题目：凯撒密码

Contents

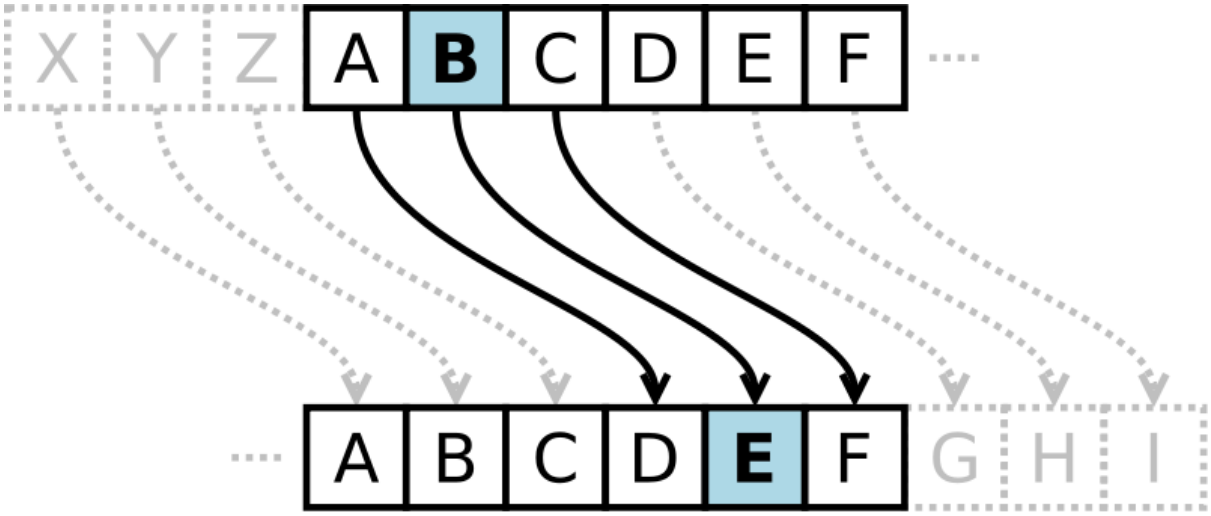
SE@SJTU 2023

1 简介与背景

小兰发给了你一个奇怪的文本文件。文件内容仅仅包含小写字母、空格和换行。小兰说，这个文件里面的信息是使用凯撒密码加密过的，需要破解之后才能知道信息的具体内容。小兰介绍说：凯撒密码是一种简单的密码算法，又称移位密码。它的原理是将明文中的每个字母都按照一个固定的偏移量进行替换，形成密文。小兰在文件中隐藏了很多秘密，需要你写一段程序把这些秘密找出来。为了进一步了解凯撒密码，你查找了维基百科，找到了下面这段资料：

> 凯撒密码（英语：Caesar cipher），或称凯撒加密、凯撒变换、变换加密，是一种最简单且最广为人知的加密技术。凯撒密码是一种替换加密技术，明文中的所有字母都在字母表上向后（或向前）按照一个固定数目进行偏移后被替换成密文。例如，当偏移量是 3 的时候，所有的字母 A 将被替换成 D，B 变成 E，以此类推。这个加密方法是以罗马共和时期凯撒的名字命名的，据称当年凯撒曾用此方法与其将军们进行联系。
>
> 凯撒密码的替换方法是通过排列明文和密文字母表，密文字母表示通过将明文字母表向左或向右移动一个固定数目的位置。例如，当偏移量是左移 3 的时候（解密时的密钥就是 3）：
>
> 明文字母表：ABCDEFGHIJKLMNOPQRSTUVWXYZ
>
> 密文字母表：DEFGHIJKLMNOPQRSTUVWXYZABC
>
> 使用时，加密者查找明文字母表中需要加密的消息中的每一个字母所在位置，并且写下密文字母表中对应的字母。需要解密的人则根据事先已知的密钥反过来操作，得到原来的明文。例如：
>
> 明文：THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
>
> 密文：WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ
>
> 凯撒密码的加密、解密方法还能够通过同余的数学方法进行计算。首先将字母用数字代替，A=0，B=1，…，Z=25。此时偏移量为 n 的加密方法即为：

> $E_n(x) = (x + n) mod 26$
>
> 解密就是：
>
> $D_n(x) = (x - n) mod 26$



为了获知小兰隐藏的信息，你需要写一段程序完成以下指定的任务。

2 题目要求

2.1 [步骤 1] 读取字典和加密信息

函数名：本步骤需要实现为一个名为 `LoadData` 的函数。

你需要从标准输入中读取两行文本。第一行文本为“字典文件”的路径（路径不含空格）；第二行文本为“加密信息文件”的路径（路径不含空格）；

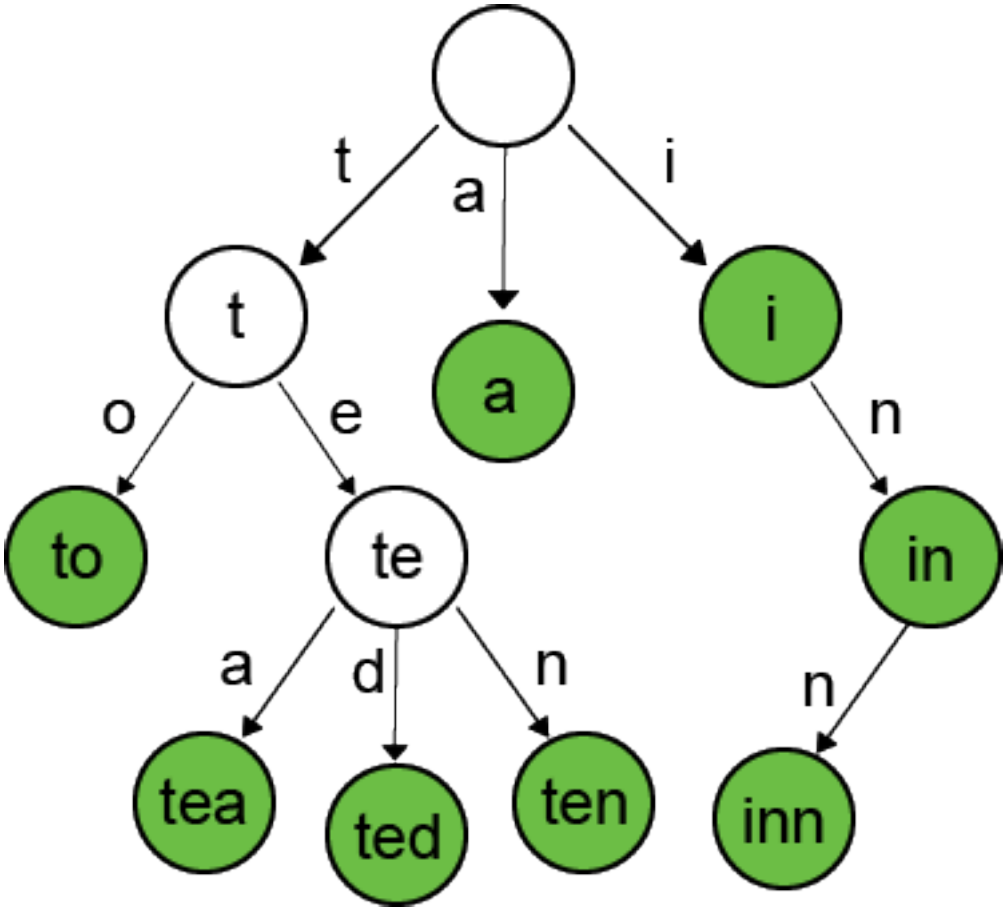
字典文件为一个多行文本文件，其中每一行保存了一个小写英文单词。加密信息文件同样为一个多行文本文件，每一行保存了一个加密后的小写英文单词，表示小兰发过来的加密信息。

标准输出：

在本步骤中，你需要的按上述方式读取数据，并在标准输出中打印一行信息，包括两个数字：第一个数字为字典文件中，以字母 c 开头的单词的数量。第二个数字为加密信息文件中单词的数量。两个数字以一个空格隔开。

2.2 [步骤 2] 构建字典树

函数名：本步骤需要实现为一个名为 `BuildTrie` 的函数。



为了让后续步骤运行的更快，你需要使用字典文件中单词构建一棵字典树。

字典树的节点可以参考下面的结构：

```
struct TrieNode {
  map<char, TrieNode *> children;
  bool isWord;
};
```

标准输出：

在本步骤中，你需要在标准输出中打印一行信息，包括一个数字：为字典树中叶子节点的个数。

注意：

1. 叶子节点个数不一定等于单词个数。
2. 本步骤目的为加速查找，跳过本步骤不影响后续步骤的正确性，但是无法得到本步骤的分数，同时可能无法在指定时间内破解信息。

2.3 [步骤 3] 单调密码序列

函数名：本步骤需要实现为一个名为 `CrackCodeInc` 的函数。

小兰偷偷告诉你说，由于凯撒密码单一的偏移量非常容易破解，她传给你的加密信息中，每个单词都有一个自己的偏移量，这些偏移量组成了一个序列，她称之为密码序列。密码序列中的第 x 个数字，为第

x 个密文单词加密时使用的偏移量。因而对第 x 个密文单词进行反向偏移后得到的单词应包含在字典文件中。小兰所使用的所有单词一定包含在此前的字典文件中。

你知道小兰很喜欢数学。因此她一定会选择一个优美的序列作为密码序列。这个优美的序列是一个由 **非负整数** 组成的 **单调递增** 序列。同时，该序列中最大偏移量（也就是最后一个数）最小。

根据上面这些信息，你打算尝试计算出密码序列。

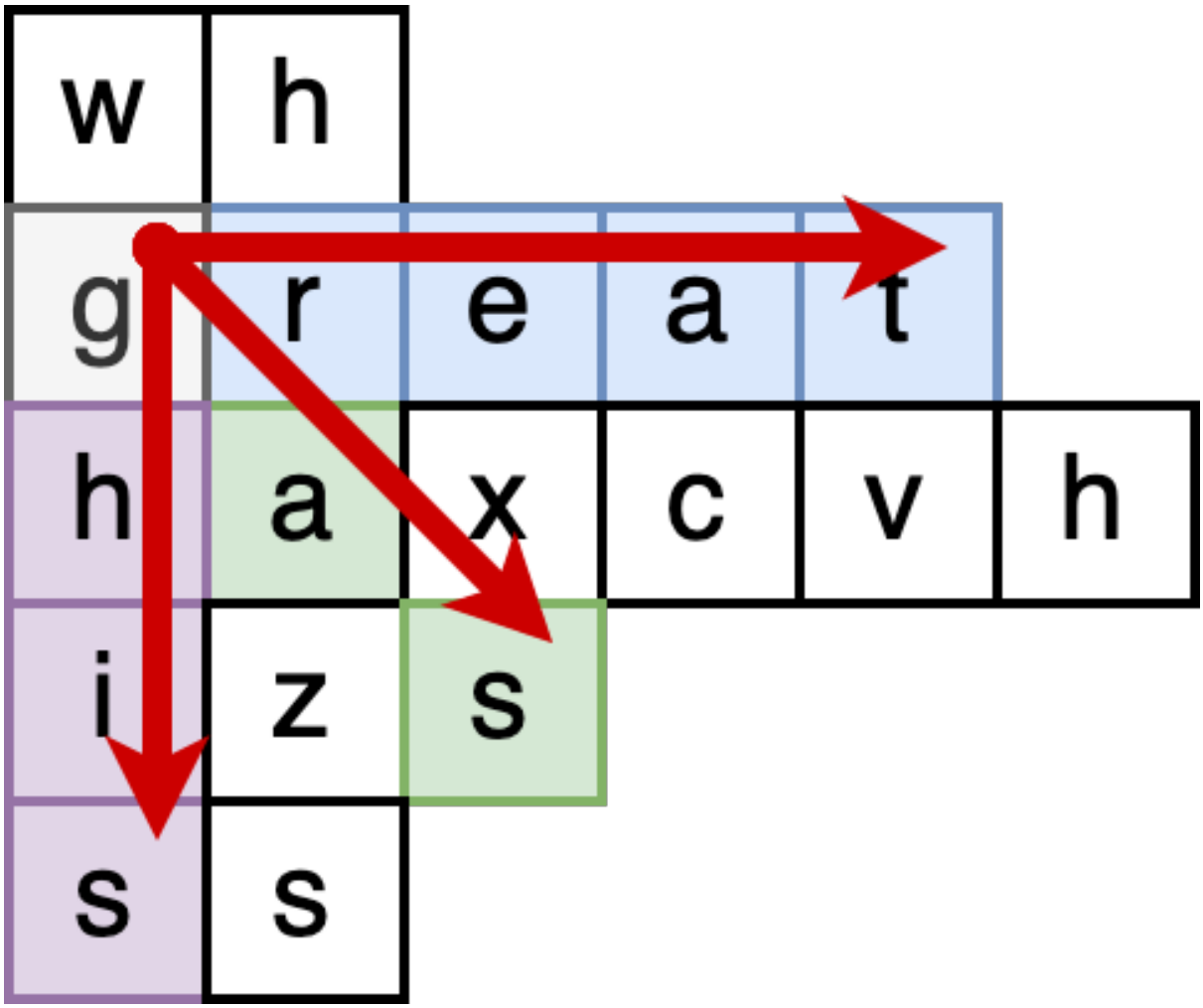
标准输出：

在本步骤中，你需要在标准输出中打印一行信息，包括两个数字：第一个数字为密码序列的第一个数字。第二个数字为密码序列的最后一个数字。两个数字以一个空格隔开。

注意：密码序列为正整数，说明一个密文单词，在 **反向** 偏移后得到的单词在单词文件中。

2.4 [步骤 4] 密文文中文

函数名：本步骤需要实现为一个名为 `CodeInCode` 的函数。



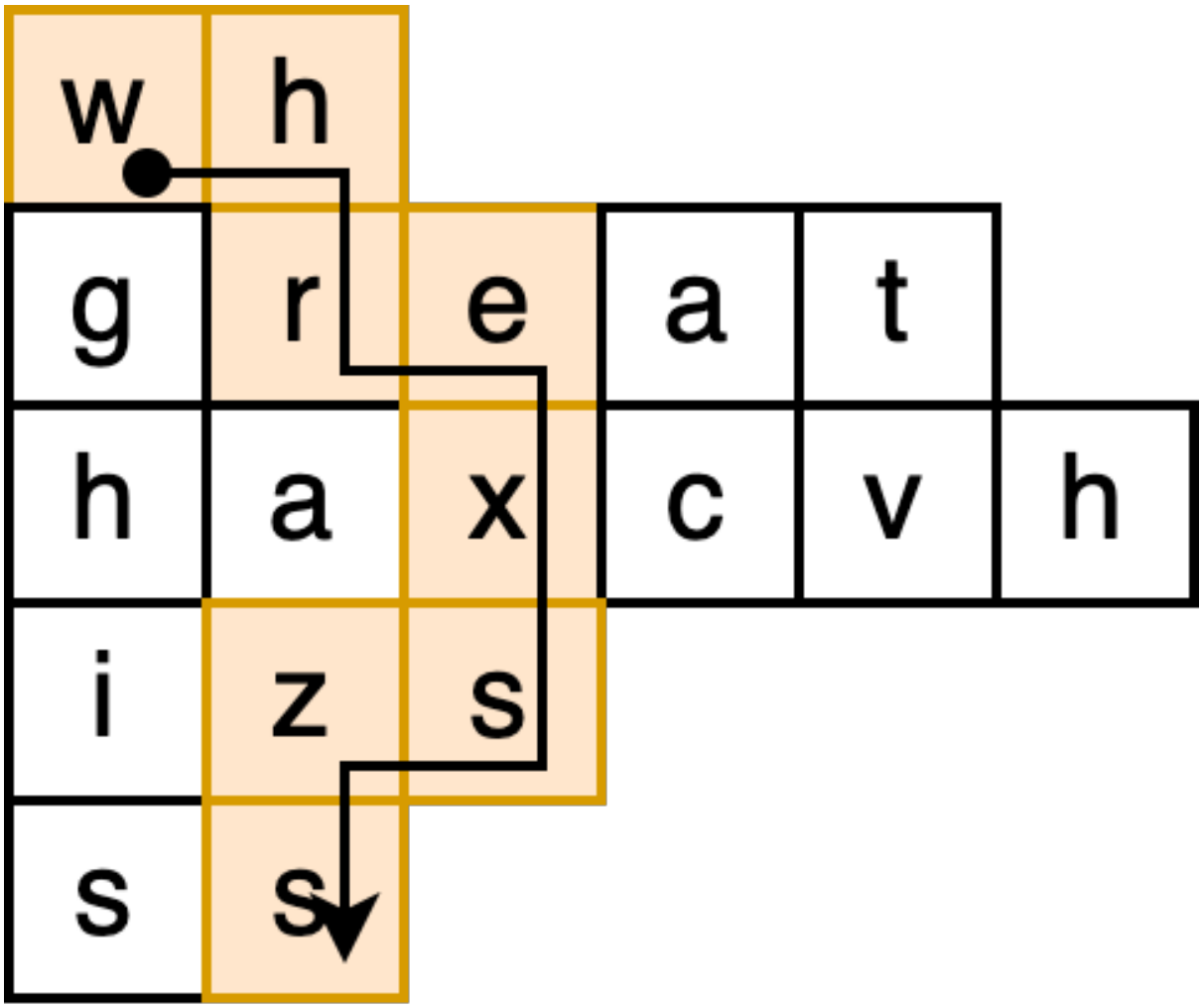
小兰说这个加密文件还远不止这么简单，密文中还保存了另外一个秘密单词。如果将所有密文单词按顺序排列，每行一个单词，可以组成一个字母阵列。在这个阵列中，隐藏着小兰的秘密单词。该单词是整个阵列中，通过往“正右方”、“正下方”、“右下方对角线”三个方向可以组成的最长单词（指字典中的未加密的单词）。如果有多个最长，字典序最小的那个为秘密单词。在本步骤中，你需要找出最长单词。

标准输出：

在本步骤中，你需要在标准输出中打印一行信息，包括两个数字和一个字符串：两个数字分别为秘密单词的首字母的行号、列号（行列均从 0 开始），字符串为该单词。三部分各以一个空格隔开。

2.5 [步骤 5] 密文最短路径

函数名：本步骤需要实现为一个名为 `PathInCode` 的函数。



在上述字母阵列中，从阵列左上角（第一个单词的第一个字母开始，即第 0 行第 0 列）开始，通过“往下”、“往右”、“往左”三个方向每次移动一个字母，可以得到无数条通往阵列右下角（最后一个单词的最后一个字母）的路径。

1. 每一步可以向左、向右、向下。
2. 走一步的代价为“旧位置上字母”到“新位置上字母”变换的偏移量（代价范围从 1 到 26）。

例如，从 A 走到 C，代价为 2；从 Z 走到 C 代价为 3；从 X 走到 X 的代价为 26；从 C 走到 A 的代价为 24。

小兰想知道所有路径中，代价之和最短的路径是哪一条。

请输出最短路径，并打印该路径到文件。对于总代价相同的路径，请打印字典序最小的路径（此处字典序为 l、r、d 三个字母组成的路径字符串的字典）。

标准输出：

在本步骤中，你需要在标准输出中打印一行信息，包括两个数字：两个数字分别为路径长度和总代价，以一个空格隔开。

文件输出：

你需要将路径的内容输出到名为“path.txt”的文件中，

路径的格式为一串由“lrd”三个字母组成的字符串，表示从阵列左上角开始，依次通过若干个：

- 往左移动一步（l）
- 往右移动一步（r）
- 往下移动一步（d）

最终能到达右下角，且该路径符合上述要求。

为了阅读方便，路径字符串拆分为多行进行输出：每输出 20 个字符进行一次换行。

3 输入输出样例

为了方便大家进行调试，我们在 data 目录中给出了样例（sample）、小规模（small）和大规模（large）数据。

给出的每个测试包括四个文本文件，以 sample 为例：

- sample.stdin.txt 为标准输入的内容（仅作参考，程序的标准输入为 `words_alpha.txt` 文件和 `sample.txt` 文件的路径）
- sample.txt 为输入文件的内容（对应标准输入中的 `./data/sample.txt`）
- sample.stdout.txt 为正确的标准输出的内容
- sample.path.txt 为正确的 path.txt 文件内容

大规模测试的 stdout.txt 和 path.txt 文件不予给出。

4 提交要求和考核标准

4.1 编码要求

语言不限，但根据题目选择合适的语言并按照要求进行设计和编码。

4.2 评分标准

设计和实现程序，完成前述功能。如果不能完成全部程序功能，也请不要担心，我们会根据各个方面独立评分。具体标准如下：

步骤	分值
[步骤 1] 读取字典和加密信息	15
[步骤 2] 构建字典树	15
[步骤 3] 单调密码序列	20
[步骤 4] 密文文中文	15
[步骤 5] 密文最短路径	15
通过所有测试数据（每个测试数据运行限时 5 秒）	10
代码规范、命名标准、代码风格、注释和说明等	10

4.3 提交要求

请将程序源代码和需要提交的文件使用 7z 格式压缩后命名为“姓名.7z”并执行以下两个操作：

- 将其 **放置在 U 盘中**，请确保文件完整性。；

压缩包中应仅包含源代码和指定提交的文件，不包含测试用的输入输出、中间文件或可执行文件。压缩包大小原则上不超过 5MB。