Задача A. Сотовая связь в большом городе

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

В одном из больших городов нашей страны хорошо развит рынок сотовой связи. На рынке присутствуют несколько операторов, каждому из которых принадлежит некоторое количество базовых станций, с помощью которых организуется связь.

Каждая базовая станция характеризуется своими координатами (для простоты размерами базовых станций пренебрегаем) и радиусом надежной связи (если абонент находится на расстоянии, не превосходящем этот радиус, от базовой станции, то она может использоваться для работы с ним — передачи ему сигналов и приема сигналов от него).

Известный производитель сотовых телефонов *Mokea* планирует оснастить свою новую модель сотового телефона функцией определения базовых станций, с которыми может работать абонент. Вам же предстоит написать программу, которая по местоположению абонента для каждого оператора определит, сколько базовых станций этого оператора могут работать с абонентом.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \leqslant n \leqslant 10000$) — количество базовых станций в городе.

Далее идут описания этих базовых станций. Каждое описание занимает две строки. На первой расположено название оператора сотовой связи, которому принадлежит эта базовая станция, а на второй — три целых числа x, y, r ($-10000 \leqslant x, y \leqslant 10000$, $1 \leqslant r \leqslant 10000$) — соответственно ее координаты и радиус надежной связи.

Последняя строка входного файла содержит два целых числа x_a, y_a ($-10000 \leqslant x_a, y_a \leqslant 10000$) — координаты абонента.

Все координаты во входном файле даны в одной и той же декартовой прямоугольной системе координат. Названия операторов — это непустые строки длиной не более 50 символов, состоящие из цифр, строчных и прописных букв латинского алфавита. Прописные и строчные буквы латинского алфавита различаются (например, `MPS` и `mps` — два разных оператора).

Формат выходного файла

На первой строке выходного файла выведите число k операторов сотовой связи, работающих в городе (разумеется, два оператора считаются разными, если их названия не совпадают).

Далее выведите k строк. Каждая из этих строк должна содержать название оператора и количество базовых станций этого оператора, доступных абоненту. Первым должно идти название оператора, число базовых станций должно быть отделено от него одним пробелом. В этом списке операторы должны быть перечислены в том же порядке, в каком они встречаются во входном файле (см. примеры). Гарантируется, что $k \leqslant 100$.

Примеры

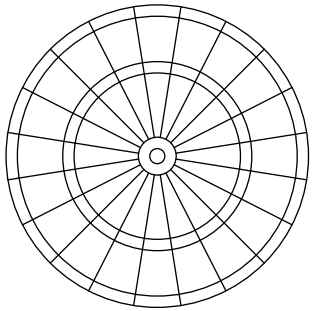
cellular.in	cellular.out
5 Megahorn 0 0 10 BeepLine 10 10 10 MPS 0 0 10 Ele2 0 0 1 SkyPink 100 100 10 5 5	5 Megahorn 1 BeepLine 1 MPS 1 Ele2 0 SkyPink 0
3 Megahorn 0 0 10 MPS 1 1 10 Megahorn 2 2 10 1 1	2 Megahorn 2 MPS 1

Задача B. Дартс

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

Игра в дартс очень популярна в Великобритании и Голландии. В игре принимают участие несколько игроков. Они по очереди бросают в мишень по три дротика.

В начале игры каждый игрок имеет некоторое количество очков, обычно 501. За каждое попадание дротика в мишень сумма игрока уменьшается на некоторое число, в зависимости от того, в какую часть мишени он попал. Первый, кто достигает нуля очков, считается победителем.



Внешний вид мишени показан на рисунке справа. Она разделена на 20 секторов, расположенных вокруг небольшого центрального круга. Этот круг, в свою очередь, делится на внутреннюю и внешнюю часть (иногда внутренняя часть называется *«яблочко»*). Попадание во внешнюю часть центрального круга оценивается 25 очков, а в «яблочко» — вдвое больше, то есть в 50 очков. Стоимость сектора равняется числу, которое на нем написано. Кроме того на мишени выделены два кольца — внутреннее и внешнее. Попадание в них оценивается соответственно в два и в три раза больше, чем в оставшуюся часть соответствующего сектора.

Существуют дополнительные правила для последней серии бросков, в которой игрок должен достичь нуля очков. В этой серии игроку придется бросить в мишень от одного до трех дротиков. Прави Во-первых, игрок должен достичь в точности нуля очков, получение отрицательной суммы считается ошибкой. Во-вторых, последний дротик должен быть «двойным», то есть попасть во внешнее кольцо какого-либо сектора или в «яблочко» — (оно считается удвоением внешней часть центрального круга).

Например, один из правильных способов закончить игру, имея 50 очков — бросить дротики в «18» и «D16». Способы «D20», «10», или «20», «T10» не подходят: последний бросок не является удвоенным. Еще один возможный способ победить в этом случае — просто попасть в «яблочко» («Bull»). По количеству оставшихся очков, найдите все способы правильно закончить игру.

Формат входного файла

Первая строка входного файла содержит число n — количество оставшихся очков ($1 \leqslant n \leqslant 200$).

Формат выходного файла

В первой строке выходного файла выведите k — количество способов правильно завершить партию. Каждая из следующих k строк должна содержать описание одного правильного способа. При этом число от 1 до 20 отвечает попаданию в соответствующий сектор. Буква 'D' перед числом обозначает попадание во внешнее (удваивающее) кольцо, а 'T' — во внутреннее (утраивающее). Внешняя часть центрального круга обозначается как «25», а «яблочко» (bull eye) — словом «Bull».

Примеры

darts.in	darts.out
5	7 1 D1 D1 1 2 D1 1 D2 D1 1 D1 T1 D1 2 1 D1 3 D1

Задача C. Внеземные гости

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

Недавно на поле фермера Джона были обнаружены следы приземления летающих тарелок. Об этом даже писала газета *Mew Yorc Courier*.

Поле фермера Джона имеет форму круга радиусом r_1. По сообщениям журналистов были обнаружены два следа от летающих тарелок, имевшие форму кругов. Один из них имел радиус r_2, второй — радиус r_3. Также сообщается, что они находились внутри поля фермера Джона и не пересекались (при этом, они, возможно, касались друг друга и/или границы поля).

Поскольку журналисты часто склонны преувеличивать масштабы событий, необходимо написать программу, которая будет проверять, могли ли иметь место события, описанные в газете.

Формат входного файла

Входной файл содержит три целых положительных числа — r_1, r_2, r_3 ($1 \leqslant r_1, r_2, r_3 \leqslant 10^9$).

Формат выходного файла

В выходной файл выведите слово YES, если информация, опубликованная в газете может соответствовать правде, и слово NO — иначе.

Примеры

extrater.in	extrater.out
1000000000 1000000000 1000000000	NO
1000000000 300000000 400000000	YES

Задача D. Друзья

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

Несколько человек решили поехать отдохнуть на природе, подышать свежим воздухом и т.п. Как это часто бывает, некоторые из них дружат друг с другом, а некоторые — нет. Для того, чтобы не испортить никому настроение, они решили разделиться на несколько групп. При этом, в каждой группе должно быть не более 5 человек и они должны дружить друг с другом.

Найдите такое разбиение людей на группы, в котором размер наибольшей группы был бы максимальным (среди всех разбиений).

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \leqslant n \leqslant 15$) — количество людей.

Следующие n строк содержат по n чисел. Если i-ый и j-ый люди дружат, то j-ое число $i+1$-ой строки равно 1, иначе — 0.

Формат выходного файла

В первой строке выходного файла выведите число групп m ($1 \leqslant m \leqslant n$).

Во второй строке выходного файла выведите n чисел (i-ое число — номер группы, в которой находится i-ый человек). Нумеруйте группы целыми числами от 1 до m. Каждая группа должна содержать по крайней мере одного человека.

Если решений несколько, то выведите любое.

Примеры

friends.in	friends.out
3 1 1 0 1 1 0 0 0 1	2 1 1 2
8 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	3 1 1 1 1 1 2 2 3

Задача E. Телешоу

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

В новом интеллектуальном телешоу участнику, проходящему в суперфинал, предлагается следующая игра: на каждом из n секторов большого барабана записывается буква латинского алфавита l_i. После минуты на размышления игрок указывает одну из позиций на барабане i. Его выигрыш вычисляется по такому правилу: для каждой позиции j меньшее из расстояний по и против часовой стрелке от i до j, измеренное в секторах, умножается на абсолютною величину разности номеров в алфавите букв l_i и l_j, после чего все такие величины суммируются.

А вы можете написать программу, находящую способ получения наибольшего выигрыша?

Формат входного файла

Первая строка содержит натуральное число n ($1 \leqslant n \leqslant 100000$) — размер барабана. Во второй строке задаются разделенные пробелами строчные латинские буквы, записанные на барабане.

Формат выходного файла

В первой строке выходного файла выведите наибольший выигрыш, который можно получить при заданном расположении букв на барабане. Во второй строке выведите номер какого-нибудь из секторов, на который игрок должен для этого указать.

Примеры

game.in	game.out
4 r e a r	55 3

Задача F. Распознавание языка

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

Важным понятием теории формальных грамматик и автоматов является *формальный язык*. Неформально его можно определить как некоторое множество слов, где под *словом* понимается некоторая строка из символов.

В этой задаче необходимо проверить, принадлежит ли данное слово языку $\{0^n1^n2^n, n \geqslant 1\}$. В этот язык входят те и только те слова, которые имеют такую структуру: в них нулей столько же, сколько единиц, а единиц — столько же, сколько и двоек. При этом любой ноль находится ближе к началу слова, чем любая единица, а любая единица находится ближе к началу слова, чем любая двойка. Например, слово 001122 принадлежит этому языку, а слово 0000111122220 — не принадлежит.

Формат входного файла

Первая строка входного файла содержит целое положительное число n ($n \leqslant 10$) — количество слов, которые надо проанализировать. Далее идут n строк, каждая из которых содержит по одному слову. Слова имеют длину не более тридцати тысяч символов и состоят только из нулей, единиц и двоек. Каждое из слов состоит хотя бы из одного символа.

Формат выходного файла

Выходной файл должен содержать ровно n строк. Для каждого слова из входного файла выведите по одной строке, содержащей слово YES, если оно принадлежит указанному выше языку, и NO — иначе.

Примеры

lang.in	lang.out
3 001122 00011122222 000111222	YES NO YES
2 0000111122220 012	NO YES

Задача G. Цветной лабиринт

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

В одном из парков одного большого города недавно был организован новый аттракцион *Цветной лабиринт*. Он состоит из n комнат, соединенных m двунаправленными коридорами. Каждый из коридоров покрашен в один из ста цветов, при этом от каждой комнаты отходит не более одного коридора каждого цвета. При этом две комнаты могут быть соединены любым количеством коридоров, но никакой коридор не соединяет комнату саму с собой.

Человек, купивший билет на аттракцион, оказывается в комнате номер один. Кроме билета, он также получает описание пути, по которому он может выбраться из лабиринта. Это описание представляет собой последовательность цветов $c_1 ... c_k$. Пользоваться ей надо так: находясь в комнате, надо посмотреть на очередной цвет в этой последовательности, выбрать коридор такого цвета и пойти по нему. При этом если из комнаты нельзя пойти по коридору соответствующего цвета, то человеку приходится дальше самому выбирать, куда идти.

В последнее время в администрацию парка стали часто поступать жалобы от заблудившихся в лабиринте людей. В связи с этим, возникла необходимость написания программы, проверяю-

щей корректность описания и пути, и, в случае ее корректности, сообщающей номер комнаты, в которую ведет путь.

Описание пути некорректно, если на пути, который оно описывает возникает ситуация, когда из комнаты нельзя пойти по коридору соответствующего цвета.

Формат входного файла

Первая строка входного файла содержит два целых числа: n ($1 \leqslant n \leqslant 10000$) и m ($1 \leqslant m \leqslant 100000$) — соотвественно количество комнат и коридоров в лабиринте.

Следующие m строк содержат описания коридоров. Каждое описание содержит три числа u ($1 \leqslant u \leqslant n$), v ($1 \leqslant v \leqslant n$), c ($1 \leqslant c \leqslant 100$) — соответственно номера комнат, соединенных этим коридором, и цвет коридора.

Следующая, $(m+2)$-ая строка входного файла содержит длину описания пути — целое число k ($0 \leqslant k \leqslant 100000$). Последняя строка входного файла содержит k целых чисел, разделенных пробелами, — описание пути по лабиринту.

Формат выходного файла

Если описание пути некорректно, выведите строку `INCORRECT`, иначе выведите номер комнаты, в которую ведет описанный путь в выходной файл. Помните, что путь начинается в комнате номер один.

Примеры

maze.in	maze.out
3 3 1 2 10 1 3 5 5 10 10 10 10 5	3
2 2 1 2 10 2 3 5 5 5 10 10 10 10	INCORRECT
3 3 1 2 10 1 3 5 4 10 10 10 5	INCORRECT

Задача H. Точки на прямой

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

На прямой отмечено n точек. Требуется найти такой отрезок длины l, на котором лежат m из отмеченных точек ($m \geqslant 2$), что величина l/m минимальна. Считается, что точки, совпадающие с одним из концов отрезка, лежат на нем.

Формат входного файла

В первой строке входного файла содержится количество точек n ($2 \leqslant n \leqslant 10000$). На второй строке записаны координаты этих точек x_i — целые числа, разделенные пробелами. При этом $|x_i| \leqslant 30000$ и $x_i < x_j$ при $i < j$.

Формат выходного файла

В выходной файл выведите координаты начала и конца найденного отрезка a и b ($a < b$).

Примеры

points.in	points.out
3 -2 -1 1	-2 -1

Задача I. Поиск

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

Обычно программы, предоставляющие возможность поиска заданных строк в текстовых файлах, недостаточно гибко обрабатывают различные пробельные символы. Например, если в некотором тексте слова «`Internet`» «`olympiad`» разделены переводом строки, словосочетание «`Internet olympiad`» чаще всего не будет обнаружено в этом месте. В данной задаче пробельными символами мы будем считать пробелы, символы табуляции (код символа 9), а так же переводы строк. Любую последовательность идущих подряд непробельных символов будем называть словом.

Ваша программа должна производить обработку одного запроса на поиск словосочетания в тексте. Словосочетание будет задано как последовательность слов, состоящих из цифр и строчных и прописных букв латинского алфавита, каждые два из которых разделены пробелом. Будем считать, что некоторая последовательность символов, первый и последний из которой непробельные, является вхождением этого словосочетания в текст, если после замены каждого блока пробельных символов из этой последовательности на один пробел она совпадет с заданным словосочетанием с точностью до регистра букв. Для представления ответа перед каждым вхождением словосочетания в исходный текст следует поставить символ «`@`».

Формат входного файла

Первая строка входного файла, заканчивающаяся переводом строки, задает запрос. Длина словосочетания не превосходит 100 символов. Последующие строки описывают сам текст, размер которого не превосходит 2000 символов. Файл заканчивается переводом строки.

Формат выходного файла

В выходной файл выведите результат применения к тексту описанной процедуры. Он должен отличаться от исходного текста только добавлением символов «`@`».

Примеры

search.in	search.out
internet olympiad Internet Olympiads Everyone is welcome to participate in internet olympiads. Jury of internet olympiads	@Internet Olympiads Everyone is welcome to participate in @internet olympiads. Jury of @internet olympiads

Задача J. Словарные квадраты

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

Некоторые наборы из n слов длины n обладают интересным свойством — их можно расположить в клетках квадрата $n \times n$ так, что все слова набора можно прочитать как во вертикали, так и по горизонтали.

Примером такого набора слов является { “DATE”, “FIND”, “IDEA”, “NEXT” }. Их можно расположить так:

F	I	N	D
I	D	E	A
N	E	X	T
D	A	T	E

Заметьте, что каждое слово можно прочитать как по горизонтали, так и по вертикали. Такие квадраты называются *словарными квадратами*, наибольший известный словарный квадрат в английском языке имеет размер 10×10.

Рассмотрим еще один пример словарного квадрата:

C	R	A	B
R	A	R	E
A	R	T	S
B	E	S	T

Вам даны такие $2n$ слов, что из них можно построить два различных словарных квадрата размера $n \times n$. Ваша задача состоит в том, чтобы разбить эти слова на две группы, по n слов в каждой, и построить из слов каждой группы словарный квадрат.

Гарантируется, что все данные вам слова являются английскими словами (некоторые из них могут быть достаточно редкими словами, именами, или специальными терминами).

Формат входного файла

Первая строка входного содержит целое число n ($2 \leqslant n \leqslant 10$). Каждая из следующих $2n$ строк содержит слово, состоящее из заглавных букв латинского алфавита. Каждое слово содержит ровно n букв.

Формат выходного файла

Выведите два словарных квадрата, постоенных из данных слов. Разделите квадраты пустой строкой.

Примеры

square.in	square.out
4 ARTS BEST CRAB DATE FIND IDEA NEXT RARE	FIND IDEA NEXT DATE CRAB RARE ARTS BEST

Задача K. Подстрока

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

Рассмотрим слова, состоящие из букв «A», «B», «a» и «b». Скажем, что два слова эквивалентны, если одно может быть получено из другого с помощью следующих операций:

- удалить в любой позиции подстроку, равную Aa, aA, Bb или bB;
- добавить в любую позицию подстроку, равную Aa, aA, Bb или bB.

Например, слова $abAaBBabbA$ и $aAaBabaAbA$ эквивалентны:

$$abAaBBabbA \to abBBabbA \to aBabbA \to aAaBabbA \to aAaBabaAbA,$$

а слова $abAB$ и $baBA$ — нет.

Интересно отметить, что для произвольных слов α и β найдется такое слово γ, эквивалентное α, которое содержит β в качестве подстроки. Ваша задача — найти кратчайшее такое слово.

Формат входного файла

Первая строка входного файла содержит α. Вторая строка содержит β. Оба слова непусты и каждое из них имеет длину не больше 2000.

Формат выходного файла

Выведите одну строку, содержащую минимальное по длине слово, эквивалентное α, содержащее β в качестве подстроки. Если решений несколько, выведите любое.

Примеры

subword.in	subword.out
abAaBBabbA AaBaba	aAaBabaAbA

Задача L. Таблица HTML

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

При разработке программ для просмотра веб-страниц одной из наиболее сложных задач является корректное отображение таблиц. Компания «*Kozilla*», в которой вы работаете, планирует

разработать новую модель браузера «*Waterrat*», и просит вас написать фрагмент ядра отображения веб-страниц, ответственный за формирование структуры таблиц.

Рассмотрим упрощенную версию фрагмента языка HTML, ответственного за описание таблиц. HTML-документ представляет собой последовательность открывающих и закрывающих *тегов*, которые ограничивают элементы документа. Внутри тегов и между ними может находится текст, который представляет собой содержимое веб-страницы.

Открывающий представляет собой последовательность символов вида `<имя-тега атрибуты>`, закрывающий — `</имя тега>`. Здесь атрибуты — последовательность описаний вида `имя="значение"`, атрибуты позволяют указать различные свойства элемента веб-страницы.

Таблица представляет собой набор ячеек, упорядоченных по строкам и столбцам. Для задания таблицы используется пара тегов `<table>` — `</table>`. Между ними находится описание таблицы.

Описание таблицы представляет собой последовательность описаний строк таблицы. Описание каждой строки представляет собой пару тегов `<tr>` — `</tr>`, между которыми расположено описание последовательности ячеек этой строки. Описание ячейки представляет собой пару тегов `<td>` — `</td>`, между которыми находится текст, который должен отображаться в этой ячейке.

Заметим, что некоторые ячейки в некоторых строках могут отсутствовать.

Для рисования более сложных таблиц используется возможность рисовать ячейку, занимающую более одной строки или столбца. А именно, у тега `<td>` могут быть атрибуты `rowspan` и `colspan`, задающие количество строк и столбцов, которые занимает ячейка, соответственно. При этом описание соответствующей ячейки размещается в описании таблицы в том месте, где располагалось бы описание ее верхнего левого угла.

Таблица формируется строка за строкой. Если в процессе формирования строки очередная ячейка не помещается в строку, поскольку ей мешает некоторая ячейка X одной из предыдущих строк, считается что произошла ошибка.

По заданному описанию таблицы вам требуется вывести в выходной файл ее изображение. При этом следует игнорировать содержимое ячеек, выводя вместо него в качестве содержимого каждой ячейки пробелы.

Формат входного файла

Входной файл содержит фрагмент веб-страницы, который описывает таблицу. Гарантируется, что описание корректно и соответствует в точности одной таблице. Входной файл содержит только следующие теги: `<table>`, `</table>`, `<tr>`, `</tr>`, `<td>` и `</td>`. Только теги `<td>` могут иметь атрибуты, при этом используются только атрибуты `rowspan` и `colspan`. При распознавании имен тегов и атрибутов следует игнорировать регистр букв. Значения атрибутов `rowspan` и `colspan` — целые положительные числа, не превосходящие 10.

Таблицы содержит по крайней мере одну строку, описание каждой строки содержит описание по крайней мере одной ячейки.

Внутри пары тегов `<td>` — `</td>` может находиться произвольный текст, состоящий из символов с ASCII-кодами от 32 до 126, кроме символа «<». Между другими тегами, а также внутри тега между названием тега и названием атрибута, между атрибутами, по обе стороны от знака «=» и между кавычкой, следующей после значения последнего атрибута, и символом «>» может быть произвольное число пробельных символов — пробелов и символов перевода строки. Их следует игнорировать.

Гарантируется, что описанная таблица содержит не более 100 ячеек, размер входного файла не превышает 10 килобайт.

Формат выходного файла

Выведите в выходной файл изображение таблицы.

Используйте следующие символы:

'+' (плюс) — для отображения пересечения линий;
'|' (вертикальная черта, код ASCII 124) — для отображения вертикальных линий;
'-' (минус) — для отображения горизонтальных линий;
' ' (пробел) — во всех остальных случаях.

Ячейка, занимающая m строк и n столбцов должна быть изображена с использованием $2m - 1$ строки по $2n - 1$ пробела.

Если в описании таблицы содержатся ошибка, выведите "ERROR" в выходной файл.

Примеры

table.in	table.out

`<table>`
`<TR><td>1</td><td>2</td></TR>`
`<TR><td>3</td></TR>`
`<TR><td>4</td><td>5</td><td>6</td></TR>`
`</table>`
	`+-+-+`
	`| | |`
	`+-+-+`
	`| |`
	`+-+-+-+`
	`| | | |`
	`+-+-+-+`

`<table>`
`<tr>`
`  <td>1</td>`
`  <td rowspan="2">2</td>`
`</tr>`
`<tr>`
`  <td colspan="2≫3</td>`
`</tr>`
`</table>`
	`ERROR`

`<table>`
`<tr>`
`  <td colspan="3≫1</td>`
`  <td rowspan="2" colspan="2">2</td>`
`</tr>`
`<tr>`
`  <td colspan="2≫3</td>`
`  <td>4</td>`
`  <td rowspan="2">5</td>`
`</tr>`
`<tr>`
`  <td>6</td>`
`</tr>`
`<tr>`
`  <td>7</td>`
`</tr>`
`<tr>`
`  <td rowspan="2">8</td>`
`  <td rowspan="2">9</td>`
`</tr>`
`<tr>`
`  <td colspan="2" rowspan="2">0</td>`
`</tr>`
`</table>`
	`+---+--+`
	`|  |  |`
	`+--+-+  +-+`
	`|  | |  | |`
	`+-+-+-+--+ |`
	`| |  | |`
	`+-+  +-+`
	`| |`
	`+-+-+`
	`| | |`
	`| | +--+`
	`| | |  |`
	`+-+-+  |`
	`  |  |`
	`  +--+`