Задача A. Соберите команду!

Имя входного файла:	`team.in`
Имя выходного файла:	`team.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

При формировании команды для олимпиад по программированию важную роль играет не только ее состав, но и распределение ролей. Например, если один человек хорошо знает алгоритмы, но не очень хорошо умеет программировать, то не разумно, чтобы он писал все программы в команде.

Андрей решил создать школьную команду, чтобы участвовать в командных олимпиадах по программированию. В команду он пригласил двух своих одноклассников: Петю и Диму. Теперь у них возник вопрос распределения ролей. Андрей считает, что в их команде есть три роли: «программист», «математик» и «тестер». Первый из них, соответственно, должен заниматься написанием программ, второй — придумыванием решений, а третий должен составлять тесты, для программ и решений.

Поскольку мальчики очень хорошо знают друг друга, им не составило большого труда определить, насколько эффективно каждый из них выполняет ту или иную роль. Таким образом, в распоряжении Андрея (который был избран капитаном команды) оказалась табличка размером 3 на 3, в которой первая строка соответствует Пете, вторая — Диме, а третья — Андрею. Столбцы же этой таблички соответствуют: первый — роли «программиста», второй — роли «математика», третий — роли «тестера». Таким образом, например, число, стоящее во второй строке третьего столбца показывает, насколько эффективно Дима выполняет роль «тестера».

Андрей, Петя и Дима хотят так распределить роли между собой, чтобы их команда действовала как можно более эффективно. Разумеется, при этом на каждую роль должен быть назначен ровно один человек. Эффективность распределения ролей, при котором Андрей назначен на роль, которую он выполняет с эффективностью A, Дима назначен на роль, которую он выполняет с эффективностью D, а Петя назначен на роль, которую он выполняет с эффективностью P, будем считать равной $\sqrt{A^2 + D^2 + P^2}$.

Помогите мальчикам найти распределение ролей между ними, которое обладает максимально возможной эффективностью.

Формат входного файла

Входной файл содержит три строки, каждая из которых содержит по три целых числа, — табличку, которую составили Андрей, Петя и Дима. Все числа во входном файле положительны и не превосходят 1000.

Формат выходного файла

Выведите в выходной файл единственное вещественное число — максимальную эффективность, которую могут достичь Андрей, Дима и Петя, распределив между собой роли. Выведенный ответ должен отличаться от правильного не более, чем на 10^{-6}.

Примеры

team.in	team.out
1 1 1 1 1 1 1 1 1	1.7320508075688772
1 2 3 6 5 4 7 8 9	11.0

Задача B. Готовьтесь к олимпиадам!

Имя входного файла:	`prepare.in`
Имя выходного файла:	`prepare.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Подготовка к олимпиадам очень важна для успешного выступления на них. Эту истину хорошо понимает Вася. Поэтому он решил подойти к своей подготовке с научной точки зрения. Вася считает, что уровень подготовки к олимпиаде определяется неким интегральным показателем, который он называет *умение решать задачи*.

Прочитав несколько книг по олимпиадному программированию, Вася узнал, что готовится к олимпиадам можно в двух направлениях: в теоретическом и практическом. На подготовку к ближайшей олимпиаде у Васи осталось n дней. Исходя из календаря своих биоритмов, Вася для каждого из этих дней определил два числа: t_i — насколько увеличится его умение решать задачи, если он i-ый день посвятит изучению теории, и p_i — насколько увеличится его умение решать задачи, если он i-ый день посвятит практике. При этом в один день Вася может заниматься либо только теорией, либо только практикой, и обязательно хотя бы один день должен быть посвящен теории, и хотя бы один день — практике.

Помогите Васе составить такое расписание подготовки, про котором его умение решать увеличится до максимального возможного значения. Считайте, что перед началом подготовки показатель умения решать задачи у Васи равен нулю.

Формат входного файла

Первая строка входного файла содержит число n ($2 \le n \le 100$). Вторая строка входного файла содержит n разделенных пробелами целых чисел: p_1, p_2, …, p_n. Третья строка входного файла содержит n разделенных пробелами целых чисел: t_1, t_2, …, t_n.

Все числа p_i и t_i положительны и не превосходят 1000.

Формат выходного файла

В выходной файл выведите максимальное значение умения решать задачи, которое может достигнуть Васе за n дней. Обратите внимание на то, что хотя бы один день Вася должен заниматься теорией и хотя бы один день — практикой.

Примеры

prepare.in	prepare.out
2 1 2 2 1	4
3 1 2 3 1 2 3	6

Задача C. Правильно расставьте стулья!

Имя входного файла:	`chairs.in`
Имя выходного файла:	`chairs.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Знаете ли вы, что то, как Вы сидите во время соревнования, может серьезно повлиять на Ваши результаты?

В частности, положение стульев относительно столов и компьютера очень сильно влияет на взаимоотношения в команде.

Однажды всем известная олимпиадная команда *Dream Team* участвовала в соревнованиях *NIRC*. Каждой команде на этом соревновании предоставлялся один компьютер, который стоял на треугольном столе, и три стула.

Наиболее удобным в команде *Dream Team* считается такое расположение участников команды, при котором каждый из них сидит за своей стороной треугольного стола, причем посередине этой стороны. Разумеется, стулья придется расставить таким же образом.

В то же время очень важно, чтобы во время олимпиады участники команды сидели не очень далеко друг от друга. Капитан команды *Dream Team* считает, что для оценки того, насколько далеко сидят участники команды друг от друга, необходимо вычислить среднее расстояние между ними.

В данном случае придется вычислить среднее расстояние между серединами сторон треугольного стола. Напишите программу, вычисляющую эту величину.

Формат входного файла

Входной файл содержит три положительных целых числа, не превосходящих 100, — длины сторон стола. Гарантируется, что стол с такими длинами сторон имеет ненулевую площадь.

Формат выходного файла

В выходной файл выведите среднее расстояние между центрами двух сторон стола. Выведенный ответ должен отличаться от правильного не более, чем на 10^{-6}.

Примеры

chairs.in	chairs.out
3 4 5	2.00000000
5 5 7	2.83333333

Задача D. Напишите шаблон!

Имя входного файла:	`template.in`
Имя выходного файла:	`template.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Во время олимпиады при написании решений к задачам часто приходится выполнять одни и те же действия, например, открыть и закрыть входной и выходной файлы и т.д.

Каждый член Вашей команды неоднократно сталкивался с этой проблемой в своем олимпиадном прошлом, и в конечном счете придумал для себя какой-то шаблон.

Шаблон — это тело программы, являющееся общим для всех набираемых программ, и поэтому набираемое всего один раз в начале соревнования.

Кроме того, каждый из вас пишет в основном на каком-либо языке программирования. Поэтому и шаблон у него готов для этого языка.

Вы собрались вместе и решаете, на каком языке будете писать. Вы хотите сэкономить время, необходимое для написания шаблона, поэтому решаете выбрать язык, написание шаблона на котором займет наименьшее время.

Предположим, что клавиатура, используемая вами на соревнованиях, прямоугольная, и каждый символ, встречающийся на ней, присутствует ровно один раз. Введем на этой клавиатуре прямоугольную декартову систему координат так, чтобы координаты всех клавиш были целочисленными и принимали значения $1 \ldots W$ по координате x и $1 \ldots H$ по координате y, причем левому нижнему углу соответствует координата $(1, 1)$.

Расстоянием между символами на клавиатуре назовем максимальную из разностей координат клавиш, на которых расположены эти символы. Например, если символу `A` соответствует клавиша с координатами (X_A, Y_A), а символу `B` — клавиша с координатами (X_B, Y_B), то расстояние между `A` и `B` будет равно $max(|X_A - X_B|, |Y_A - Y_B|)$.

Время, затрачиваемое членом Вашей команды на набор шаблона, равно сумме расстояний между первым и вторым, вторым и третьим, ..., предпоследним и последним символами шаблона (перевод строки за символ шаблона не считается).

Перед соревнованием каждый из вас предложил свой шаблон для своего языка программирования. Выберите оптимальный шаблон.

Формат входного файла

В первой строке входного файла находятся два числа W, H ($1 \le W, H \le 100$) — ширина и высота клавиатуры.

В следующих H строках находятся по W символов ASCII с возможными кодами от 32 до 126 — раскладка клавиатуры. Каждый символ встречается в раскладке не более одного раза.

После описания раскладки следует пустая строка.

Далее следуют три блока, описывающие предлагаемый шаблон.

Первая строка каждого блока — это название языка программирования. Следующие несколько строк (не менее одной) описывают собственно шаблон. Блок заканчивается пустой строкой. Любой символ, входящий в шаблон, можно найти на клавиатуре. Количество символов в каждом блоке не более 10000 (переводы строки не учитываются).

Формат выходного файла

В первой строке выходного файла выведите название языка, шаблон для которого выбран Вашей командой. Во второй строке — время, требуемое для набора данного шаблона. Если возможных вариантов несколько, выведите тот из них, который идет раньше во входном файле.

Примеры

template.in
3 1 abc LanguageA a LanguageAB ab LanguageB b
template.out
LanguageA 0

template.in

```
10 9
1234567890
!@#$%^&*()
qwertyuiop
QWERTYUIOP
asdfghjkl;
ASDFGHJKL:
zxcvbnm,./
ZXCVBNM<>?
 []{}='"-|

Pascal
begin
  reset(input, 'filename.in');
  rewrite(output, 'filename.out');
end.

C
int main()
{
    freopen("filename.in", "r", stdin);
    freopen("filename.out", "w", stdout);
}

Java
import java.io.*;
import java.util.*;
public class Main {
    public static void main(String[] args) throws IOException {
        Scanner in = new Scanner(new File("filename.in"));
        PrintWriter out = new PrintWriter("filename.out");
        //TODO
        in.close();
        out.close();
    }
}
```

template.out

```
Pascal
278
```

Обратите внимание на то, что во втором примере в начале последней строки описания раскладки клавиатуры находится пробел. В этом примере отступ строк кода формируется с помощью пробелов.

Задача E. Не играйте во время олимпиады!

Имя входного файла:	`dontplay.in`
Имя выходного файла:	`dontplay.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Участникам соревнований не следует играть во время олимпиады, как и заниматься другими посторонними делами. Иначе, результаты могут быть весьма неожиданными...

Вот и сегодня, Дима и Федя поспорили, кто из них будет писать простую задачу. Для этого они сыграли несколько туров известной игры «камень, ножницы, бумага».

Напомним правила этой игры. Один ее тур проходит следующим образом: каждый из игроков одновременно загадывает одну из трех «фигур»: «камень», «ножницы» или «бумагу», после чего они одновременно показывают эти фигуры противнику. Считается, что «камень» сильнее «ножниц», «ножницы» сильнее «бумаги», «бумага» сильнее «камня». Если первый из игроков задумал фигуру, которая сильнее, чем у второго игрока, то ему начисляется одно очко; если фигура сильнее у второго игрока, то это очко достается ему; в противном случае, тур считается закончившимся ничьей, и очки никому не добавляются. По истечении нескольких туров выигрывает тот, у кого больше очков.

Но вот беда: под конец игры Дима и Федя забыли, каков же счет игры. Однако они помнят, сколько раз каждый из них загадывал «камень», «ножницы» или «бумагу». Как же определить, кто из них выиграл?

Назовем «исходом» игры разницу между числом очков, набранных первым игроком, и числом очков, набранных вторым игроком. Помогите мальчикам и найдите все возможные «исходы» игры.

Формат входного файла

В первой строке находятся три неотрицательных целых числа R_1, S_1, P_1 — число соответственно «камней», «ножниц» и «бумаг» первого игрока.

Во второй строке находятся R_2, S_2, P_2 — аналогичные числа для второго игрока. Известно, что $R_1 + S_1 + P_1 = R_2 + S_2 + P_2$. Также известно, что $R_1 + S_1 + P_1 \leq 20$.

Формат выходного файла

В первой строке выведите число возможных различных «исходов» игры.

Во второй строке выведите собственно «исходы» в порядке возрастания.

Примеры

dontplay.in	dontplay.out
0 1 0 0 0 1	1 1
1 1 1 1 1 1	3 -3 0 3

Задача F. Читайте условие внимательно!

Имя входного файла:	`careful.in`
Имя выходного файла:	`careful.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Очередной «`Wrong Answer, test 3`»! Да что же это такое?!

«Может быть, мы невнимательно читаем условие?» — спросил капитан команды. «Давайте его перечитаем!»

Некоторые условия могут быть неправильно поняты, если при их прочтении пропущено одно или несколько слов. Вот и сейчас Вы задумались над тем, сколько раз может встретиться определенное, безусловно, важное для понимания слово в тексте условия задачи.

Посчитайте, сколько раз данное слово встречается в тексте условия задачи.

Учтите, что слово, которое Вы ищете, в тексте может быть написано как со строчной буквы, так и с заглавной (если первый символ слова — буква).

Также слово может быть разорвано при переносе строки. В этом случае в конце строки после части искомого слова будет стоять символ дефиса («`-`», ASCII-код этого символа 45), а на следующей строке слово будет продолжено с места разрыва.

Случаи, в которых слово, которое Вы ищете, является подстрокой другого слова, также надо учитывать.

Формат входного файла

В первой строке находится слово, которое надо проанализировать. Символы, из которых может состоять слово, включают большие и маленькие латинские буквы, а также цифры. Длина слова не меньше единицы.

Во второй строке находится число N — число строк в условии задачи.

В последующих N строках находится само условие задачи. Символы, входящие в условие, имеют ASCII-коды от 32 до 126 включительно.

Размер входного файла не превышает 10^6 символов, включая переводы строки.

Формат выходного файла

Выведите в первой и единственной строке выходного файла число — сколько раз данное слово встречается в условии задачи.

Примеры

careful.in
```
problem
2
Problem A.
Output 0.
```

careful.out
```
1
```

careful.in
```
Output
3
Output the number of letters in the problem's statement. Then out-
put zero, please, to mark the end of your output. All your outputs
should be different.
```

careful.out
```
4
```

Задача G. Проверяйте ответы!

Имя входного файла:	`control.in`
Имя выходного файла:	`control.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В некоторых задачах ответы, то есть содержимое выходных файлов, может быть достаточно сложным. В таких случаях полезным бывает проверить ответ по нескольким простым критериям, чтобы убедиться, что он если не правильный, то хотя бы разумный.

Рассмотрим эту стратегию на примере задачи о поиске максимального потока минимальной стоимости.

Напомним необходимые определения.

Сеть — это ориентированный граф $G = (V, E)$, каждому ребру $(u, v) \in E$ которого поставлено в соответствие число $c(u, v) \geq 0$, называемое *пропускной способностью* ребра. В случае $(u, v) \notin E$ удобно положить $c(u, v) = 0$. В сети выделены две вершины: *исток* s и *сток* t. Каждому ребру можно также сопоставить неотрицательное число $s(u, v)$, называемое *стоимостью пропускания единицы потока* по ребру.

Поток в сети G — это функция $f : V \times V \to \mathbb{R}$, обладающая следующими тремя свойствами:

- *Подчиненность пропускной способности* — $\forall u, v \in V\ f(u, v) \leq c(u, v)$;
- *Антисимметричность* — $\forall u, v \in V\ f(u, v) = -f(v, u)$;
- *Сохранение потока* — $\forall u \in V \setminus \{s, t\}\ \sum_{v \in V} f(u, v) = 0$.

Величина потока определяется как сумма потоков по всем ребрам, выходящим из истока.

Стоимость потока определяется как следующая сумма:

$$\sum_{(u,v) \in E, f(u,v) > 0} f(u, v) \cdot s(u, v)$$

В рассматриваемой нами задаче требуется найти такой поток, величина которого максимально возможная, а среди всех таких тот, у которого стоимость наименьшая.

Правильный ответ является потоком, поэтому мы должны проверить этот факт. К сожалению, мы не знаем ни мощности, ни стоимости правильного ответа, но для самопроверки можем вычислить эти значения для своего ответа и вывести их.

Вам будет дана исходная сеть и функция $f : V \times V \to \mathbb{R}$. Необходимо определить, является ли данная функция потоком в данной сети, и если является, то найти величину и стоимость этого потока.

Формат входного файла

В первой строке входного файла находится число N ($2 \leq N \leq 200$) — число вершин в сети.

В следующих N строках заданы пропускные способности ребер сети — по N чисел в каждой строке. j-е число в $i + 1$-ой строке задает $c(i, j)$. Пропускные способности неотрицательны и не превосходят 10^4. Гарантируется, что $c(i, i) = 0$.

Далее идет пустая строка.

В следующих N строках заданы в аналогичном формате стоимости пропускания единицы потока для ребер сети. Стоимости неотрицательны и не превосходят 10^4. Если $c(i, j) = 0$, то $s(i, j) = 0$.

После этого блока также идет пустая строка.

В следующих N строках заданы значения функции f в аналогичном формате. Эти значения не превосходят 10^4 по абсолютному значению.

Истоком сети служит вершина с номером 1, стоком — вершина с номером N.

Формат выходного файла

Если данная функция не является потоком, выведите в первой и единственной строке слово `NO`. В противном случае, в первой строке выведите `YES`, а во второй строке выведите F — величину и S — стоимость потока, разделив их пробелом.

Примеры

control.in	control.out
4 0 1 3 0 0 0 0 2 0 0 0 4 0 0 0 0 0 2 1 0 0 0 0 2 0 0 0 1 0 0 0 0 0 1 2 0 -1 0 0 1 -2 0 0 2 0 -1 -2 0	YES 3 8
4 0 1 3 0 0 0 0 2 0 0 0 4 0 0 0 0 0 7 6 0 0 0 0 5 0 0 0 4 0 0 0 0 0 2 1 0 -2 0 0 2 -1 0 0 1 0 -2 -1 0	NO

Задача H. Генерируйте тесты!

Имя входного файла:	`testgen.in`
Имя выходного файла:	`testgen.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

При решении некоторых задач иногда возможно получить результат *Wrong Answer*, *Runtime Error* или *Time Limit Exceeded* на довольно большом по номеру тесте.

Часто бывает, что ошибка скрывается очень глубоко, и составление тестов вручную либо невозможно по причине большого объема ввода, либо до критического теста трудно додуматься сразу. В таком случае помогает написание *генератора тестов* — небольшой программы, которая создает тесты для неподдающейся задачи.

Рассмотрим задачу, для которой ответом является некоторое число. *Максимальным* назовем тест, ответ на который является максимально возможным для данного типа задачи. Некоторые типичные ошибки в программе можно найти, запуская ее на максимальных тестах. Часто максимальные тесты достаточно просто генерируются небольшим генератором тестов.

Пусть дана простая задача: среди чисел от 2 до N найти такое число, что количество делителей у него максимально возможное, и вывести это количество делителей.

Найдите количество максимальных тестов для ограничения $2 \le N \le K$.

Формат входного файла

В первой строке находится единственное число K ($2 \le K \le 10^7$).

Формат выходного файла

Выведите количество максимальных тестов для данной задачи с данным ограничением K.

Примеры

testgen.in	testgen.out
11	6
12	1

В первом примере максимальными тестами являются все числа, начиная с 6 и заканчивая 11, потому что на отрезке $[2, 11]$ число делителей числа не превосходит 4. Во втором примере это единственный максимальный тест 12.

Задача I. Выиграйте олимпиаду!

Имя входного файла:	`win.in`
Имя выходного файла:	`win.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Плоха та команда, которая не мечтает выиграть олимпиаду. Однако для того, чтобы выиграть необходимо четко понимать свои возможности, правильно рассчитывать свои силы и четко реализовывать свои идеи.

Уже известная вам команда *Dream Team* хорошо умеет все это делать. В частности, в самом начале олимпиады им удается для каждой из предлагаемых для решения задач определить, сколько времени им потребуется на ее решение.

Таким образом, они могут эффективно распределять свое время в течение олимпиады. Однако для победы этого недостаточно. Требуется решить максимальное количество задач.

Напишите программу, которая по времени, которое требуется команде *Dream Team* на решение каждой задачи, определит, какое максимальное количество задач эта команда может решить за 180 минут, в течение которых длится Интернет-олимпиада в базовой номинации.

Формат входного файла

Первая строка входного файла содержит число n — количество задач. Вторая строка содержит n чисел: $t_1, t_2, \ldots, t_n$, где t_i есть время, которое требуется на решение i-ой задаче. Все числа во входном файле положительны и не превосходят 10^5. Времена заданы в секундах. Напомним, что одна минута содержит 60 секунд.

Формат выходного файла

Выведите в выходной файл одно число — максимальное количество задач, которые успеет решить команда *Dream Team*.

Примеры

win.in	win.out
2 60 10800	1
3 3600 7200 10800	2