题目描述：

给定一个字符串，只包含字母和数字，按要求找出字符串中的最长（连续）子串的长度，字符串本身是其最长的子串，子串要求：

1、 只包含 1 个字母(a~z, A~Z)，其余必须是数字；

2、 字母可以在子串中的任意位置；

如果找不到满足要求的子串，如全是字母或全是数字，则返回-1。

输入描述：
字符串(只包含字母和数字)
输出描述：
子串的长度
补充说明：
收起
示例 1
输入：
abC124ACb
输出：
4
说明：
满足条件的最长子串是 C124 或者 124A，长度都是 4
示例 2
输入：
a5
输出：
2
说明：
字符串自身就是满足条件的子串，长度为 2
示例 3
输入：
aBB9
输出：
2
说明：
满足条件的子串为 B9，长度为 2
示例 4
输入：
abcdef
输出：

-1
说明：
没有满足要求的子串，返回-1

```
#include <cctype>
#include <iostream>
using namespace std;
int alpha = 0;
int num = 0;

bool check(){
    if (alpha == 1 && num > 0) return true;
    return false;
}

bool check2(){
    if(alpha < 2) return true;
    return false;
}
int main() {
    string s;
    cin >> s;
    long long n = s.size();
    if(n < 2) {
        cout << -1 << endl;
        return 0;
    }
    long long l = 0, r = 0;
    long long res = -1;
    if(isalpha(s[0])) ++alpha;
    else ++num;
    while(check2() && r < n) {
        if(check()) {
            res = max(r-l+1, res);
        }
        ++r;
        if(isalpha(s[r])) ++alpha;
        else ++num;
    }
    for(long long i=1; i<n; ++i){
        ++l;
        if(isalpha(s[i-1])){
            alpha--;
        } else {
            num--;
```

```
            }

            while(check2() && r < n) {
                if(check()) {
                    res = max(r-l+1, res);
                }
                ++r;
                if(isalpha(s[r])) ++alpha;
                else ++num;
            }

        }
        cout << res << endl;
        return 0;
}
// 64  位输出请用  printf("%lld")
```