Leonardo é um nômade digital e viaja pelo mundo programando em diferentes cafés das cidades por onde passa. Recentemente, resolveu criar um blog, para compartilhar suas experiências e aprendizados com seus amigos.

Para criação do blog, ele optou por utilizar uma ferramenta pronta, que há um limite de caracteres que se pode escrever por dia, e Leonardo está preocupado que essa limitação, afinal, irá impedir de contar suas melhores experiências. Para contornar esse problema, decidiu usar um sistema de abreviação de palavras em seus posts.

O sistema de abreviações é simples e funciona da seguinte forma: para cada letra, é possível escolher uma palavra que inicia com tal letra e que aparece no post. Uma vez escolhida a palavra, sempre que ela aparecer no post, ela será substituída por sua letra inicial e um ponto, diminuindo assim o número de caracteres impressos na tela.

Por exemplo, na frase: “hoje eu programei em Python”, podemos escolher a palavra “programei” para representar a letra ‘p', e a frase ficará assim: “hoje eu p. em Python”, economizando assim sete caracteres. Uma mesma palavra pode aparecer mais de uma vez no texto, e será abreviada todas as vezes. Note que, se após uma abreviação o número de caracteres não diminuir, ela não deve ser usada, tal como no caso da palavra “eu” acima.

Leonardo precisa que seu post tenha o menor número de caracteres possíveis, e por isso pediu a sua ajuda. Para cada letra, escolha uma palavra, de modo que ao serem aplicadas todas as abreviações, o texto contenha o menor número de caracteres possíveis.

**Entrada**

Haverá diversos casos de teste. Cada caso de teste é composto de uma linha, contendo uma frase de até 10⁴ caracteres. A frase é composta de palavras e espaços em branco, e cada palavra é composta de letras minúsculas ('a'-'z'), e contém entre 1 e 30 caracteres cada.

O último caso de teste é indicado quando a linha dada conter apenas um “.”, o qual não deverá ser processado.

**Saída**

Para cada caso de teste, imprima uma linha contendo a frase já com as abreviações escolhidas e aplicadas.

Em seguida, imprima um inteiro N, indicando o número de palavras em que foram escolhidas uma letra para a abreviação no texto. Nas próximas N linhas, imprima o seguinte padrão “C. = P”, onde C é a letra inicial e P é a palavra escolhida para tal letra. As linhas devem ser impressas em ordem crescente da letra inicial.

| Exemplo de Entrada | Exemplo de Saída |
| --- | --- |
| abcdef abc abc abc . | a. abc abc abc 1 a. = abcdef |

***Solução***

import java.io.IOException;

import java.util.stream.Collectors;

import java.util.\*;

public class Main {

public static void main(String[] args) throws IOException {

List<String> alfabeto = new ArrayList(Arrays.asList("abcdefghijklmnopqrstuvwxyz".split("")));

List<String> entradas = new ArrayList<String>();

Scanner sc = new Scanner(System.in,"ISO-8859-1");

while(true) {

String frase = sc.nextLine()

.trim()

.toLowerCase()

.replaceAll("\n", " ")

.replaceAll("\t", " ");

if(frase.equals(".")) break;

if(frase.length() <= 0) continue;

List<String> vetPalavrasFrase = new ArrayList(Arrays.asList(frase.split(" ")));

Map<String, String> dicionario = new HashMap<String, String>();

alfabeto.stream().forEach(letra -> {

dicionario.put(letra, "");

});

Map<String, Map<String, Integer>> repeticaoPalavra = new HashMap<String, Map<String, Integer>>();

alfabeto.stream().forEach(letra -> {

repeticaoPalavra.put(letra, new HashMap<String, Integer>());

});

vetPalavrasFrase.stream().forEach(palavra -> {

repeticaoPalavra.get(palavra.substring(0, 1)).put(palavra, 0);

});

vetPalavrasFrase.stream().forEach(palavra -> {

int qt = repeticaoPalavra.get(palavra.substring(0, 1)).get(palavra);

qt++;

repeticaoPalavra.get(palavra.substring(0, 1)).put(palavra, qt);

});

alfabeto.stream().forEach(letra -> {

Map<String, Integer> map = repeticaoPalavra.get(letra);

List<String> chaves = new ArrayList<String>(map.keySet());

int qtTotalCharPorLetra = 0;

for (String chave : chaves) {

qtTotalCharPorLetra += map.get(chave) \* chave.length();

}

int qtTotalCharMelhorCenario;

int qtTotalCharMelhorCenarioAux = 10000;

for (String chave : chaves) {

qtTotalCharMelhorCenario = (qtTotalCharPorLetra - (map.get(chave) \* chave.length())) + (map.get(chave) \* 2);

if((qtTotalCharMelhorCenarioAux > qtTotalCharMelhorCenario) && chave.length() > 2) {

qtTotalCharMelhorCenarioAux = qtTotalCharMelhorCenario;

dicionario.put(letra, chave);

}

}

});

String fraseAbreviada = vetPalavrasFrase

.stream()

.map(palavra -> {

String primeiraLetra = palavra.substring(0, 1);

return dicionario.get(primeiraLetra).equals(palavra) ? primeiraLetra + "." : palavra;

}).collect(Collectors.joining(" "));;

System.out.println(fraseAbreviada);

int quantidadeAbreviacoes = 0;

for(String letra : alfabeto) {

if(!dicionario.get(letra).equals("")) {

quantidadeAbreviacoes++;

}

}

System.out.println(quantidadeAbreviacoes);

alfabeto.stream().forEach(letra -> {

if(!dicionario.get(letra).equals("")) {

System.out.println(letra + ". = " + dicionario.get(letra) );

}

});

}

}

}