

## Задача А. ДКА, проверяющий делимость

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

В условиях данной задачи разрешим в двоичной записи чисел присутствие ведущих нулей. Кроме того разрешим записывать число 0 в двоичной записи как  $\epsilon$  (как пустую строку).

Таким образом, строки «101» и «000101» являются двоичными записями числа 5, а строки «000000» и «» — двоичными записями числа 0.

Дано множество положительных нечетных чисел  $S$ .

Постройте ДКА над алфавитом  $\{0, 1\}$ , принимающий те и только те строки, которые являются двоичными записями чисел, делящихся на хотя бы одно число из множества  $S$ .

Число состояний в построенном автомате не обязано быть минимальным, но не должно превышать 20 000. Гарантируется, что существует искомый ДКА с не более чем 1 000 состояний.

### Формат входного файла

В первой строке входного файла содержится число  $|S|$  ( $1 \leq |S| \leq 1000$ ) — мощность множества  $S$ .

Затем следуют  $|S|$  различных положительных нечетных чисел, не превышающих  $10^9$ .

### Формат выходного файла

Выведите описание искомого автомата в следующем формате:

Первая строка содержит число  $|Q|$  — количество состояний автомата ( $1 \leq |Q| \leq 20000$ ). (Состояния нумеруются числами от 1 до  $|Q|$ .)

Следующая строка содержит число  $q_0$  ( $1 \leq q_0 \leq |Q|$ ) — номер начального состояния, затем число  $|T|$  — количество терминальных состояний, затем  $|T|$  чисел от 1 до  $|Q|$  — номера терминальных состояний.

Следующие  $|Q|$  строк содержат по  $|\delta|$  чисел — описание функции переходов  $\delta$ . Для каждого состояния в отдельной строке приводятся номера состояний, в которые из него ведут переходы по символам 0 и 1.

### Пример

| стандартный ввод | стандартный вывод     |
|------------------|-----------------------|
| 3                | 15                    |
| 3 5 15           | 15 7 3 5 6 9 10 12 15 |
|                  | 2 3                   |
|                  | 4 5                   |
|                  | 6 7                   |
|                  | 8 9                   |
|                  | 10 11                 |
|                  | 12 13                 |
|                  | 14 15                 |
|                  | 1 2                   |
|                  | 3 4                   |
|                  | 5 6                   |
|                  | 7 8                   |
|                  | 9 10                  |
|                  | 11 12                 |
|                  | 13 14                 |
|                  | 15 1                  |

## Задача В. Укладка плиток

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 256 мегабайт      |

Фирма «Автоматическая смена пола» разработала робота, который замощает плиткой дорожки. У робота есть неограниченный запас плиток размером  $2 \times 1$ . Дорожка, которую надо замостить, имеет размеры  $m \times n$ . Здесь  $m$  — ширина дорожки,  $n$  — ее длина. Дорожка идет с запада на восток.

Программа для робота — это последовательность команд «Н», «V» и «S». Робот начинает выполнять ее, стоя на северо-западной клетке дорожки.

Получив команду «Н», робот кладет очередную плитку горизонтально — длинной стороной с запада на восток, так что ее западная половина совпадает с клеткой, на которой робот стоит в данный момент.

Получив команду «V», робот кладет очередную плитку вертикально — длинной стороной с севера на юг, так что ее северная половина совпадает с клеткой, на которой робот стоит в данный момент.

Получив же команду «S», робот не кладет плитку.

После обработки команды робот передвигается на одну клетку на юг. Если он выходит за пределы дорожки, то он сдвигается на одну клетку на восток, и идет в самую северную клетку нового столбца.

Дорожка считается правильно замощенной, если каждая ее клетка покрыта ровно одной плиткой, и ни одна плитка не выходит за границы дорожки.

Программа называется правильной для конкретного  $m$ , если существует такое  $n$ , что робот, выполняющий эту программу на дорожке  $m \times n$ , правильно ее замостит и (выполнив последнюю команду) сойдет с юго-восточной клетки дорожки.

Например, программа «ННSS» корректна для  $m = 2$ , поскольку робот, выполняющий ее на дорожке  $2 \times 2$ , замостит дорожку правильно и покинет юго-восточную клетку. В то же время программа «ННV» не корректна для  $m = 2$  по двум причинам: во-первых, первые две плитки пересекаются с третьей, а во-вторых, робот не покидает дорожку (ни при каком  $n$ ).

Теперь главного программиста фирмы попросили написать программу, которая верифицирует правильность программ для этого робота. Но главный программист недавно прослушал лекцию по конечным автоматам, и решил составить автомат, принимающий те и только те строчки, которые являются корректными программами для конкретного  $m$ .

Однако главный программист уехал кататься на лыжах, и этот проект поручили вам. Теперь вам по данному  $m$  нужно построить конечный автомат, распознающий корректные (для этого  $m$ ) программы.

Поскольку проект коммерческий, автомат должен быть детерминированным. Число состояний в построенном автомате не обязано быть минимальным, но не должно превышать 20 000.

### Формат входного файла

Во входном файле содержится целое число  $m$  ( $1 \leq m \leq 10$ ).

### Формат выходного файла

Выведите описание искомого автомата в следующем формате:

Первая строка содержит число  $|Q|$  — количество состояний автомата ( $1 \leq |Q| \leq 20\,000$ ). (Состояния нумеруются числами от 1 до  $|Q|$ .)

Следующая строка содержит число  $q_0$  ( $1 \leq q_0 \leq |Q|$ ) — номер начального состояния, затем число  $|T|$  — количество терминальных состояний, затем  $|T|$  чисел от 1 до  $|Q|$  — номера терминальных состояний.

Следующие  $|Q|$  строк содержат по  $|\delta|$  чисел — описание функции переходов  $\delta$ . Для каждого состояния в отдельной строке приводятся номера состояний, в которые из него ведут переходы по символам «Н», «V» и «S».

## Пример

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

## Задача С. Непересекающиеся регулярные выражения

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

Павел разрабатывает новую социальную сеть ВКосмосе для обитателей спутников Марса Фобоса и Деймоса. Недавно он решил добавить на страницы информацию о том, на каком спутнике живет владелец страницы. Конечно, можно было бы спросить соответствующую информацию о пользователях у них самих, но он решил автоматизировать процесс. Для этого он изучил, как устроены имена обитателей спутников.

Имя каждого пользователя ВКосмосе представляет собой непустую строку, состоящую из строчных букв латинского алфавита. У пользователей с Фобоса имена подходят под регулярное выражение  $P$ , а у пользователей с Деймоса имена подходят под регулярное выражение  $D$ .

Однако Павел задумался над таким вопросом: а вдруг у какого-нибудь пользователя имя подходит под оба регулярных выражения. Два таких выражения будем называть *непересекающимися*, если никакая непустая строка  $s$  не подходит одновременно под оба выражения.

Помогите Павлу определить, являются ли заданные регулярные выражения непересекающимися. Если они не являются непересекающимися, требуется найти кратчайшую непустую строку  $s$ , которая подходит под оба выражения.

### Пояснение

- Одна буква  $s$  представляет собой корректное регулярное выражение. Под него подходит единственная строка, состоящая из одной буквы  $s$ .
- Операция выбора: если  $P$  и  $Q$  представляют собой регулярные выражения, то  $(P|Q)$  — регулярное выражение, под которое подходят все строки  $\alpha$ , которые подходят под  $P$  или под  $Q$ .
- Конкатенация: если  $P$  и  $Q$  представляют собой регулярные выражения, то  $(PQ)$  представляет собой регулярное выражение, под которое подходят строки  $\alpha$ , которые можно представить в виде  $\alpha = \beta\gamma$ , где  $\beta$  подходит под  $P$ , а  $\gamma$  подходит под  $Q$ .
- Звездочка Клини: если  $P$  представляет собой регулярное выражение, то  $(P^*)$  представляет собой регулярное выражение, под которое подходят строки  $\alpha$ , которые можно представить в виде конкатенации нуля или более строк  $\alpha_1\alpha_2\dots\alpha_k$ , где каждая из  $\alpha_i$  подходит под  $P$ . В частности, пустая строка всегда подходит под звездочку Клини любого выражения.

Можно опускать скобки, в этом случае звездочка Клини имеет максимальный приоритет, затем конкатенация и затем выбор. Например, “ $abc^*|de$ ” означает “ $(ab(c^*))|(de)$ ”.

### Формат входного файла

Вход содержит две строки. Первая строка содержит регулярное выражение  $P$ . Вторая строка содержит регулярное выражение  $D$ . Длина каждого регулярного выражения от 1 до 100 символов.

### Формат выходного файла

Если выражения являются непересекающимися, выведите “Correct”. В противном случае выведите “Wrong” на первой строке, а на второй строке выведите кратчайшую строку, которая подходит под оба выражения. Если таких строк несколько, выведите любую.

## Пример

| стандартный ввод     | стандартный вывод |
|----------------------|-------------------|
| a(ab)*b<br>a(a b)*ab | Correct           |
| a(ab)*a<br>a(a b)*ba | Wrong<br>aaba     |