Общие замечания по интерактивным задачам

После каждого действия вашей программы выводите перевод строки.

После каждого действия вашей программы делайте сброс потока вывода.

Если вы используете «`writeln`» в Паскале, «`cout << ... << endl`» в C++, «`System.out.println`» в Java, «`print`» в Python, «`Console.WriteLine`» в C#, то сброс потока вывода у вас происходит автоматически, дополнительно ничего делать не требуется. Если вы используете другой способ вывода, рекомендуется делать сброс потока вывода. Обратите внимание, что перевод строки надо выводить в любом случае. Для сброса потока вывода можно использовать «`fflush(stdout)`» в C и C++, «`flush(output)`» в Паскале, «`System.out.flush()`» в Java, «`sys.stdout.flush()`» в Python, «`Console.Out.Flush()`» в C#. Обратите внимание, что в Borland Delphi «`flush`» делать обязательно.

Типичные ошибки в интерактивных задачах:

- «`Wrong Answer`» означает, что ответ или промежуточные действия неверны, либо что ваша программа нарушила протокол. Обратите внимание, что результата проверки «~~`Presentation Error`~~» не бывает, если ваша программа нарушит формат сообщений при общении с жюри, вы все равно получите «`Wrong Answer`».
- «`Idleness Limit Exceeded`» означает, что ваша программа ожидает ввода, но данных в стандартном потоке ввода нет. Например,
  - ваша программа ошибочно ожидает ввода, а она должна вывести информацию для программы жюри либо завершиться;
  - ваша программа не вывела перевод строки или не произвела сброс потока вывода, программа жюри не получила вывод вашей программы и не может выполнить свои действия.
- «`Runtime Error`» редко означает проблемы с интерактивностью и чаще возникает из-за обычных ошибок в программе. Хотя ничего нельзя исключать.

Задача A. Игра в неразличимость (версия 1)

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Возьмем шифр одноразовых блокнотов для случая однобитовых сообщений. Игру в неразличимость можно переписать следующим образом (мы приносим в игру генерацию ключа, подчеркивая то, что ключ может быть использован лишь однократно), и переписанную версию мы назовем $IND - EAV - 1BIT$:

1. $b \leftarrow \{0, 1\}$
2. $k \leftarrow \{0, 1\}$
3. $c := b \oplus k$
4. $b' \leftarrow Adv(c)$

Шифр является абсолютно стойким если $Pr[IND-EAV-1BIT] = \frac{1}{2}$. Если атакующему удастся распознать загаданный бит с вероятностью большей $\frac{1}{2}$, он выигрывает игру.

Вам предлагается поиграть за атакующего. К счастью для Вас, известно, что генератор ключей использует плохую случайность, что повышает шансы на успех.

Вам предстоит написать программу, которая будет получать на вход бит c, выводить b', и получать результат «`YES`» или «`NO`» (совпадает ли бит b с b' или нет). Вы будете играть в игру 10000 раз, и ваша задача заключается в том, чтобы выиграть как минимум чем в 6000 случаев из них.

В этой версии генератор случайных битов является нечестной монетой, выкидывающей 0 с вероятностью p, где $|p - 0.5| \geqslant 0.2$.

Протокол взаимодействия

Происходит 10000 раундов взаимодействия. В каждом раунде взаимодействия происходит три передачи сообщений:

1. Интерактор выдает вам один бит c.
2. Вы отвечаете b' — свою догадку для b.
3. Интерактор возвращает «`YES`» или «`NO`», в зависимости от того, $b = b'$ или нет.

В примере показано два раунда возможного взаимодействия. После совершения 10000 раундов взаимодействия завершите вашу программу.

Не забывайте делать `flush` после каждого вывода.

Пример

стандартный ввод	стандартный вывод
1 0	1 YES 0 NO

Задача B. Игра в неразличимость (версия 2)

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Возьмем шифр одноразовых блокнотов для случая однобитовых сообщений. Игру в неразличимость можно переписать следующим образом (мы приносим в игру генерацию ключа, подчеркивая то, что ключ может быть использован лишь однократно), и переписанную версию мы назовем $IND - EAV - 1BIT$:

1. $b \leftarrow \{0, 1\}$
2. $k \leftarrow \{0, 1\}$
3. $c := b \oplus k$
4. $b' \leftarrow Adv(c)$

Шифр является абсолютно стойким, если $Pr[IND{-}EAV{-}1BIT] = \frac{1}{2}$. Если атакующему удастся распознать загаданный бит с вероятностью большей $\frac{1}{2}$, он выигрывает игру.

Вам предлагается играть за атакующего. К счастью для Вас, известно, что генератор ключей использует плохую случайность, что повышает шансы на успех.

Вам предстоит написать программу, которая будет получать на вход бит c, выводить b', и получать результат «`YES`» или «`NO`» (совпадает ли бит b с b' или нет). Вы будете играть в игру 10000 раз, и ваша задача заключается в том, чтобы выиграть как минимум в 6000 из них.

В этой версии генератор случайных битов имеет небольшой период (не больше 1000).

Протокол взаимодействия

Происходит 10000 раундов взаимодействия. В каждом раунде взаимодействия происходит три передачи сообщений:

1. Интерактор выдает вам один бит c.
2. Вы отвечаете b' — свою догадку для b.
3. Интерактор возвращает «`YES`» или «`NO`», в зависимости от того, $b = b'$ или нет.

В примере показано два раунда возможного взаимодействия. После совершения 10000 раундов взаимодействия завершите вашу программу.

Не забывайте делать `flush` после каждого вывода.

Пример

стандартный ввод	стандартный вывод
1 0	1 YES 0 NO