猜词（word）

这是一道交互题。

【题目描述】

在本题中，你需要和交互库玩一款经典的游戏。在每局游戏中，交互库会从词库中生成一个 5 个字母的单词，并告诉你它的首字母，你需要在 5 次机会内猜中它。

每次猜测都需要猜一个词库中存在的单词。如果猜对了，游戏结束；在每次猜错后，交互库会返回哪些字母的位置是正确的（以金色表示），以及哪些字母在待猜单词中出现了但位置是错误的（以银色表示）。

具体来说，交互库会返回两个布尔类型的数组 `gold` 和 `silver`。`gold[i]`（$0 \le i < 5$，下同）表示第 i 个字母是否猜对了（位置和内容均正确）；`silver[i]` 表示如果第 i 个字母没猜对（即不为金色），这个字母是否在本次猜测非金色字母的部分出现过。

例如，待猜单词为 `panic`，猜测 `paper` 后交互库会返回 `gold[0] = true`（`p` 正确），`gold[1] = true`（`a` 正确），其余均为 `false`（注意 `paper` 中的第二个 `p` 虽然在 `panic` 中出现过，但出现位置为本次猜测中的金色字母部分，因此 `silver[2] = false`）。

又如，待猜单词为 `apple`，猜测 `paper` 后交互库会返回 `gold[2] = true`（`p` 正确），`silver[0] = true`（`p` 在本次猜测非金色字母的部分出现过），`silver[1] = true`（`a` 出现过），`silver[3] = true`（`e` 出现过），其余均为 `false`。

【评分方式】

由于每局游戏具有较高的随机性，在本题中，你需要连续玩 $T = 1000$ 局游戏。每局游戏的评分标准如下：

- 如果任意一次猜测单词的长度不等于 5，或者**猜测的单词不在词库中，得 0 分**；
- 如果第 1 次猜测猜对，得 150 分；
- 如果第 2 次猜测猜对，得 120 分；
- 如果第 3 次猜测猜对，得 100 分；
- 如果第 4 次猜测猜对，得 90 分；
- 如果第 5 次猜测猜对，得 85 分；
- 如果 5 次猜测均错，得 0 分。你在本题中的得分为【1000 局游戏的平均分】和 100 分的较小值。

【如何使用交互库】

本题只支持 `C++`。

你只能提交一个源文件 `word.cpp` 实现下列函数，并且遵循下面的命名和接口。

对于使用 C++ 的选手

你需要包含头文件 word.h。

你不需要，也不应该实现主函数。

你需要实现的函数有：

```
const char *guess(int num_testcase, int remaining_guesses, char
    initial_letter, bool *gold, bool *silver);
void init(int num_scramble, const char *scramble);
```

其中，第 i 局游戏的 num_testcase 参数为 i（$1 \leq i \leq T$），每局游戏会调用 $1 \sim 5$ 次 guess 函数，第 j 次调用的 remaining_guesses 参数为 $6-j$（$1 \leq j \leq 5$）。initial_letter 参数为当前局游戏待猜单词的首字母（保证为小写字母）。保证每次调用 guess 函数的 num_testcase 单调不降；保证 num_testcase 相同时initial_letter不变且 remaining_guesses 单调递降。如果某次猜测猜对或非法，则该局游戏结束，下次调用 guess 函数为下一局游戏。

gold 和 silver 为如上所述的两个布尔数组。当 remaining_guesses 参数为 5 时，gold 和 silver 数组不可用（即，可能为空指针），请避免使用它们；当remaining_guesses 参数小于 5 时，gold 和 silver 为两个大小为 5 的布尔数组，存储着上一次猜测的结果。

guess 函数的返回值需要是一个长度为 5 的字符串，表示猜测的单词。该单词需要在词库中。

init 函数会在调用所有 guess 函数之前调用恰好一次。其中 num_scramble 参数是词库大小，scramble 是一个长度为 num_scramble * 5 的字符串，存储着词库中的所有单词，每个单词长度为 5，中间没有任何分隔符。

【附加文件】

本题下发的文件有 word.h, word_sample.cpp, play.cpp, grader.cpp, scramble.txt, scramble.csv, scramble_pure.txt。

word_sample.cpp 是你要实现的 word.cpp 的一个样例。

grader.cpp 为示例评测程序（编译命令：g++ -o grader grader.cpp word.cpp）。scramble.txt, scramble.csv, scramble_pure.txt 均为本题所使用的词库文件，其中scramble.txt 以换行符分隔单词，scramble.csv 以逗号分隔单词，scramble_pure.txt 不分隔单词（即，与 init 函数中的 scramble 参数内容相同）。

play.cpp 是一个可以让你和你的程序玩这个游戏的程序（编译命令：g++ -o play play.cpp word.cpp）。下面介绍 play.cpp 的输入与输出格式。

【样例输入格式】

输入第一行包含一个正整数 T，表示游戏局数。

接下来 T 局游戏，每局游戏第一行输入一个小写字母，表示待猜单词的首字母。

接下来 0 ～ 4 行，每行一个长度为 5 的由 `g`、`s`、`-` 组成的字符串，其中第 i 位（$0 \le i < 5$，下同）是 `g` 表示 `gold[i] = true`，第 i 位是 `s` 表示 `silver[i] = true`，第 i 位是 `-` 表示 `gold[i] = silver[i] = false`。如果猜对或猜测单词非法，则该局游戏结束，因此输入的行数是可变的。

【样例输出格式】

对于除了第一行游戏局数以外的每行输入，输出一个长度为 5 的字符串，表示猜测的单词。样例输出加入了额外的空行以便阅读。

【样例输入】

```
7
p
gg---
gg---
ggg--
a
g----
ssgs-
a
g---g
gggg-
a
g---g
g---g
g---g
g---g
a
a
c
-sss-
```

【样例输出】

```
paper
paths
panda
panic

aargh
paper
apple

afore
apply
apple

apple
apple
apple
apple
apple

abcde

apple

kraal
cobra
```

【样例解释】

对于第 1 局游戏，待猜单词为 panic，在第 4 次猜测猜对。

对于第 2 局游戏，待猜单词为 apple，在第 3 次猜测猜对。

对于第 3 局游戏，待猜单词为 apple，在第 3 次猜测猜对。注意即使每个位置都有至少一次猜测为金色，也需要额外一次猜测才算猜对。

对于第 4 局游戏，待猜单词为 above，5 次猜测均错。注意第 5 次猜测的结果并不需要输入，也不会传入 guess 函数。

对于第 5 局游戏，待猜单词为 apple，第 1 次猜测非法（不在词库内），该局游戏

直接结束。注意样例程序并不会自动识别这种情况。

对于第 6 局游戏，待猜单词为 `apple`，在第 1 次猜测猜对。

对于第 7 局游戏，待猜单词为 `cobra`。注意猜测 `kraal` 时两个 `a` 均为银色。注意由于样例程序并不知道待猜单词是什么，需要手动结束程序。

【数据范围】

对于 100% 的数据，$T = 1000$，`num_scramble` $= 8869$。每次待猜的单词均在词库中所有单词这一范围内独立均匀随机生成，这些单词在调用 `guess` 函数之前已经完全确定，不会根据和你的程序的交互过程动态构造。

交互库本身使用的时间不超过 1 秒，使用的内存不超过 $16MB$。

【提示】

由于本题只有 1 组评测数据，运行错误、超时、内存超限等错误都会导致本题总分为 0 分。建议仔细检查避免此类错误。