Задача A. Шифр Юлия

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Юлий Цезарь использовал свой способ шифрования текста. Каждая буква заменялась на следующую по алфавиту через K позиций по кругу. Необходимо по заданной шифровке определить исходный текст.

Формат входного файла

В первой строке дана шифровка, состоящая из заглавных латинских букв и не превышающая по длине 255 символов. Во второй строке задано число K ($1 \le K \le 10$).

Формат выходного файла

Требуется вывести результат расшифровки.

Примеры

stdin	stdout
XPSE 1	WORD

Задача B. Добыча золота

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

Андрей играет в следующую экономическую игру: дана шахта, добывающая k единиц золота за 1 ход. В начале игры количество золота у Андрея равно нулю. В начале каждого хода можно принять решение о постройке одной (или более) новых шахт. Постройка каждой шахты занимает T ходов и требует S единиц золота (золото вычитается у игрока в момент принятия решения о постройке новой шахты). Игра заканчивается через N ходов. Определите максимальное количество единиц золота, с которым Андрей может завершить игру.

Формат входного файла

В единственной строке через пробел вводятся числа k ($1 \le k \le 100$), T ($1 \le T \le 10$), S ($1 \le S \le 10000$) и N ($1 \le N \le 100$).

Формат выходного файла

В выходной файл выведите единственное число – максимальное количество единиц золота, которое можно получить за N ходов. Гарантируется, что результат не превышает 10^{18}.

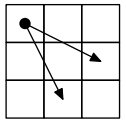
Примеры

gold.in	gold.out
100 5 50 6	600 (комментарий: в начале игры количество золота равно нулю, следовательно начать строительство можно только со второго хода; строительство будет длиться 5 ходов (ходы со второго по шестой), поэтому к концу 6 хода новые шахты только будут построены, но еще не добудут нового золота. Исходя из этого решение строить новые шахты будет неоптимальным. Решение строить новые шахты меньше чем за 5 ходов до конца игры также будет неоптимальным.)

Задача C. Ход конём

Имя входного файла:	`knight.in`
Имя выходного файла:	`knight.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Дана прямоугольная доска $N \times M$ (N строк и M столбцов). В левом верхнем углу находится шахматный конь, которого необходимо переместить в правый нижний угол доски. В данной задаче конь может перемещаться на две клетки вниз и одну клетку вправо или на одну клетку вниз и две клетки вправо.



Необходимо определить, сколько существует различных маршрутов, ведущих из левого верхнего в правый нижний угол.

Формат входного файла

Входной файл содержит два натуральных числа N и M ($1 \leqslant N, M \leqslant 50$).

Формат выходного файла

В выходной файл выведите единственное число — количество способов добраться конём до правого нижнего угла доски.

Примеры

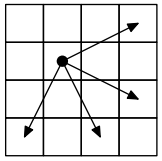
knight.in	knight.out
3 2	1
31 34	293930

Задача D. Ход конём - 2

Имя входного файла: knight2.in
Имя выходного файла: knight2.out
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Дана прямоугольная доска $N \times M$ (N строк и M столбцов). В левом верхнем углу находится шахматный конь, которого необходимо переместить в правый нижний угол доски.

При этом конь может ходить следующим образом:



Необходимо определить, сколько существует различных маршрутов, ведущих из левого верхнего в правый нижний угол.

Формат входного файла

Входной файл содержит два натуральных числа N и M ($1 \leqslant N, M \leqslant 50$).

Формат выходного файла

В выходной файл выведите единственное число — количество способов добраться конём до правого нижнего угла доски.

Примеры

knight2.in	knight2.out
4 4	2
2 3	1

Задача E. Объединение последовательностей

Имя входного файла:
Имя выходного файла: merge.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 1 мегабайт

Дан следующий код:

```
#include <fstream>
#include <cstdlib>
#include <cstdio>

using namespace std;

void solve(char *filename1, char *filename2);

int main(int argc, char* argv[]) {
        solve(argv[1], argv[2]);
        return 0;
}
```

Реализуйте функцию слияния двух текстовых файлов. Заголовок функции должен выглядеть следующим образом:

```
void  solve(char *filename1, char *filename2);
```

При вызове функции параметр $filename1$ будет содержать полный путь к первому файлу, параметр $filename2$ — полный путь ко второму файлу. Гарантируется, что файлы не пусты.

Известно, что каждый файл содержит целые числа (по модулю не превышающие 10^9), по одному в строке (пустых строк в файле нет). Признаком конца каждого из файлов является число 0 (гарантируется, что в каждом из файлов оно встречается только один раз, в последней строке). Число 0 не относится к последовательности чисел, а используется только как признак окончания файла, при этом оно может быть единственным числом в файле. Однако гарантируется, что хотя бы один файл содержит хотя бы одно число, отличное от нуля. Количество чисел в каждом файле неизвестно и может быть сколь угодно большим, числа упорядочены по неубыванию (каждое следующее больше либо равно предыдущему). Функция должна создать файл с именем merge.out, и записать в него все числа, содержащиеся в исходных файлах (за исключением конечных нулей, их выводить в новый файл не нужно). При этом в файле merge.out числа также должны быть расположены по неубыванию.

Для проверки этой задачи тестирующей системой отправьте только функцию (без include, main() и т.д.).

Функция может использовать любые локальные переменные (при условии, что выполняется ограничение по памяти). Никакие глобальные переменные использовать нельзя.