# Задача A. Задача коммивояжёра

| | |
|---|---|
| Имя входного файла: | *стандартный ввод* |
| Имя выходного файла: | *стандартный вывод* |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 512 мебибайт |

Вам предлагается решить вариацию классической задачи, называемой *задачей коммивояжёра*.

Известен список городов, которые требуется посетить. Города представлены точками на плоскости. Расстояние между городами определяется как обычное евклидово расстояние между точками. Требуется найти как можно более короткий маршрут, проходящий по всем городам. Заметьте, что возвращаться назад не нужно.

Задачу не обязательно решать точно. Но чем оптимальнее вы её решите, тем больше баллов получите.

## Формат входных данных

В первой строке ввода записано целое число $N$ — количество городов в маршрутном листе коммивояжёра ($2 \leqslant N \leqslant 5000$). Города не совпадают.

В следующих $N$ строках записано по два целых числа — координаты городов. Все координаты не превышают $10^9$ по абсолютному значению.

## Формат выходных данных

Выведите $N$ различных целых чисел от 1 до $N$ — номера городов в порядке обхода.

## Примеры

| стандартный ввод | стандартный вывод |
|---|---|
| 5<br>1 0<br>0 0<br>1 2<br>0 2<br>2 1 | 2 1 5 3 4 |

## Система оценки

Тесты, в которых $2 \leqslant N \leqslant 18$, оцениваются суммарно из 25 баллов.

Тесты, в которых $30 \leqslant N \leqslant 100$, оцениваются суммарно из 25 баллов.

Тесты, в которых $1000 \leqslant N \leqslant 5000$, оцениваются суммарно из 50 баллов.

Каждый из тестов будет генерироваться одним из следующих двух способов:

- Генерируется случайное $R$ от $10^6$ до $10^9$. Каждая точка генерируется случайным образом внутри квадрата $-R \leqslant x, y \leqslant R$.
- Генерируются случайные $R_1$ и $R_2$ от $10^6$ до $10^9$. Каждая точка генерируется случайным образом внутри кольца $\min(R_1, R_2)^2 \leqslant x^2 + y^2 \leqslant \max(R_1, R_2)^2$.

В рамках каждой из трех групп тестов, определённых выше, будет одинаковое количество тестов каждого из этих двух типов.

Баллы за каждый тест будут вычислены по следующей формуле:

$$SCORE \cdot 10^{-\left(\frac{Len}{\min(Len, Best)} - 1\right)}$$

Здесь:

- $SCORE$ — максимальное количество баллов за тест,
- $Best$ — длина пути, найденного решением жюри,
- $Len$ — длина пути, предоставленного вами.

## Задача B. Путь и сливы

| | |
|---|---|
| Имя входного файла: | *стандартный ввод* |
| Имя выходного файла: | *стандартный вывод* |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 512 мебибайт |

Котёнок отправляется в дальний путь. Начиная из своего родного города, он хочет пройти как можно больше городов. Для того, чтобы справиться с такой задачей, ему нужна постоянная поддержка в виде слив.

В путь котёнок взял с собой $k$ слив, каждой из которых хватает ровно на километр (конечно, котёнок может пройти и меньшее расстояние). При переходе из одного города в другой котёнок не обязан съедать целое количество слив. Определите, какое наибольшее количество различных городов он сможет обойти.

### Формат входных данных

В первой строке ввода записаны числа $n$ и $k$ — количество городов и количество слив у котёнка. В следующих $n$ строках записано $n$ различных пар целых чисел $x_i$, $y_i$ — координаты городов, заданные в километрах.

Ограничения: $1 \leqslant n \leqslant 100$, $0 \leqslant x_i, y_i \leqslant 100$, $0 \leqslant k \leqslant 10^5$.

Котёнок начинает путь из города номер 1. Города нумеруются начиная с единицы.

### Формат выходных данных

В первой строке выведите число городов в наилучшем найденном пути. Во второй строке выведите их номера в порядке обхода.

### Пример

| *стандартный ввод* | *стандартный вывод* |
|---|---|
| 3 1<br>0 0<br>1 1<br>1 0 | 2<br>1 3 |

### Система оценки

Баллы за тест будут равны $2^{V_p - V_m} \cdot X$, где $V_p$ — количество пройденных городов в вашем ответе, $V_m$ — максимальное значение корректного ответа на тест среди всех отправленных программ всех участников и программ жюри, а $X$ — номинальная стоимость теста.

Тесты, в которых ваше решение не выдаст корректный ответ, будут оценены нулём.

В тестах суммарной стоимостью 30 количество городов не будет превышать 20.

## Задача C. Студенты

| | |
|---|---|
| Имя входного файла: | *стандартный ввод* |
| Имя выходного файла: | *стандартный вывод* |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 512 мебибайт |

На Всероссийские сборы студентов по программированию приехали $k$ студентов. Руководство сборов планирует прочитать студентам несколько лекций по $n$ различным темам так, чтобы после этого все студенты были знакомы со всеми темами.

Однако какие-то студенты уже знакомы с некоторыми из $n$ тем, представленных в теоретическом курсе. Доподлинно известно, что, если какому-то студенту за время сборов прочитать лекции на более чем одну тему, с которой он и так знаком, то этот студент засыпает прямо на лекции, и в дальнейшем его эффективность падает. Руководство сборов хочет избежать таких нежелательных последствий.

Решено было разбить студентов на группы, и каждой группе читать какое-то одно подмножество тем. Однако лекторов мало, поэтому нужно придумать такое разбиение, которое бы минимизировало количество различных групп.

### Формат входных данных

В первой строке входного файла записаны два целых числа $n$ и $k$ через пробел ($1 \leqslant n \leqslant 6$, $0 \leqslant k \leqslant 100$). Следующие $k$ строк содержат описания студентов; $i$-ая из этих строк содержит $n$ чисел $a_{i,1}$, $a_{i,2}$, …, $a_{i,n}$ через пробел. Число $a_{i,j}$ равно 1, если студент $i$ уже знаком с темой $j$, и 0 в противном случае.

### Формат выходных данных

В первой строке выходного файла выведите число $p$ — минимальное количество групп, на которое удастся разбить всех студентов. В следующих $p$ строках выведите сами группы, по одной на строке, в формате $q_i$ $d_{i,1}$ $d_{i,2}$ … $d_{i,q_i}$, где $q_i$ — количество студентов в группе, а $d_{i,j}$ — номера студентов в этой группе. Студенты нумеруются с единицы в том же порядке, в каком они заданы во входном файле. Каждый из $k$ студентов должен оказаться ровно в одной группе. Порядок вывода групп и порядок вывода номеров студентов в группе не имеет значения. Если оптимальных ответов несколько, можно вывести любой из них.

### Пример

| стандартный ввод | стандартный вывод |
|---|---|
| 5 3<br>0 0 1 0 1<br>1 1 0 1 1<br>1 1 0 0 1 | 2<br>1 1<br>2 2 3 |

### Пояснение к примеру

В этом примере первой группе читаются темы 1, 2 и 4, а второй группе — темы 3 и 4. Объединить всех студентов в одну группу не получится, поскольку тогда им необходимо рассказать все темы, кроме пятой, и студенты 2 и 3 заснут, когда прослушают лекции по темам 1 и 2.

## Задача D. Универсальный путь

| | |
|---|---|
| Имя входного файла: | *стандартный ввод* |
| Имя выходного файла: | *стандартный вывод* |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 512 мебибайт |

Рассмотрим лабиринты размера $10 \times 10$ квадратных клеток. Каждая клетка либо пуста, либо заполнена непроходимой стеной. Одна из клеток лабиринта выбрана в качестве стартовой позиции, другая — в качестве пункта назначения.

После этого в стартовую позицию ставится Робот. Робот может делать шаги; один шаг перемещает Робота в соседнюю клетку в одном из четырёх основных направлений. Робот не может делать шаг в стену, а также покидать пределы лабиринта.

У Робота есть программа, которую он должен выполнить. Программа — это строка, состоящая из следующих команд: сделать один шаг на север («`N`»), на запад («`W`»), на юг («`S`») или на восток («`E`»). Команды выполняются последовательно в том порядке, в котором они записаны в программе. Для каждой команды, если Робот может сделать шаг из своей текущей позиции в заданном направлении, он делает этот шаг. В противном случае (если в этом направлении находится стена, или же этот шаг выводит Робота за пределы лабиринта) Робот просто игнорирует команду.

Задача Робота — пройти лабиринт, то есть оказаться в пункте назначения. Если в какой-то момент Робот перемещается в пункт назначения, его задача выполнена. Если Робот выполнил все команды в своей программе, но ни разу не побывал в пункте назначения, его задача провалена.

Количественная мера достижений Робота — штраф, определяемый следующим образом. Если задача выполнена, штраф — это количество команд, выполненных перед тем, как Робот в первый раз побывал в пункте назначения (здесь считаются как команды, в результате которых был сделан шаг, так и команды, которые были проигнорированы). Если же задача Робота провалена, штраф равен константе 20 000.

**Ваша задача** — написать такую программу для Робота, чтобы у него хорошо получалось проходить лабиринты *в среднем*, то есть чтобы средний штраф не превысил заданный предел $t$. В вашей программе должно быть не более 10 000 команд.

Ваша программа будет проверяться следующим образом. Всего жюри заготовило 20 тестов. В каждом тесте на вход вашему решению подаётся лишь целое число $t$. У жюри есть также 1000 заранее построенных лабиринтов для каждого теста (всего 20 000 лабиринтов), но они держатся в секрете. После того, как ваше решение выведет программу для Робота, на каждом тесте проверяющая программа запускает её на 1000 заранее построенных лабиринтов. Если среднее значение штрафа на этих лабиринтах не превысило заданный предел, ваше решение проходит этот тест; в противном случае оно получает вердикт «`Wrong Answer`» на этом тесте.

Все заранее построенные лабиринты сгенерированы следующим алгоритмом. Сначала на поле $10 \times 10$ случайным образом выбирается 20 различных клеток. После этого следует 500 итераций добавления стен. На каждой итерации выбирается случайная клетка на поле $10 \times 10$. Если эта клетка свободна и не отмечена, на неё ставится стена. После этого, если все отмеченные клетки всё ещё достижимы друг из друга, стена остаётся на клетке; в противном случае она убирается, и клетка опять становится свободной.

После того, как стены установлены, выбираются стартовая позиция и пункт назначения. Для каждой пары достижимых друг из друга клеток лабиринта расстоянием будем считать длину кратчайшего пути между ними; здесь длина пути — это количество шагов в нём. Далее случайным образом выбирается одна из тех пар клеток, для которых расстояние получилось максимальным. Одна из клеток пары становится стартовой позицией, а другая — пунктом назначения.

В описанном выше алгоритме все случаи случайного выбора независимы и равномерно распределены. Все лабиринты генерируются случайно и независимо. Заметьте, что вашему решению не требуется хорошо работать на лабиринтах, отличных от тех, которые строятся по этому алгоритму.

### Формат входных данных

На первой строке ввода задано целое число $t$ — предел среднего штрафа ($1000 \leqslant t \leqslant 1500$). Всего в этой задаче 20 тестов. Гарантируется, что в тесте с номером $n$ предел равен $t = \max(1000, 1550 - n \cdot 50)$. Так, предел равен 1500 в тесте 1, 1450 в тесте 2, 1400 в тесте 3, …, 1050 в тесте 10 и 1000 в тестах 11–20. Первые 10 тестов могут показать, насколько хорошо работает ваша программа. Последние 10 тестов предназначены для проверки того, что ваша программа действительно хорошо работает.

Заметьте, что ваше решение не обязано использовать число, заданное во вводе, это число дано лишь для удобства.

### Формат выходных данных

Ваше решение должно выводить программу для Робота на отдельной стро-

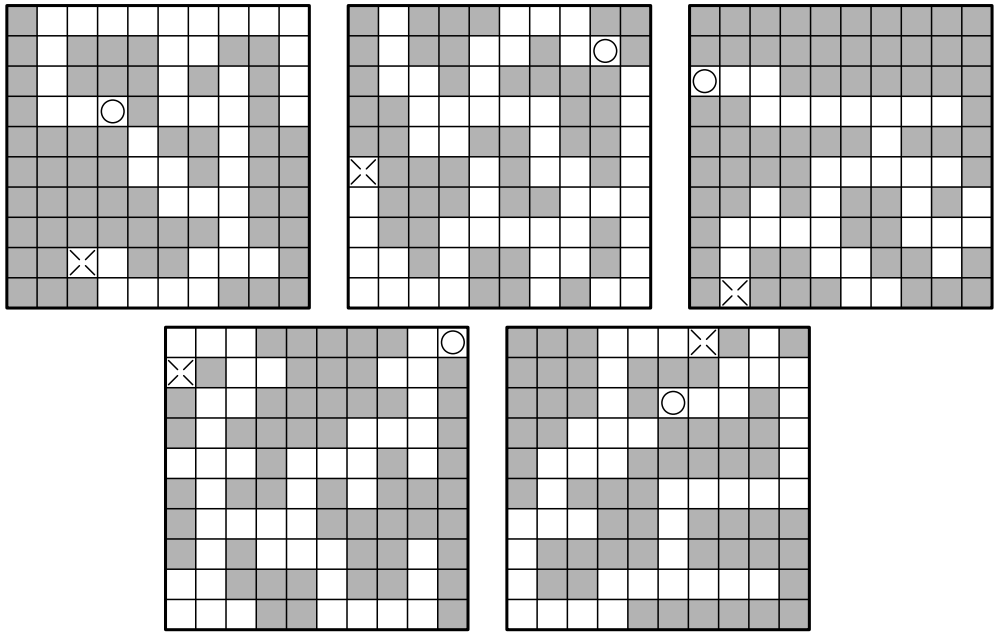
ке. Её длина должна быть не более 10 000 символов. Каждый символ должен быть либо «N» для движения на север, либо «W» для движения на запад, либо «S» для движения на юг, либо «E» для движения на восток.

Помните, что, если вашему решению требуется значительное время для нахождения хорошего ответа, вы можете найти этот ответ на своём локальном компьютере и послать лишь программу, которая его выводит.

## Пример

В этой задаче нет примеров тестов.

В качестве иллюстрации к работе алгоритма, генерирующего лабиринты, ниже приводится несколько сгенерированных им лабиринтов.



## Задача E. Игра с цифрами

| | |
|---|---|
| Имя входного файла: | *стандартный ввод* |
| Имя выходного файла: | *стандартный вывод* |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 512 мебибайт |

Витя играет в следующую игру с цифрами. По окружности расставлено $k$ десятичных цифр. Одним ходом Витя выбирает две соседних цифры и получает баллы, равные их произведению. Затем Витя удаляет эти две цифры с окружности, а на их месте появляются две новые цифры. Цель Вити — набрать как можно большую сумму баллов за $m$ ходов.

Валя помогает Вите играть в игру. Она использует генератор псевдослучайных чисел, чтобы предоставить Вите первоначальные $k$ цифр, а также $2m$ цифр, которые появляются после каждого из $m$ ходов на месте удалённых.

Валя решила заготовить и показать Вите все требуемые цифры заранее. Помогите ребятам сыграть в игру так, чтобы набрать как можно больше очков.

В этой задаче во всех тестах $k = 10$. Кроме того, во всех тестах, кроме примера, $m = 5000$, а в примере $m = 5$.

### Формат входных данных

В первой строке заданы $k+2m$ цифр без пробелов — цифры, заготовленные Валей. Первые $k$ из этих цифр изначально расставляются по окружности по часовой стрелке. Каждая следующая пара цифр ставится на место удалённой пары в порядке обхода по часовой стрелке. Это означает, что направление от первого ко второму поставленному числу соответствует движению по окружности по часовой стрелке.

Гарантируется, что для получения цифр использовался генератор псевдослучайных чисел, который создал $k+2m$ целых чисел от 0 до 9 включительно, равномерно распределённых в этом интеравле и независимых между собой.

### Формат выходных данных

Выведите $m$ пар цифр без пробелов, по одной паре в строке. Каждая пара цифр $(a_i, b_i)$ должна соответствовать двум позициям на окружности в порядке обхода по часовой стрелке, то есть должно быть верно $b_i = (a_i + 1) \bmod k$. Позиции нумеруются от 0 до 9 в том порядке, в котором на них расставляются изначальные цифры. После этого выведите в отдельной строке одно число — суммарное количество очков.

### Пример

| стандартный ввод | стандартный вывод | пояснение |
|---|---|---|
| 30137634348786663999 | 56 | 3013763434 $+6 \cdot 3$ |
| | 90 | 3013787434 $+4 \cdot 3$ |
| | 45 | 6013787438 $+7 \cdot 8$ |
| | 89 | 6013667438 $+3 \cdot 8$ |
| | 34 | 6013667439 $+3 \cdot 6$ |
| | 128 | 6019967439 |

### Система оценки

В этой задаче один пример, за который очки не начисляются, и 20 тестов, за каждый из которых можно получить до 5 очков. Количество очков за тест равно $5 \cdot \frac{current}{best}$, где $current$ — это суммарные баллы участника на этом тесте, а $best$ — наилучший результат всех участников и жюри на этом тесте за всё время тестирования.

Пример ответа в условии может не быть оптимальным.

## Задача F. Цифры в таблице

| | |
|---|---|
| Имя входного файла: | *стандартный ввод* |
| Имя выходного файла: | *стандартный вывод* |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 512 мебибайт |

У Антона есть квадратная таблица из $n \times n$ клеток. В каждой клетке находится цифра от 1 до 9. Соседними считаются клетки, имеющие общую сторону или угол.

Назовём множество клеток *хорошим*, если в нём никакие две клетки не являются соседними. Помогите Антону выбрать хорошее множество так, чтобы сумма цифр в нём была как можно больше.

### Формат входных данных

В первой строке задано целое число $n$ ($5 \leqslant n \leqslant 50$). Следующие $n$ строк содержат по $n$ цифр от 1 до 9 без пробелов и описывают таблицу.

Гарантируется, что для получения цифр использовался генератор псевдослучайных чисел, который выдал $n^2$ целых чисел от 1 до 9 включительно, равномерно распределённых в этом интеравле и независимых между собой.

### Формат выходных данных

В первой строке выведите сумму выбранных цифр. В следующих $n$ строках выведите по $n$ цифр — состояние таблицы. Выбранные клетки должны содержать исходную цифру, а не выбранные — цифру 0.

### Пример

| стандартный ввод | стандартный вывод |
|---|---|
| 5<br>34913<br>99196<br>31267<br>17294<br>84196 | 44<br>00900<br>90000<br>00207<br>00000<br>80090 |

### Система оценки

В этой задаче один пример, за который очки не начисляются, и 20 тестов, за каждый из которых можно получить до 5 очков. Количество очков за тест равно $5 \cdot (\frac{current}{best})^3$, где $current$ — это суммарные баллы участника на этом тесте, а $best$ — наилучший результат всех участников и жюри на этом тесте за всё время тестирования.

Пример ответа в условии может не быть оптимальным.