NOI 2022 冬令营

冬令营

测试

时间：2022 年 1 月 27 日 08:30 ～ 13:30

题目名称	序列变换	秃子酋长	猜词
题目类型	传统型	传统型	交互型
目录	oper	rrads	word
可执行文件名	oper	rrads	word
输入文件名	oper.in	rrads.in	word.in
输出文件名	oper.out	rrads.out	word.out
每个测试点时限	2.0 秒	5 秒	60 秒
内存限制	512 MiB	512 MiB	1 GiB
子任务数目	20	20	1
测试点是否等分	是	是	是

提交源程序文件名

对于 C++　语言	oper.cpp	rrads.cpp	word.cpp

编译选项

对于 C++　语言	-O2

注意事项（请仔细阅读）

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. C/C++ 中函数 main() 的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
3. 提交的程序代码文件的放置位置请参考各省的具体要求。
4. 因违反以上三点而出现的错误或问题，申述时一律不予受理。
5. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
6. 程序可使用的栈空间内存限制与题目的内存限制一致。
7. 全国统一评测时采用的机器配置为：Inter(R) Core(TM) i7-8700K CPU @3.70GHz，内存 32GB。上述时限以此配置为准。
8. 只提供 Linux 格式附加样例文件。
9. 评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以此为准。

序列变换（oper）

【题目描述】

你手里有两个长度均为 $2n$ 的合法括号序列 s_1, s_2。

在你眼中，不同的括号序列带来的视觉美感不尽相同。因此，你对具有某一种结构的括号序列特别喜欢，而讨厌具有其他一些结构的括号序列。你希望对 s_1 进行一些变换，以消除掉一些自己不喜欢的结构。

具体而言，形如 `((A)B)(C)`（其中 `A,B,C` 均为合法括号序列，下同）的结构是你喜欢的，而如下几种则是不喜欢的：`(((A)B)C)`、`((A)(B)C)`、`(A)((B)C)` 和 `(A)(B)(C)`。相应地，你有 4 种变换操作，分别表示取出原括号序列中的一个你不喜欢的子串，并将其变换为你喜欢的结构后放回原位置。

形式化地，这 4 种变换操作如下：

- 操作 1：将形如 `p(((A)B)C)q` 的串变换为 `p((A)B)(C)q`（其中`p,q`为任意串，可以为空，但不一定分别为合法括号序列，下同）；
- 操作 2：将形如 `p((A)(B)C)q` 的串变换为 `p((A)B)(C)q`；
- 操作 3：将形如 `p(A)((B)C)q` 的串变换为 `p((A)B)(C)q`；
- 操作 4：将形如 `p(A)(B)(C)q` 的串变换为 `p((A)B)(C)q`；

另外，你还希望拥有一些能够改变字符串长的操作，于是你希望能够在字符串中任意位置插入一对括号 `()`，或者将任意位置的一对括号 `()` 删除，形式化描述如下：

- 操作 5：将形如 `pq` 的串变换为 `p()q`；
- 操作 6：将形如 `p()q` 的串变换为 `pq`；

但由于一些限制条件，你执行上述两条操作的次数**最多分别不超过 2 次**（部分子任务中无此限制条件，详见数据范围部分）。

容易证明，对于任意合法括号序列实行上述 6 种操作之一，得到的仍为合法括号序列。

你现在想知道的是：凭借上述操作，能否将 s_1 变换为 s_2？如果可以的话，你希望找到一个操作次数不太多的变换方案。

【输入格式】

从文件 ***oper.in*** 中读入数据。

每个测试点由多组数据组成。

第 1 行：2 个正整数 id, T，分别表示测试点编号和数据组数。其中测试点编号可以帮助你判断测试点的特殊条件。

对于每组数据而言：

第 1 行：两个正整数 n, k，其中 k 表示你的操作步数的上限。

第 2 行：一个长度为 $2n$ 的括号序列 s_1。

第 3 行：一个长度为 $2n$ 的括号序列 s_2。

【输出格式】

输出到文件 ***oper.out*** 中。

对于每组数据分别输出若干行：

每组数据的第 1 行，一个整数 m ，表示你的操作次数。需要保证 $m \leq k$ 。

接下来 m 行，每行输出 2 个非负整数 $op\ x$ ，描述一个操作。

其中 op 为当前操作的编号，需满足 $1 \leq op \leq 6$；x 描述此次操作的位置，为方便起见，统一定义为形式化描述中 p 的长度。

你需要确保给出的 $op\ x$ 确实能描述一个符合要求的操作；在此基础上，可以证明所有符合要求的操作可以由 $op\ x$ 唯一确定。

同时你需要保证，每组数据中操作 5 和 6 的使用次数分别不得超过 2 次，有特殊说明的子任务除外。

如果有多种变换方案符合要求，输出任意一种即可。

特别地，如果该组数据无法在 k 步之内实现变换，你只需要对于该组数据输出一个 -1 即可。

【样例 1 输入】

```
0 1
3 6
(())()
((()))
```

【样例 1 输出】

```
3
5 6
4 0
6 6
```

【样例 1 解释】

在所有样例文件中，id 均为 0 。

本组数据的变换过程如下：

```
(())()
((())()
(((())))()
```

```
((()))
```

【样例 2 输入】

```
0 2
3 10
(()())
((()()
4 20
((()))()
(()())()
```

【样例 2 输出】

```
1
2 0
2
6 2
5 1
```

【数据范围】

测试点编号	$n \leq$	$k =$	特殊条件
$1 \sim 3$	10	10^5	无
$4 \sim 6$	100	10^4	
$7 \sim 8$	500	10^5	操作 5 和 6 可使用任意多次
$9 \sim 11$	1000	10^4	无
$12 \sim 13$	5000	2×10^4	
$14 \sim 16$	10^5	3×10^5	操作 5 和 6 可使用任意多次
$17 \sim 20$			无

对于 100% 的数据，$T \leq 3, n \leq 10^5, k \leq 3 \times 10^5$。

【提示】

称一个字符串 s 为**合法括号序列**，当且仅当 s 仅由数量相等的字符 (和) 组成，且对于 s 的每一个前缀而言，其中 (的数量均不少于) 的数量。特别地，空串也是合法括号序列。

秃子酋长（rrads）

【题目描述】

传说在明斯克航空航天局中，有一名强大的秃子酋长。

秃子酋长法力无边，他的头上没有头发，而且头特别硬，跑得还不慢。

这一天，豌豆射手来到了明斯克航空航天局。

秃子酋长为了考验这一位新人，给他出了这样一道题：

给一个长为 n 的排列 $a_1, \ldots, a_n$，有 m 次询问，每次询问区间 $[l, r]$ 内，排序后相邻的数在原序列中的位置的差的绝对值之和。

【输入格式】

从文件 ***rrads.in*** 中读入数据。

第一行两个数表示 n, m；

之后一行 n 个数依次表示序列 a 中的元素；

之后 m 行，每行两个数 l, r 表示一次查询。

【输出格式】

输出到文件 ***rrads.out*** 中。

对于每次询问，输出一行一个数表示答案。

【样例输入】

```
5 2
5 4 2 3 1
3 4
2 5
```

【样例输出】

```
1
5
```

【样例解释】

第一个询问，$2, 3$ 排序后为 $2, 3$，在原序列中的位置为 $3, 4$，相邻元素在原序列中位置差的绝对值之和为 $|3 - 4| = 1$；

第二个询问，$4,2,3,1$ 排序后为 $1,2,3,4$，在原序列中的位置为 $5,3,4,2$，相邻元素在原序列中位置差的绝对值之和为 $|5-3|+|3-4|+|4-2|=5$。

【数据范围】

对 10% 的数据，$n,m\leq 10^3$；
对另外 10% 的数据，$n,m\leq 5\times 10^4$；
对另外 10% 的数据，$n,m\leq 10^5$；
对另外 10% 的数据，$n,m\leq 2\times 10^5$；
对另外 20% 的数据，$|a_i-i|\leq 10$；
对另外 20% 的数据，$m=\frac{n(n-1)}{2}$；
对其余数据，无特殊限制。

对于 100% 的数据，满足 $1\leq n,m\leq 5\times 10^5$，$1\leq a_i\leq n$，$a_i$ 互不相同，$1\leq l\leq r\leq n$，所有数值为整数。

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。