**算法实习报告：实验二 字符串比较问题**

题目： 字符串比较问题（动态规划算法）

1. **需求分析**

对于长度相同的2个字符串A和B，其距离定义为相应位置字符距离之和。2个非空格字符的距离是它们的ASCII码之差的绝对值。空格与空格的距离为0，空格与其他字符的距离为一定值k。

在一般情况下，字符串A和B的长度不一定相同。字符串A的扩展是在A中插入若干空格字符所产生的字符串。在字符串A和B的所有长度相同的扩展中，有一对距离最小的扩展，该距离称为字符串A和B的扩展距离。

算法要求如下

1、 数据输入：第1行是字符串A，第2行是字符串B，第3行是空格与其他字符的距离定值k。

2、 输出：字符串A和B的扩展距离。

例如

输入：

cmc

snmn

2

输出：10

1. **概要设计**

由题意知，因为是在支付中任意的插入空格进行扩展，那么对于A中的一个字符a，可以与B中的任意一个字符或者空格进行大小比较。那么a与B中的字符的最短距离就是a与B中所有的字符比较大小以及与空格比较大小后最终的到的最小值。

于是，我们可以设计一个双层循环结构，对两个字符串中的所有字符进行依次遍历，查找它与另一个字符串的最小距离。将值储存在val(i,j)中，表示（‘ ’+A）表第i个字符与（‘ ’+B）表第j个字符的最小距离。

设字符串A和B的子串A[1..i]和B[1..j]的扩展距离为val(i,j)，

依题意，字符串A和B有三种可能的情况：

1）A串最后一个字符是空格，B串最后一个字符是字母，则val(i,j)=val(i-1,j)+k；

2）A串最后一个字符时字母，B串最后一个字符时空格，则val(i,j)=val(i,j-1)+k；

3）A串和B串最后一个字符均是字母，则val(i,j)=val(i-1,j-1)+dist(ai,bi)；

则val(i,j)具有最优子结构性质，递归定义为：val(i,j)=min{val(i-1,j)+k, val(i,j-1)+k, val(i-1,j-1)+dist(ai,bj)}

（使用动态规划算法，自底向上的计算各个子问题并利用每次计算的结果，避免重复运算，从而降低算法复杂度。）

1. **详细设计**

#include "iostream"

#include "string"

using namespace *std*;

#define MAXNUM 65535

int val[100][100]; //存放最小值

int k; //空格与其他字符的距离值

*string* str1, str2;

//计算两个字符之间的ASCII距离

int dist(char a, char b) {

return *abs*(a - b);

}

//求最小距离

int mindist() {

int len1 = str1.*length*();

int len2 = str2.*length*();

int temp;

val[0][0] = 0; //表示空格

for (int i = 0; i < len1; i++) { //下标从1开始，便于与上次结果进行操作

for (int j = 0; j < len2; j++) {

if (i + j > 0) { //至少有一个不是空格,应该空格相互比较对于结果没有变换，操作是多余的

val[i][j] = MAXNUM;

//情况1

if (i > 0) {

temp = val[i - 1][j] + k;

if (temp < val[i][j]) {

val[i][j] = temp;

}

}

//情况2

if (j > 0) {

temp = val[i][j - 1] + k;

if (temp < val[i][j]) {

val[i][j] = temp;

}

}

//情况3

if (i \* j > 0) {

temp = val[i - 1][j - 1] + dist(str1[i], str2[j]);

if (temp < val[i][j]) {

val[i][j] = temp;

}

}

}

}

}

return val[len1-1][len2-1];

}

int main() {

*cin* >> str1 >> str2;

*cin* >> k;

str1 = " " + str1; //在字符的比较中加入与空格的比较

str2 = " " + str2;

*cout* << mindist() << *endl*;

return 0;

}

1. **调试分析**

从动态规划递归式可知，算法的时间复杂度为O（mn），m和n分别是字符串A和B的长度。

1. **用户手册**

1、 数据输入：第1行是字符串A，第2行是字符串B，第3行是空格与其他字符的距离定值k。

2、 输出：字符串A和B的扩展距离。

1. **测试结果**

测试了几组数据终端输出结果如下：

1. 第一组

输入：

cmc

snmn

2

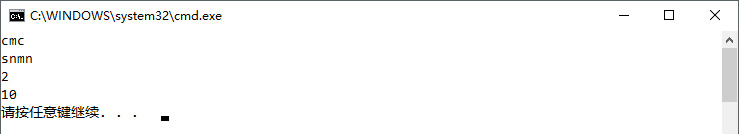
输出：

最优情况如下：

\_c\_m\_c

s\_nmn\_ （用下划线”\_”表示插入的空格）

所以，结果为2+2+2+0+2+2=10。



1. 第二组

输入：

anc

mm

20

输出：

最优情况如下：

anc

\_mm （用下划线”\_”表示插入的空格）

所以，结果为20+1+10=31。

