003. Longest Substring Without Repeating Characters

003 Longest Substring Without Repeating Characters

- **Array+Hash Table+String**

Description

Given a string, find the length of the **longest substring** without repeating characters.

Examples:

Given `"abcabcbb"`, the answer is `"abc"`, which the length is 3.

Given `"bbbbb"`, the answer is `"b"`, with the length of 1.

Given `"pwwkew"`, the answer is `"wke"`, with the length of 3. Note that the answer must be a **substring**, `"pwke"` is a *subsequence* and not a substring.

1. Thought Line

(1) Bit manipulation for duplicate elements.

2. Hash Table + String

```
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
        int res = 0;
        set<char> setHash;
        int loopStart = 0;

        for (int i=0; !s.empty()&&i<=s.size()-1; ++i){
            char chr = s[i];
            if (setHash.find(chr) == setHash.end())
                setHash.insert(chr);
            else{
                // calculate the none duplicate number in this term.
                res = (setHash.size()>res)?setHash.size():res;

                // find out the index of which one leaded to this
                // duplication issue.
                for (int duplicateSpt = loopStart; duplicateSpt<=i-1;
                    ++duplicateSpt)
                {
                    setHash.erase(s[duplicateSpt]);
                    if (s[duplicateSpt]==chr){
                        i = i-1;
                        loopStart = duplicateSpt+1;
                        break;
                    }
                }
            }

        }
        res = (s.size()-loopStart>res)?s.size()-loopStart:res;
        return res;
    }
};
```