

```
#include <bits/stdc++.h>
using namespace std;
const int N = 1e3 + 7;
char s[N];
int a[N], len = 0, n = 0, ans = 0, p = 0, l, r;
int main() {
    scanf("%s", s + 1); len = strlen(s + 1);
    n = (len + 1) / 2;
    for(int i = 1;i <= n; i++) a[i] = s[2 * (i - 1) + 1] - '0';
    for(int i = 1;i <= n; i++) if(a[i]) break; else ans = i;
    for(int i = n;i >= 1; i--) if(a[i]) break; else ans = max(ans, n - i + 1);
```

```
    a[0] = 1; a[n + 1] = 1;
    for(int i = 1;i <= n; i++) {
        l = 0, r = 0;
        if(a[i]) continue;
        for(int j = i - 1;j >= 0; j--) if(a[j]) {l = j; break;}
        for(int j = i + 1;j <= n + 1; j++) if(a[j]) {r = j; break;}
        l = i - l; r = r - i;
        ans = max(ans, min(l, r));
    }
    cout << ans;
}
// 64 位输出请用 printf("%lld")
```