P1098 [NOIP2007 提高组] 字符串的展开

题目描述

在初赛普及组的“阅读程序写结果”的问题中，我们曾给出一个字符串展开的例子：如果在输入的字符串中，含有类似于“d-h”或者“4-8”的字串，我们就把它当作一种简写，输出时，用连续递增的字母或数字串替代其中的减号，即，将上面两个子串分别输出为“defgh”和“45678"。在本题中，我们通过增加一些参数的设置，使字符串的展开更为灵活。具体约定如下：

(1) 遇到下面的情况需要做字符串的展开：在输入的字符串中，出现了减号“-”，减号两侧同为小写字母或同为数字，且按照ASCII码的顺序，减号右边的字符严格大于左边的字符。

(2) 参数p_1：展开方式。p1=1时，对于字母子串，填充小写字母；p_1=2时，对于字母子串，填充大写字母。这两种情况下数字子串的填充方式相同。p_1=3时，不论是字母子串还是数字字串，都用与要填充的字母个数相同的星号“*”来填充。

(3) 参数p_2：填充字符的重复个数。p_2=k表示同一个字符要连续填充k个。例如，当p_2=3时，子串“d-h”应扩展为“deeefffgggh”。减号两边的字符不变。

(4) 参数p_3：是否改为逆序：p_3=1表示维持原来顺序，p_3=2表示采用逆序输出，注意这时候仍然不包括减号两端的字符。例如当p_1=1、p_2=2、p3=2时，子串“d-h”应扩展为“dggffeeh”。

(5) 如果减号右边的字符恰好是左边字符的后继，只删除中间的减号，例如：“d-e”应输出为“de”，“3-4”应输出为“34”。如果减号右边的字符按照ASCII码的顺序小于或等于左边字符，输出时，要保留中间的减号，例如：“d-d”应输出为“d-d”，“3-1”应输出为“3-1”。

输入格式

共两行。

第1行为用空格隔开的3个正整数，依次表示参数p1,p2,p3。

第2行为一行字符串，仅由数字、小写字母和减号“-”组成。行首和行末均无空格。

输出格式

共一行，为展开后的字符串。

输入样例

```
1 2 1
abcs-w1234-9s-4zz
```

输出样例

abcsttuuvvw1234556677889s-4zz

解析

模拟题，逐一条件进行翻译即可。

编码

```
#include<bits/stdc++.h>

using namespace std;
int p1, p2, p3 = 0;
char ch[300];

//修改字符
void Modify(int l) {
    //开始的字符
    char c = l;
    //填充大写
    if (p1 == 2) {
        if (c >= 'a') {
            c -= 32;
        }
    }
    //所有都用*代替
    if (p1 == 3) {
        c = '*';
    }
    for (int m = 0; m < p2; ++m) {
        printf("%c", c);
    }
}


int main() {
    //输入;
    scanf("%d%d%d%s", &p1, &p2, &p3, ch);
    //字符的索引
    int i = 0;
    //当ch[i]有值时;
    while (ch[i]) {
        char be = ch[i - 1];
        char af = ch[i + 1];
        //f存储ch[i],便于判断;
        char f = ch[i];
        //意思是ch[i]若为'-',就判断其前后是否满足条件，满足进入循环;
        if (f == '-' && af > be && (be >= '0' && af <= '9'
        || be >= 'a' && af <= 'z')) {
            //正序输出
```

```
            if (p3 == 1) {
                    //两be和af两个字母紧挨着的时候就不会执行循环;
                    for (int l = be + 1; l < af; l = l++) {
                        Modify(l);
                    }
                } else {
                    //两be和af两个字母紧挨着的时候就不会执行循环;
                    for (int l = af - 1; l > be; l = l--) {
                        Modify(l);
                    }
                }
            } else {
                printf("%c", ch[i]);
            }
            //继续判断下一个字符
           i++;
        }
        return 0;
}
```

逻辑航线培优教育，信息学奥赛培训专家。

扫码添加作者获取更多内容。