Общее описание задач

№	Название задачи	Ограничение на объем используемой памяти	Ограничение на время выполнения программы, секунды	Количество баллов, присвоенных задаче
1.	Взлом кода	≤ 4Mb	≤ 0,2	100
2.	Робот	≤ 1Mb	≤ 0,1	100
3.	Пароли	≤ 6Mb	≤ 0,1	100

Примечание. При равенстве общего количество очков, приоритет будет отдан участнику, набравшему первым соответствующее количество очков.

Взлом кода

Факторизацией числа **N** называется его представление в виде произведения простых чисел. Например, число 15 можно представить в виде произведения 3 и 5. Очень просто проверить, что 3 и 5 - простые числа, и их произведение равно 15. Однако обратная операция, т.е. факторизация числа, является сложной задачей. Современные криптосистемы основаны на сложности факторизации больших чисел.

Задание. Разработайте программу, которая сможет взламывать такие системы. Вам дано число **N**, которое является произведением двух простых чисел. Найдите эти числа. Если таких чисел не существует, выведите `failed`.

Входные данные. Первая строка стандартного ввода будет содержать натуральное число **N**.

Выходные данные. На первой строке стандартного вывода выведите два простых числа через пробел, упорядоченные в порядке возрастания, которые при произведении дают **N**. Если таких чисел не существует, вывести `failed`.

Ограничения$1 <= N <= 10^{12}$. Ограничения по времени выполнения и объему используемой памяти приведены в общем описании задач, предлагаемых к решению. Исходный файл должен называться `spargere.pas, spargere.c` или `spargere.cpp`.

Пример

Ввод	**Вывод**
15	3 5
10	2 5
7	failed

Робот

Группа исследователей изучает далекие планеты, и чтобы собрать образцы почвы с этих планет, им приходится использовать робота с дистанционным управлением. Прежде чем использовать робота, исследователи с помощью спутника снимают изображение изучаемой планеты. Представление поверхности планеты описывается с помощью двумерного массива из n строк и m столбцов, состоящих из нулей (0) и единиц (1). Где 0 означает области, где робот может двигаться, а области, отмеченные 1, представляют собой области, где робот не может двигаться.

Робот вынужден пересечь планету из левого верхнего угла в правый нижний угол по кратчайшему возможному пути. Известно, что робот может двигаться в следующих направлениях: Север и Юг (Вертикально), Восток и Запад (Горизонтально), Северо-Запад, Северо-Восток, Юго-Запад и Юго-Восток (По Диагонали). Известно, что переход от одной координаты к другой считается сделанным шагом. Приземление на стартовую координату также считается шагом.

Задача. Разработайте программу, определяющую минимальное количество шагов, необходимых для передвижения от левого верхнего угла до правого нижнего угла в двумерном массиве, следуя описанным выше правилам.

Входные данные. Стандартный ввод в первой строке содержит два целых числа, разделённые пробелом `n` и `m`, где `n` — количество строк, а `m` — количество столбцов. Далее в n строках следуют m целых чисел (0 или 1), образующих карту данной планеты.

Выходные данные. Стандартный вывод будет содержать целое число - минимальное количество шагов, необходимое для пересечения планеты. Если построить маршрут невозможно, программа выдаст `-1`.

Ограничение. $1 \le n \le 100$, $1 \le m \le 100$. Ограничения по времени выполнения и объему используемой памяти приведены в общем описании задач, предлагаемых к решению. Исходный файл будет иметь имя *robotul*.pas, *robotul*.c или *robotul*.cpp.

Примеры.

Ввод

```
4 4
0 1 0 1
1 0 0 1
1 0 1 1
0 0 0 0
```

Вывод

```
5
```

Объяснение. Чтобы добраться из левого верхнего угла в правый нижний угол, необходимо сделать 5 шагов. Первый шаг - приземление, второй шаг - движение на Юго-Восток (по диагонали), третий шаг - движение на Юг, четвертый шаг - движение на Юго-Восток (по диагонали) и последний шаг – движение на Восток.

Ввод

```
2 3
1 0 1
0 1 1
```

Вывод

```
-1
```

Объяснение. Дойти до конечной точки невозможно.

Пароли

На секретном предприятии генерация пароля очень важная вещь, которая проходит ряд проверок. В этом смысле внутри данной компании существует особая политика проверки паролей. Пароль может состоять только из букв: `a, e, i, o, u` (все строчные). И есть следующие правила, по которым можно сформировать пароль.

- После буквы 'a' может следовать только буква 'e';
- После буквы 'e' может следовать только буква 'a' или 'i';
- После буквы 'i' не может следовать другая буква 'i';
- После буквы 'o' может следовать только буква 'i' или 'u';
- После буквы 'y' может следовать только буква 'a'.

Таким образом, требуется выяснить, сколько уникальных паролей длины N можно составить, следуя описанным правилам.

Задача. Разработайте программу, определяющую количество уникальных паролей, которые можно составить по описанным правилам. Поскольку число может быть очень большим, выведите его в уменьшенном виде после применения операции $\bmod 10^9 + 7$.

Входные данные. Стандартный ввод содержит целое число *N*, которое представляет количество букв в пароле.

Выходные данные. Стандартный вывод будет содержать целое число (уменьшенный через $\bmod 10^9 + 7$), представляющее, сколько уникальных паролей можно сгенерировать **.**

Ограничения. $1 \le N \le 10^4$. Ограничения по времени выполнения и объему используемой памяти приведены в общем описании задач, предлагаемых к решению. Исходный файл будет называться *parole*.pas, *parole*.c или *parole*.cpp.

Примеры.

Ввод	**Вывод**
2	10

Объяснение. Можно сгенерировать 10 уникальных паролей. Это: ae, ea, ei, ia, ie, io, iu, oi, ou, ua.