|  |  |  |  |
| --- | --- | --- | --- |
| 题目名称 | dictionary | install | paint |
| 输入文件 | dictionary.in | install.in | paint.in |
| 输出文件 | dictionary.out | install.out | paint.out |
| 每个测试点时限 | 1s | 1s | 1s |
| 测试点数目 | 10 | 20 | 10 |
| 每个测试点分值 | 10 | 5 | 10 |
| 运行内存上限 | 128M | 64M | 256M |

dictionary

**Description**

输入n个字符串，作为字典中的单词，然后再输入m个字符串，分别查询与它们编辑距离为1的单词个数，如果没有，则输出-1。

在这里，字符串a与b的编辑距离是指:可以通过以下操作，将a变为b或b变为a，最少操作次数即为编辑距离距离。

1. 删除a串中某个位置的字母;
2. 添加一个字母到a串中某个位置;
3. 替换a串中某一位置的一个字母为另一个字母;

**Input**

第一行包含两个正整数N和M。

接下来的N行，每行一个字符串，表示一个词典中的单词。

再接下来M行，每行一个字符串，表示一个待查字符串。

**Output**

输出应包括M行，表示每个询问的答案。

**Sample Input**

4 3

abcd

abcde

aabc

abced

abcd

abc

abcdd

**Sample Output**

-1

2

3

**Hint**

对于50%的数据，0 <= N,M <= 1000

对于100%的数据，0 <= N,M <= 10000，所有字符串长度均在1至20之间，且均由小写字母构成。所有词典中的单词互不相同，但是查询字符串可能有重复。

install

**Description**

现在我们的手头有N个软件，对于一个软件i，它要占用Wi的磁盘空间，它的价值为Vi。我们希望从中选择一些软件安装到一台磁盘容量为M计算机上，使得这些软件的价值和尽可能大（即Vi的和最大）。

但是现在有个问题：软件之间存在依赖关系，即软件i只有在安装了软件j（包括软件j的直接或间接依赖）的情况下才能正确工作。幸运的是，一个软件最多依赖另外一个软件(可能有环)。如果一个软件不能正常工作，那么它能够发挥的作用为0。

我们现在知道了软件之间的依赖关系：软件i依赖软件Di。现在请你设计出一种方案，安装价值尽量大的软件。一个软件最多只能被安装一次，如果一个软件没有依赖则Di=0，这时只要这个软件安装了，它就能正常工作。

**Input**

第一行包含两个整数N, M

第二,三,四行均包含n个整数,分别代表W,V,D

**Output**

一个整数,表示最大价值。

**Sample Input**

3 10

5 5 6

2 3 4

0 1 1

**Sample Output**

5

**Hint**

对于10%的数据, N<=18

对于另外20%的数据, 保证没有环

对于100%的数据, 0<=N<=100, 0<=M<=500, 0<=Wi<=M, 0<=Vi<=1000, 0<=Di<=N, Di≠i

paint

**Description**

有 N 条木板需要被粉刷。 每条木板被分为 M 个格子。 每个格子要被刷成红色或蓝色。你每次粉刷时，只能选择一条木板上一段连续的格子，然后涂上一种颜色。 每个格子最多只能被粉刷一次。

如果windy只能粉刷 T 次，他最多能正确粉刷多少格子？

(一个格子如果未被粉刷或者被粉刷错颜色，就算错误粉刷)

**Input**

第一行包含三个整数，N M T。

接下来有N行，每行一个长度为M的字符串，'0'表示红色，'1'表示蓝色。

**Output**

包含一个整数，最多能正确粉刷的格子数。

**Sample Input**

3 6 3

111111

000000

001100

**Sample Output**

16

**Hint**

对于30%的数据，保证 1 <= N,M <= 10 ； 0 <= T <= 100

对于100%的数据，保证 1 <= N,M <= 50 ； 0 <= T <= 2500