do by me

简单思路描述

遍历每个字符，对于每个字符扩展不重复字符串，每次扩展的时候需要判断扩展的字符是否在已经有的字符集合中，如果有的话就停止扩展。找到每个s[j]为左子母的无重复字符串的长度，然后每次迭代的过程中，把最大的保存下来。

代码

```cpp
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
         int result = 0;  // 最终结果
        int temp = 0;    // 临时长度

        for(int i = 0; i < s.length(); i++) {
            std::unordered_set<char> charSet;  // 用来检查重复字符
            temp = 0;  // 重置临时长度

            for(int j = i; j < s.length(); j++) {
                // 如果当前字符已经在集合中，说明遇到重复字符
                if(charSet.find(s[j]) != charSet.end()) {
                    break;
                }
                // 未重复，加入集合并增加长度
                charSet.insert(s[j]);
                temp++;
            }

            // 更新最大长度
            if(temp > result) {
                result = temp;
            }
        }

        return result;
    }
```

```
};```

## 题解-滑动窗口+哈希表（升级版）

````C++
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
        unordered_map<char, int> dic;
        int i = -1, res = 0, len = s.size();
        for(int j = 0; j < len; j++) {
            if (dic.find(s[j]) != dic.end())
                i = max(i, dic.find(s[j])->second); // 更新左指针
            dic[s[j]] = j; // 哈希表记录
            res = max(res, j - i); // 更新结果
        }
        return res;
    }
};
```

官方题解

49. 字母异位词分组 - 力扣（LeetCode）

知识补充

字符串的排序：**ranges::sort**

哈希表的创建：unordered_map <string , vector> hashtable;

遍历哈希表的关键字：　　for(const auto& pair :hashtable){

　　result.push_back(pair.second);

　}

```
1
```
```