Задача A. Подмножества

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

Рассмотрим два множества целых чисел: $X = \{x_1, x_2, \ldots, x_m\}$ и $Y = \{y_1, y_2, \ldots, y_n\}$. Пусть $x_1 < x_2 < \ldots < x_m$ и $y_1 < y_2 < \ldots < y_n$.

Будем говорить, что множество X лексикографически меньше множества Y, если выполнено одно из двух:

1. для некоторого i выполнено $x_1 = y_1$, $x_2 = y_2$, $\ldots$, $x_{i-1} = y_{i-1}$, $x_i < y_i$;
2. $m < n$ и $x_1 = y_1$, $x_2 = y_2$, $\ldots$, $x_m = y_m$.

Требуется вывести все подмножества множества $\{1, 2, \ldots, n\}$ в лексикографическом порядке.

Формат входного файла

Входной файл содержит одно число n $(1 \le n \le 16)$.

Формат выходного файла

Выведите в выходной файл все подмножества множества $\{1, 2, \ldots, n\}$ в лексикографическом порядке.

Пример

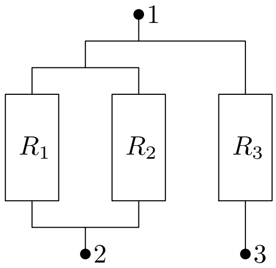
<table>
<tr><th>subsets.in</th><th>subsets.out</th></tr>
<tr><td><pre>3</pre></td><td><pre>
1
1 2
1 2 3
1 3
2
2 3
3</pre></td></tr>
</table>

Задача B. Схема

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

Недавно Вася стал увлекаться электричеством. Нет, он, конечно, не балуется и не пытается проверить наличие электрического тока засовыванием пальцев в розетку. Вася спаял схему, состоящую из N контактов, соединенных M резисторами. Некоторые два контакта A и B могут быть не соединены ни одним резистором, могут быть соединены одним резистором, а могут быть соединены несколькими параллельно подключенными резисторами. В последнем случае эти резисторы по описанным ниже правилам могут быть мысленно заменены одним.

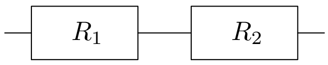
Изучив спаянную схему, Вася понял, что она обладает весьма интересным свойством — в ней существует путь из резисторов между любыми двумя контактами. Более того, если каждую из групп параллельно подключенных резисторов мысленно заменить на один резистор, то между любыми двумя контактами будет ровно один путь.



Пример схемы: к контакту 1 *подключается «плюс», к* 2 *и* 3 *можно подключить «минус».*

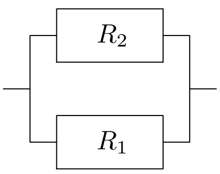
Напомним, как складываются сопротивления резисторов:

- Если два резистора с сопротивлениями R_1 и R_2 соответственно подключены последовательно, то



$$R_\Sigma = R_1 + R_2$$

- Если два резистора с сопротивлениями R_1 и R_2 соответственно подключены параллельно, то



$$\frac{1}{R_\Sigma} = \frac{1}{R_1} + \frac{1}{R_2}$$

Как R_Σ на рисунках обозначено сопротивление резистора, на который можно заменить указанную схему (резистора, эквивалентного схеме). Отметим, что эти формулы можно распространить и на большее количество резисторов — если параллельно подключены резисторы с сопротивлениями R_1, R_2, …, R_k, то для сопротивления эквивалентного резистора справедлива формула

$$\frac{1}{R_\Sigma} = \frac{1}{R_1} + \frac{1}{R_2} + \ldots + \frac{1}{R_k}.$$

Если же эти резисторы подключены последовательно, то для сопротивления эквивалентного резистора справедлива формула:

$$R_\Sigma = R_1 + R_2 + \ldots + R_k.$$

В схеме выделен один из контактов. К нему Вася по собственному замыслу может подсоединить «плюс», а к некоторым другим контактам Вася может подсоединить «минус». Теперь у юного электрика Василия возник вопрос, каким же будет суммарное сопротивление схемы при подключении «минуса» к каждому из этих контактов.

Напомним, что при подсчете сопротивления учитываются лишь резисторы, находящиеся на пути от «плюса» к «минусу».

Формат входного файла

В первой строке входного файла находятся четыре целых числа N, M, K и L $(2 \le N \le 1000,\ N-1 \le M \le 100000,\ 1 \le K \le N-1)$, где L — номер контакта, к которому подключен «плюс», а K — количество контактов, к которым может быть подключен «минус». Во второй строке находятся K чисел — номера контактов, к которым может быть подключен «минус». Далее следуют M строк — описания резисторов. В каждой строке тройка чисел a_i, b_i и R_i — это означает, что контакты с номерами a_i и b_i соединены резистором с сопротивлением R_i $(0 < R_i \le 100000)$.

Формат выходного файла

В выходном файле должны быть K строк, в каждой из которых два числа — номер контакта, к которому может быть подключен «минус», и сопротивление схемы при подключении схемы к этому контакту с точностью до 7 знаков после запятой.

Примеры

resistor.in	resistor.out
3 3 2 1 2 3 1 2 1 1 2 2 1 3 3	3 3.0 2 0.6666666666666666
4 4 1 1 4 1 2 1 2 3 2 2 3 2 3 4 1	4 3.0

Примечание

Решения, в которых рассматривается только случай, когда в схеме отсутствуют параллельно подключенные резисторы, будет оцениваться из 40 баллов.

Задача C. Палиндромность

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

Напомним, что *палиндромом* называется строка, которая читается одинаково как слева направо, так и справа налево. Например, палиндромами являются строки «abba» и «madam».

Для произвольной строки s введем операцию *деления пополам*, обозначаемую $half(s)$. Значение $half(s)$ определяется следующими правилами:

- Если s не является палиндромом, то значение $half(s)$ не определено;
- Если s имеет длину 1, то значение $half(s)$ также не определено;
- Если s является палиндромом четной длины $2m$, то $half(s)$ — это строка, состоящая из первых m символов строки s;
- Если s является палиндромом нечетной длины $2m + 1$, большей 1, то $half(s)$ — это строка, состоящая из первых $m + 1$ символов строки s.

Например, значения $half(\text{inforamatics})$ и $half(\text{i})$ не определены, $half(\text{abba}) = \text{ab}$, $half(\text{madam}) = \text{mad}$.

Палиндромностью строки s будем называть максимальное число раз, которое можно применить к строке s операцию деления пополам, чтобы результат был определен.

Например, палиндромность строк «informatics» и «i» равна 0, так как к ним нельзя применить операцию деления пополам даже один раз. Палиндромность строк «abba» и «madam» равна 1, а палиндромность строки «totottotot» равна 3, поскольку операция деления пополам применима к ней три раза: «totottotot» → «totot» → «tot» → «to».

Задана некоторая строка s. Необходимо изменить в ней минимальное число символов так, чтобы ее палиндромность стала равной k.

Формат входного файла

Первая строка входного файла содержит число k ($0 \le k \le 20$). Вторая строка входного файла содержит непустую строку s, состоящую из строчных букв латинского алфавита. Ее длина не превосходит 10^5 символов.

Формат выходного файла

В выходной файл выведите минимальное число символов, которые требуется изменить, или -1, если требуемым образом изменить строку невозможно.

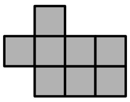
Примеры

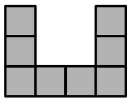
palin.in	palin.out
2 abaabc	1
1 ababa	1
2 aa	-1

Задача D. Выпуклые пермутомино

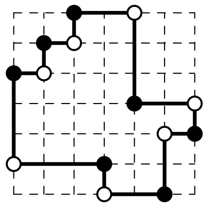
Имя входного файла:	`permutominoes.in`
Имя выходного файла:	`permutominoes.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Полимино представляет собой связную фигуру из единичных квадратиков. Полимино называется *выпуклым*, если каждая его строка и каждый его столбец представляет собой связное множество клеток. Например, полимино на рисунке (a) выпуклое, а на рисунке (b) — нет.


(a)


(b)

Полимино называется *пермутомино*, если у его контура $2n$ вершин, все его вершины имеют координаты от 1 до n, и никакие два ребра полимино не лежат на одной прямой. На рисунке приведен пример пермутомино с 14 вершинами. Отметим, что оно является выпуклым.



По заданному n найдите количество выпуклых пермутомино с $2n$ вершинами. При сравнении пермутомино, равные с точностью до поворота или отражения считаются различными.

Формат входного файла

Входной файл содержит число n ($2 \le n \le 300$).

Формат выходного файла

Выведите в выходной файл количество пермутомино с $2n$ вершинами, вычисленное по модулю $10^9 + 7$.

Пример

`permutominoes.in`	`permutominoes.out`
3	4
4	18

Заметьте, что бывают невыпуклые пермутомино с 8 и более вершинами, пример такого пермутомино приведен на рисунке справа. Вам следует посчитать количество выпуклых пермутомино.