3. Longest Substring Without Repeating Characters (Medium)

Given a string `s`, find the *length of the longest substring* without duplicate characters.

A substring is a contiguous sequence of characters within a string.

Example 1:

```
Input: s = "zxyzxyz"

Output: 3
```

Explanation: The string "xyz" is the longest without duplicate characters.

Example 2:

```
Input: s = "xxxx"

Output: 1
```

Constraints:

- `0 <= s.length <= 1000`
- `s` may consist of printable ASCII characters.

▼ 思路:

作法一：hashmap 儲存記錄過字母比對

T:$O(n)$, S:$O(n)$

作法一

```
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
```

```
        int ans=0,tmp=0,tmplast=0;
        unordered_map<char,int> check;
        for(int i=0;i<s.size();i++){
            if(check.find(s[i]) == check.end()){
                tmp += 1;
            }
            else{
                if(tmplast<=check[s[i]]){
                    ans = (ans > tmp) ? ans : tmp;
                    tmplast = check[s[i]]+1;
                    tmp = i-tmplast+1;
                }
                else tmp+=1;
            }
            check[s[i]] = i;
        }
        return (tmp>ans)?tmp:ans;
    }
};
```

作法一精簡

```
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
        int ans=0,tmplast=0,r=0;
        unordered_map<char,int> check;
        while(r<s.size()){
            if(check.find(s[r]) != check.end()){
                tmplast = (tmplast>check[s[r]]+1)?tmplast:check[s[r]]+1;
            }
            check[s[r]] = r;
            ans = (ans>r-tmplast+1)?ans:r-tmplast+1;
            r++;
        }
        return ans;
```

```
    }
};
```