# 题目

URL化。编写一种方法，将字符串中的空格全部替换为%20。假定该字符串尾部有足够的空间存放新增字符，并且知道字符串的“真实”长度。（注：用Java实现的话，请使用字符数组实现，以便直接在数组上操作。）

示例 1：

输入："Mr John Smith ", 13

输出："Mr%20John%20Smith"

示例 2：

输入：" ", 5

输出："%20%20%20%20%20"

提示：

字符串长度在 [0, 500000] 范围内。

# 分析

## 方法一：模拟

思路：

代码：

class Solution {

public:

std::string replaceSpaces(std::string S, int length) {

// 创建一个新的string对象来存放替换后的结果

std::string result;

// 遍历字符串中的前length个字符

for (int i = 0; i < length; ++i) {

if (S[i] == ' ') {

// 如果遇到空格，则追加%20

result += "%20";

} else {

// 否则追加原字符

result += S[i];

}

}

// 返回替换后的结果

return result;

}

};

## 方法二：原地修改

思路：

代码：

class Solution {

public:

std::string& replaceSpaces(std::string& S, int length) {

int spacesCount = 0;

// 计算空格数量

for (int i = 0; i < length; ++i) {

if (S[i] == ' ') {

++spacesCount;

}

}

// 计算替换后字符串的新长度

int newLength = length + spacesCount \* 2;

// 确保字符串有足够的空间

S.resize(newLength);

// 从后向前替换空格

for (int i = length - 1, j = newLength - 1; i >= 0; --i) {

if (S[i] == ' ') {

S[j--] = '0';

S[j--] = '2';

S[j--] = '%';

} else {

S[j--] = S[i];

}

}

return S;

}

};