# 题目

罗马数字包含以下七种字符: I，V，X，L，C，D 和M。

字符 数值

I 1

V 5

X 10

L 50

C 100

D 500

M 1000

例如，罗马数字2写做 II ，即为两个并列的1。12写做XII，即为X + II 。27写做XXVII, 即为XX + V + II 。

通常情况下，罗马数字中小的数字在大的数字的右边。但也存在特例，例如4不写做 IIII，而是 IV。数字1在数字5的左边，所表示的数等于大数5减小数1得到的数值4。同样地，数字9表示为 IX。这个特殊的规则只适用于以下六种情况：

I可以放在V (5)和X (10)的左边，来表示4和9。

X可以放在L (50)和C (100)的左边，来表示40和 90。

C可以放在D (500)和M (1000)的左边，来表示 400和 900。

给定一个罗马数字，将其转换成整数。输入确保在1到3999的范围内。

**示例 1:**

输入: "III"

输出: 3

**示例 2:**

输入: "IV"

输出: 4

**示例 3:**

输入: "IX"

输出: 9

**示例 4:**

输入: "LVIII"

输出: 58

解释: L = 50, V= 5, III = 3.

**示例 5:**

输入: "MCMXCIV"

输出: 1994

解释: M = 1000, CM = 900, XC = 90, IV = 4.

# 分析

## 方法一：暴力破解

## 方法二：哈希表

**思路：**

转换类问题，换一种理解就是查找对应关系，即哈希表映射问题。

**代码：**

class Solution {

public:

int romanToInt(string s) {

int result=0;

map<char,int> mp={

{'I',1},

{'V',5},

{'X',10},

{'L',50},

{'C',100},

{'D', 500},

{'M', 1000}

};//初始化哈希表

for(int i=0;i<s.length();i++)

{

if(mp[s[i]] < mp[s[i+1]])

result -= mp[s[i]];

else

{

result += mp[s[i]];

}

}

return result;

}

};

**复杂度分析：**

时间复杂度：O(N)，遍历了一遍数组。

空间复杂度：O(1)，使用了int。

或：

class Solution {

public:

int romanToInt(string s) {

int result = 0;

unordered\_map<char,int> mp;

mp['I']=1;

mp['V']=5;

mp['X']=10;

mp['L']=50;

mp['C']=100;

mp['D']=500;

mp['M']=1000;

for(int i=0;i<s.length();i++)

{

if(mp[s[i]] < mp[s[i+1]])

result -= mp[s[i]];

else

{

result += mp[s[i]];

}

}

return result;

}

};