Задача A. Abat-jour (Division 1 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Великий математик Джон часами разглядывал свой новый абажур. Лампочки в нем образовывали правильный многоугольник. Знакомая история...

Яркое рекламное объявление гласило, что этот потрясающий и модный абажур M-регулярен и имеет N лампочек, покрашенных в некоторые из K фирменных цветов (при этом N делится на M). Абажур называется M-регулярным, если для любых M последовательных лампочек следующие M за ними имеют те же цвета и в том же порядке. Джона ждал сюрприз: одна из лампочек была грязно-серого цвета (который явно не входил в число фирменных). Соответственно, с M-регулярностью тоже возникли проблемы. Абажур был полу-M-регулярным, то есть для любых M подряд идущих ламп цвета следующих M за ними могли отличаться не более, чем в одной позиции. Воображение Джона ожидаемо породило тысячи полу-M-регулярных абажуров с одной грязно-серой лампочкой...

Вы были наняты исследователем, чтобы приоткрыть завесу тайны гения Джона. Вы должны посчитать количество различных полу-M-регулярных абажуров из N лампочек с одной грязно-серой лампочкой и другими, раскрашенными в какие-то из K фирменных цветов. Ответ требуется вывести по модулю $1\,000\,000\,007$. Абажуры, переводимые друг в друга поворотом, считать совпадающими.

Формат входных данных

На вход подаются три целых числа N, M, K, разделенные пробелами($2 \leqslant N, M, K \leqslant 500$, N делится на M) — число лампочек в абажуре, параметр регулярности и число фирменных цветов соответственно.

Формат выходных данных

Выведите единственное число — ответ на задачу по модулю $1\,000\,000\,007$.

Примеры

standard input	standard output
8 2 2	16

Задача B. Winery

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	1 second
Ограничение по памяти:	256 mebibytes

Одной начинающей винодельческой компании потребовалось решить непростую задачу по хранению вина. Как вам всем хорошо известно, вина отличаются главным образом годом сбора урожая винограда. Поэтому различают K категорий вина, которые отличаются только возрастом. Первая категория вина имеет возраст в годах в полуинтервале $[0,\ a_1)$, возраст вина второй категории имеет возраст в полуинтервале $[a_1, a_1 + a_2)$, третей — $[a_1 + a_2, a_1 + a_2 + a_3)$, ..., K-ой — $[a_1 + a_2 + \ldots + a_{K-1}, a_1 + \ldots + a_K)$. В целях повышения качества вина в международной конвенции принято, что a_{i+1} обязано делиться на a_i.

Вина разной категории нужно хранить по-разному: разные температурные режимы, разная влажность и так далее. Для предоставления полного ассортимента вин в компании было принято решение обязательно хранить все категории вин. Однако, в целях экономии на заводе построили ровно K погребов для хранения.

Помогите владельцам организовать процесс так, чтобы каждый год все категории вин, которые могли быть произведены к этому году, были всегда в наличии. В первый год урожай винограда был очень удачен, и все погреба были заполнены вином из винограда текущего года (возраста 0 лет).

Составьте план для компании на N следующих лет. Про каждый год выведите в какой погреб (нумерация с 1) нужно залить вино нового урожая. Если в этом погребе уже было вино, то оно будет немедленно продано.

Если в этот год не нужно заливать вино в погреба, то выведите 0. В этом случае молодое вино будет распродано на ближайшей ярмарке. Слишком старое вино, которое не относится ни к одной из категорий так же распродается.

Формат входных данных

В первой строке записаны два числа N и K ($1 \le N \le 10^5$, $1 \le K \le 15$). В следующей строке записаны K чисел a_i ($1 \le a_i \le 10^9$). Гарантируется, что a_{i+1} делится на a_i.

Формат выходных данных

Выведите N чисел: для каждого года — номер погреба, в который нужно заливать молодое вино в соответствующий год. Выводите 0, если в этот год вино нужно распродать. После очередного сбора урожая в погребах должны находиться вина всех категорий, у которых минимальный возраст не превосходит возраст компании. Пока количество категорий вин не достигло K, вина одной категории могут находиться в разных погребах.

Примеры

standard input	standard output
3 1 1	1 1 1
3 2 1 1	1 2 1
10 2 2 4	0 1 0 1 0 2 0 2 0 1
10 3 1 2 2	1 1 2 2 3 3 1 1 2 2

Задача C. Cdecl

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Начинающие программисты на языке C могут легко проинтерпретировать такие простые объявления:

```
int      bar[]; // bar массив целых чисел
char    *bar;   // bar указатель на тип char
```

Но всё сложнее, когда встречаются такие объявления:

```
char *(*(**foo[][])())[]; // ???
```

Некто Бьёрн решил создать новый интерпретируемый язык на основе C и назвать его C−−. Вам предстоит разработать часть интерпретатора этого языка, которая по некоторому выражению определяет тип выражения и задает определение этой переменной. Например, если вы хотите присвоить переменной `tmp` значение выражения `(*((**foo[10][4]))())`, то переменная должна быть объявлена как `char (*tmp[])`.

Вам требуется вывести определение переменной, тип которой в точности совпадает с типом некоторого выражения.

При объявлении переменная идентифицируется по имени. Имя переменой дополняется ровно одним из встроенных типов (`'int'`, `'char'` или `'void'`) и несколькими дополнительными деклараторами. В объявлении переменной участвует ровно один встроенных типов, который записывается слева от остального выражения.

Встроенные типы дополняются следующими деклараторами:

- `*` — указатель на "...". Означает что переменная это указатель на какой-то тип.
- `[]` — массив из объектов некоторого типа. В этом языке массивы не имеют фиксированных размеров. Пара `[]` может встречаться только справа от имени переменной.
- `()` — функция возвращающая некоторый тип. В языке предполагается, что функции могут принимать на вход любые параметры, поэтому их типы нам не важны. Если пара скобок стоит справа от имени переменной, то это функция. Остальные пары круглых скобок задают приоритеты в выражении.

Определение массива и функции имеют приоритет выше, чем указатель, что позволяет однозначно интерпретировать любое выражение.

Корректное определение переменной удовлетворяет следующей формальной грамматике:

```
decl : type_spec declarator
type_spec : 'void' | 'char' | 'int'
declarator : pointer direct_declarator | direct_declarator
pointer : '*' | '*' pointer
direct_declarator : id | '(' declarator ')' | direct_declarator '[' ']'
                        | direct_declarator '(' ')'
```

где `id` это идентификатор переменной, который может состоять из латинских символов, цифр и знака подчеркивания. Идентификатор не может начинаться с цифры.

Правила интерпретации объявления переменной:

1. Найдите идентификатор. Можно считать, что это имя и записать "объявление переменной, которая является ...". Дальше всё рассматривается в порядке сначала правее имени, затем левее имени.

2. Если справа ничего нет или есть ")", то переходим к шагу 4.

3. Если справа находится определение массива `[]` или функции `()`, то для каждой пары мы можем записать "массив из ...", или "функция возвращающая ...". Если символы справа закончились или вы встретили ")", то переходим к пункту 4.

4. Если слева нет ничего или открывающая скобка, то переходим к пункту 6.

5. Если вы встретили указатель "*", то до тех пор пока слева не кончатся символы или не появится "(" записываем "указатель на ..."

6. На этом шаге или уже всё рассмотрено, или встречается выражение в круглых скобках, которое мы будем рассматривать начиная с пункта 2. Если всё рассмотрено, то мы можем выписать тип и закончить разбор.

Например, `foo` это "массив из " "массивов из" "указатель на" "указатель на" "функцию, возвращающую" "указатель на" "массив из" "указателей на" char.

Корректное выражение с переменной имеет следующую грамматику:

```
exp  : unary_exp
unary_exp  : postfix_exp | *unary_exp
postfix_exp  : primary_exp | postfix_exp '[' number']' | postfix_exp '(' ')'
primary_exp  : id | '(' exp ')'
```

где `number` это целое число. Операция индексации `[]` или вызова функции `()` имеют приоритет выше, чем разадресация указателя `*`. Скобки вокруг имени имеют выше приоритет, чем `[]` или `()`.

Правила интерпретации выражения:

1. Если выражение `exp` записано как `id`, то `exp` имеет тот же тип, что и `id`

2. Тип выражения `(exp)` совпадает с типом выражения `exp`

3. Если выражение `exp` имеет тип "указатель на" "некоторый тип", то `*exp` имеет тип "некоторый тип"

4. Если выражение `exp` имеет тип "массив из" элементов "некоторый тип", то, выражение `exp[number]` имеет тип "некоторый тип"

5. Если выражение `exp` имеет тип "функция возвращающая" "некоторый тип", то выражение `exp()` имеет тип "некоторый тип"

Вашему модулю интерпретатора будут подаваться на вход корректные выражения. Кроме того, в языке C−− функции могут возвращать функции и массивы. Ваша задача вывести корректное объявление переменной в языке C−−, тип которой в точности совпадает с типом выражения.

Формат входных данных

В первой строке записано объявление переменной. Вторая строка содержит выражение с этой переменной. Третья строка содержит имя переменной, объявление которой необходимо вывести. Длина каждого из объявлений не превышает 300 000 символов. Длина третьей строки не превышает 1 000. В выражениях могут быть дополнительные пробелы.

Формат выходных данных

Выведите одну строку с объявлением переменной. Объявление должно соответствовать грамматике и иметь тот же тип, что и выражение. Если правильных объявлений, различающихся только скобками, несколько, то выведите любое из них. Длина объявления не должна превышать 1 000 000 символов.

Примеры

standard input	standard output
char *(*(**foo [][]) ())[] (*((* *(foo[10][4]))())) bar	char (*bar[])

Задача D. Matrix (Division 1 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Дана квадратная матрица размером $N \times N$, состоящая из нулей и единиц. Требуется изменить в ней ровно L элементов так, чтобы определитель матрицы стал нечетным. Разрешается менять только 0 на 1 или 1 на 0, причем один и тот же элемент менять дважды нельзя.

Формат входных данных

Первая строка входных данных содержит два целых числа N и L ($3 \le N \le 3000$, $1 \le L \le N$). В следующих N строках задана матрица. Каждая из этих строк имеет длину N и состоит только из нулей и единиц. j-й символ в i-й сверху строке задаёт элемент (i, j).

Формат выходных данных

В случае, если решения не существует, выведите одну строку "`-1 -1`". Иначе выведите L строк, каждая из которых должна содержать два целых числа — номера строки и столбца изменяемого элемента. Строки и столбцы нумеруются с единицы. Если решений несколько, то вывести требуется любое из них.

Примеры

standard input	standard output
3 3 100 001 010	1 2 2 2 1 3
3 1 110 110 110	-1 -1

Задача E. Stable network

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	1 second
Ограничение по памяти:	256 mebibytes

Требуется организовать компьютерную сеть из N компьютеров так, чтобы при выходе из строя любых двух компьютеров сохранялась возможность передачи информации между любыми двумя оставшимися непосредственно или через другие, не вышедшие из строя.

Напишите программу, определяющую, какое наименьшее число соединений компьютеров между собой при этом можно сделать.

Формат входных данных

На вход подается одно число N — количество компьютеров, которые необходимо соединить в сеть ($4 \le N \le 10\,000$).

Формат выходных данных

Выдайте одно число — наименьшее число соединений компьютеров между собой, с помощью которого можно организовать требуемую сеть.

Примеры

standard input	standard output
4	6

Задача F. Dice Roll

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	3 seconds
Ограничение по памяти:	256 mebibytes

Леша и Сережа играют в следующую игру. В самом начале игры Сережа платит Леше 6 у.е. Далее он делает не более N бросков M-гранной кости (на гранях кости написаны числа $1, \ldots, M$, вероятности выпадения всех этих чисел одинаковы). Броски совершаются последовательно, Сережа видит результаты предыдущих бросков и в любой момент может остановиться. Игра заканчивается либо когда сделано N бросков, либо когда Сережа останавливается. В конце игры считается среднее арифметическое результатов всех бросков (т.е. $X = (a_1 + \cdots + a_k)/K$, где K — количество бросков, а a_i — результаты бросков), и Леша возвращает Сереже X у.е.

Сережа хочет потерять в результате игры как можно меньше денег. Понятно, что для этого ему нужно максимизировать среднее арифметическое. Найдите математическое ожидание среднего арифметического в конце игры, если Сережа играет оптимально.

Формат входных данных

Единственная строка входных данных содержит два разделенных пробелом целых числа N — максимальное количество бросков и M —количество граней кости ($1 \leqslant N \leqslant 100$, $1 \leqslant M \leqslant 10000$).

Формат выходных данных

Единственная строка вывода должна содержать единственное число X — ответ на задачу. Ответы с абсолютной или относительной погрешностью менее 10^{-9} считаются правильными.

Примеры

standard input	standard output
3 6	4.0185185185

Задача G. GIT (Division 1 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Компания по разработке программного обеспечения использует систему GIT для контроля за версиями своего исходного кода.

GIT позволяет разработчикам командой commit помещать свои правки кода в центральное хранилище. Каждый commit является атомарным и успешным помещением изменений кода в хранилище. Также каждый commit имеет уникальную временную метку того, когда он произошел. Таким образом центральное хранилище линейно изменяется со временем посредством команд commit от разработчиков.

В начале процесса разработки программного обеспечения хранилище GIT имеет одну предопределенную ветвь исходного кода с именем “master”, которая содержит весь код. Каждый разработчик в любое время может сделать новую ветвь исходного кода от любой уже существующей (это процесс называется ответвлением) путем копирования в новую ветвь всего содержимого и состояния существующей ветви в момент ответвления. После этого каждый разработчик может осуществлять правки новой ветви вместо “master” по умолчанию — все изменения будут сохраняться в данной ветке и не будут влиять на ветвь “master” или какую-либо другую. Данная техника часто используется для разработки нового функционала программ.

После того, как новый функционал готов, разработчик может слить его ветвь B обратно к ветви “master” или любой другой существующей ветви A. Подразумевается, что каждое слияние атомарно и успешно выполняется без конфликтов в исходном коде. Ветвь A после этого содержит все изменения как из ветви A, так и из ветви B до момента времени слияния этих веток. Обе ветви продолжают существовать после слияния, причем ветвь A является результатом слияния.

Компания уже выпустила несколько версий своего продукта и имеет несколько ветвей исходного кода в центральном хранилище. Недавно компания обнаружила, что в одном из изменений кода содержится критическая ошибка. Вам необходимо определить все ветви, в которых данная ошибка присутствует по состоянию на текущий момент.

Формат входных данных

Входные данные содержат архив команд, которые были отданы системе GIT до текущего момента. В первой строке дано целое число N — общее количество команд в архиве ($1 \le N \le 2000$). Считается, что команда C_i была отдана разработчиком и полностью исполнена системой GIT в момент времени i.

На следующих N строках даны команды системе GIT. Они могут быть в одном из следующих форматов:

- commit (branch-name) — команда commit в ветвь с именем “branch-name”;
- branch (new-branch) from (existing-branch) — создание новой ветви “new-branch” от существующей ветви “existing-branch”;
- merge (from-branch) to (to-branch) — слияние ветви с именем “from-branch” в ветвь с именем “to-branch”

Имена ветвей содержат только маленькие латинские буквы a-z и цифры 0-9, а длина каждого имени не превосходит 16 символов.

Следующая строка содержит одно целое число K — количество тестов для решения ($1 \le K \le 2000$). На следующих K строках дано целое число T — время внесения ошибочного кода в систему командой commit. Ошибка в каждом тесте своя, т.е. в систему вносится всего один ошибочный код.

Формат выходных данных

Для каждого теста от 1 до K выведите на отдельной одной строке в алфавитном порядке (цифры считаются при сортировке "меньше" букв) разделенные одним пробелом все имена ветвей исходного кода, в которых присутствует критическая ошибка.

Примеры

standard input	standard output
5 commit master branch feature1 from master commit feature1 merge feature1 to master commit feature1 3 1 3 5	feature1 master feature1 master feature1

Задача H. Three Paths

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	1 second
Ограничение по памяти:	256 mebibytes

В Забытом Королевстве есть несколько городов, соединённых дорогами. По каждой дороге можно ездить в обе стороны, длины всех дорог неотрицательны. Поскольку жители Королевства очень не любят неоднозначность, система дорог удовлетворяет следующему свойству: для любых двух городов существует *единственный* кратчайший путь между ними.

В один прекрасный день все жители Королевства забыли схему дорог (т.е. они забыли количество городов, количество и длины дорог). К счастью, они всё ещё помнят описания кратчайших путей для трёх пар городов. Впрочем, они не совсем уверены, что помнят их правильно. Помогите им: по данным трём путям определите, существует ли схема дорог с соответствующими длинами, в которой эти пути являются уникальными кратчайшими путями.

Формат входных данных

Входные данные содержат три строки — описания трёх путей. Каждая строка начинается с целого числа n_i — количества городов на пути; оставшаяся часть строки содержит разделённые пробелами номера городов $a_{i,1}, \ldots, a_{i,n}$ в порядке следования по пути ($1 \leqslant n_i \leqslant 100\,000$; $1 \leqslant a_{i,j} \leqslant 10^9$). Одинаковые номера соответствуют одинаковым городам, разные номера — разным городам.

Формат выходных данных

Выведите одну строку, содержащую «Yes», если требуемая система дорог существует, и «No» иначе.

Примеры

standard input	standard output
5 4 5 6 7 8 5 1 5 9 13 17 5 3 7 10 13 9	Yes
5 10 20 30 40 50 5 10 21 32 40 53 5 29 30 31 32 33	No

Задача I. Flatland Holidays

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	1 second
Ограничение по памяти:	256 mebibytes

Анализируя выходные и праздники в районе Нового года и 1 мая в России, президент Флатландии пришел к выводу, что отдых своих граждан можно оптимизировать кардинально, главное, чтобы им не пришлось работать больше 6 дней подряд в одном календарном году. Он поручил Министерству труда разработать такой график переноса выходных дней (суббот и воскресений), чтобы с учетом имеющихся во Флатландии праздников граждане смогли отдыхать как можно больше дней подряд.

Напомним, что если государственный праздник Флатландии приходится на выходной день (субботу или воскресенье), то этот выходной автоматически переносится на первый после праздника рабочий день. Но по указу президента любой выходной, как совпавший с праздничным днем, так и не совпавший, может быть перенесен на любой рабочий день, праздники при этом никогда не переносятся.

Напишите программу, которая поможет Министерству труда составить требуемый график переноса выходных на очередной год. Праздники и выходные как предыдущего, так и следующего года при этом учитывать не нужно. Максимизировать требуется число дней отдыха подряд именно в одном году.

Формат входных данных

В первой строке входных данных находятся два числа — номер года Y ($2012 \le Y \le 2050$) и номер дня недели для 1 января этого года W ($1 \le W \le 7$) от понедельника до воскресенья соответственно. В этом диапазоне лет номера високосных годов делятся на 4.

Во второй строке дано число ежегодных праздников N ($0 \le N \le 366$) Флатландии. В каждой из следующих N строк записана дата очередного праздника в формате `DD.MM`. Праздничные дни перечислены в хронологическом порядке. Все даты различны и корректны относительно рассматриваемого года.

Формат выходных данных

Выдайте одно число — максимально возможное число дней непрерывного отдыха жителей Флатландии в указанном году, если выходные перенести так, что число подряд идущих рабочих дней в этом году не будет превышать 6.

Примеры

standard input	standard output
2012 7 1 01.01	63

Задача J. Cyclic Number

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Дано целое положительное число в десятичной записи. Рассмотрим числа, полученные из него циклическим сдвигом на 0, 1, ..., $N-1$ цифр, и каждое из этих N чисел умножим на его первую цифру. Требуется вывести сумму полученных произведений.

Формат входных данных

На вход подается одна строка, содержащая десятичную запись целого положительного числа, без ведущих нулей, длиной не более 250 000 цифр **(для первого дивизиона) и 25 000 цифр (для второго дивизиона).**

Формат выходных данных

Выведите одно число — ответ к задаче.

Примеры

standard input	standard output
22	88
123	1521

Задача K. Building (Division 2 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Строительная компания занимается постройкой здания, стены которого изготовлены заранее и ставятся на место с помощью подъёмных кранов. Строительная фирма нашла n площадок для установки кранов. Требуется выбрать минимальное число кранов так, чтобы середина каждой из стен была достижима как минимум одним краном. Стрела крана имеет длину r.

План здания является прямоугольником длины l и ширины w, стороны которого параллельны координатным осям, а центр совпадает с началом координат.

Формат входных данных

Первая строка входного файла содержит четыре целых положительных числа l, w, n и r, каждое из которых не превосходит 30. l и w задают длину и ширину здания, n задаёт количество площадок для установок кранов, а r задаёт длину стрелы крана.

В каждой из последующих n строках заданы по два целых числа x_i и y_i ($-100 \le x_i, y_i \le 100$), задающих координаты каждой площадки для установки крана. Начало координат находится в точке пересечения диагоналей плана здания, оси координат параллельны стенам, ось x параллельна стене длины l.

Формат выходных данных

Выведите одно целое число — минимальное количество кранов, достаточное, чтобы середины всех четырёх стен были достижимы, или "`Impossible`", если до какой-то из середин ни один кран не «дотягивается».

Пример

standard input	standard output
4 2 3 3 1 -2 4 0 -1 2	2

Задача L. Coffee or beer? (Division 2 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

В офисе компании, в которой Вы работаете, провели волшебный водопровод. Если тумблёр стоит в положении «вверх», из крана льётся кофе, если в положении «вниз» — пиво. Сотрудники разделились на две группы: одна предпочитает кофе, вторая — пиво.

Предлагается несколько вариантов правил пользования краном:

1. После использования автомата оставляйте тумблёр в позиции «вверх»;
2. После использования автомата оставляйте тумблёр в позиции «вниз»;
3. После использования автомата не трогайте тумблёр — кому надо, тот переключит.

Таким образом, сотрудник может переключать тумблёр перед использованием и, в зависимости от принятых правил, ему придётся переключать тумблёр после использования.

Вам задано начальное состояние тумблёра и предпочтения стоящих в очереди сотрудников. Вычислите, сколько раз тумблёр будет переключён в соответствии с каждым из трёх вариантов правил.

Формат входных данных

В первой и единственной строке входного файла содержится строка, состоящая из не менее 2 и не более 1000 символов `'U'` и `'D'`, задающих начальную позицию и пожелания очередного сотрудника установить тумблёр вверх и выпить кофе или установить тумблёр вниз и выпить пива.

Первый символ обозначает начальное положение тумблёра, последующие $n-1$ символов указывают предпочтения $n-1$ сотрудников в порядке очерёдности.

Формат выходных данных

Выведите три целых числа, каждое на отдельной строке — количество переключений тумблёра для первой, второй и третьей систем правил соответственно.

Пример

standard input	standard output
UUUDDUDU	6 7 4

Задача M. Decoding (Division 2 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Алиса и Боб придумали очередной шифр, кодирующий слова, набранные ЗАГЛАВНЫМИ буквами с помощью циклических сдвигов.

Циклический сдвиг буквы на S позиций обозначает, что буква заменяется на отстоящую от неё в алфавите на S мест вперёд (при этом за `Z` следует снова `A`). Например, сдвиг `B` на $S = 3$ позиции даёт `E`, а сдвиг `Z` на $S = 2$ позиции даёт `B`.

Шифр зависит от параметра K и состоит в следующем: для P-й буквы в сообщении используется сдвиг на $S = 3P + K$ позиций.

Напишите программу, выдающую по зашифрованному сообщению исходное.

Формат входных данных

Входной файл состоит из двух строк. Первая строка входного файла задаёт целое положительное число K ($K < 10$) — параметр шифра. Вторая строка содержит слово — непустую последовательность заглавных латинских букв, длина которой не превышает 20.

Формат выходных данных

Выведите декодированное сообщение.

Пример

standard input	standard output
3 FXAB	ZOOM