Задача 1. Расстановка ферзей

Источник:	базовая II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Дано прямоугольное клеточное поле из M строк и N столбцов. Нужно расставить ферзей на этом поле согласно правилам:

1. В каждой строке поля должен стоять один ферзь.
2. Никакие два ферзя не должны бить друг друга.

Ферзи на поле бьют всё, что стоит на той же горизонтали, вертикали или диагонали (как в шахматах). В некоторые клетки поля ставить ферзя запрещено.

Формат входных данных

В первой строке дано два целых числа: M — количество строк и N — количество столбцов ($1 \leqslant M, N \leqslant 12$).

В остальных M строках записана карта поля. Каждая из этих строк содержит ровно N символов. Символ вопроса `'?'` означает, что в этой клетке можно ставить ферзя, а можно не ставить. Символ точки `'.'` означает, что в этой клетке ферзя ставить запрещено.

Формат выходных данных

Если искомого решения не существует, нужно вывести слово `NO`, и больше ничего не выводить.

Если решение существует, то нужно вывести слово `YES` в первую строку, и решение в остальные M строк. Решение должно быть выведено в виде карты поля, в том же формате, как во входных данных. Только каждый символ вопроса надо заменить на `X` (заглавную букву «икс»), если в этой клетке стоит ферзь, и точку — если не стоит.

Если решений несколько, можно вывести любое из них.

Пример

input.txt	output.txt
5 8 ?.?...?? ?????.?? ......?. ??.?.?.? ?????..?	YES X....... ..X..... ......X. .X...... ...X....
3 3 ??? ??? ???	NO

Задача 2. Число разбиений

Источник:	базовая II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Разбиением числа N на слагаемые называется набор целых положительных чисел, сумма которых в точности равна N. Два разбиения считаются одинаковыми, если соответствующие наборы отличаются только порядком чисел. Нужно найти количество различных разбиений числа N на слагаемые.

Поскольку ответ может быть большим, найдите его остаток от деления на $10^9 + 7$.

Формат входных данных

В первой строке дано одно целое число N — число, разбиения которого нужно считать ($1 \leqslant N \leqslant 1\,000$).

Формат выходных данных

Найдите количество способ разбить число N на целые положительные слагаемые, и выведите остаток от деления этого числа на $1\,000\,000\,007$.

Пример

input.txt	output.txt
10	42
20	627
1000	709496666

Задача 3. Максимальный поток

Источник:	базовая I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Дан неориентированный граф без петель и кратных рёбер из N вершин, пронумерованных целыми числами от 1 до N. Для каждого ребра известна его пропускная способность.

Требуется найти величину максимального потока из вершинs 1 в вершину N. По каждому ребру поток может течь в любую сторону.

Формат входных данных

В первой строке входного файла записано два целых числа N и K — количество вершин ($2 \leqslant N \leqslant 100$) и количество рёбер ($0 \leqslant K \leqslant \frac{N \cdot (N-1)}{2}$) соответственно.

Далее идёт K строк, описывающих рёбра графа, каждая из которых содержит по три целых числа: v, u и c — номера вершин, соединённых рёбром ($1 \leqslant v, u \leqslant N$), и пропускная способность ребра ($1 \leqslant c \leqslant 10^6$) соответственно.

Формат выходных данных

Выведите одно целое число — величину максимального потока из вершины 1 в вершину N.

Пример

input.txt	output.txt
5 7 1 2 2 2 5 5 1 3 6 3 4 2 4 5 1 3 2 3 2 4 1	6

Задача 4. Наибольшая возрастающая подпоследовательность

Источник:	основная II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Дан массив из N чисел. Нужно найти в этом массиве такую подпоследовательность, что:

1. Числа этой подпоследовательности строго возрастают (слева направо).
2. Количество элементов в этой подпоследовательности максимально возможное.

Формат входных данных

В первой строке дано одно целое число N — размер массива ($1 \leqslant N \leqslant 5\,000$). Во второй строке записано N знаковых 32-битных целых чисел через пробел.

Формат выходных данных

В первую строку нужно вывести целое число K — количество элементов в искомой подпоследовательности. Саму подпоследовательность нужно ввести в оставшихся K строках. Каждый элемент подпоследовательности следует выводить в формате "`A[i] = k`", где i — индекс элемента (нумеруя с единицы), а k — значение элемента. Естественно, элементы подпоследовательности нужно выводить в порядке возрастания.

Если решений несколько, можно вывести любое из них.

Пример

input.txt	output.txt
12 18 3 18 5 7 10 5 18 20 19 7 18	6 A[2] = 3 A[4] = 5 A[5] = 7 A[6] = 10 A[8] = 18 A[9] = 20

Задача 5. Раздать задания

Источник:	основная I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Прокопий раздаёт своим любимым ученикам индивидуальные задания. К сожалению, не всякое задание подойдёт всякому ученику: какие-то задания могут оказаться слишком сложными для того или иного ученика, или того хуже — слишком простыми!

Прокопий хорошо знает, какое задание подойдёт какому ученику. Он не хочет выдавать одинаковое задание нескольким ученикам, и не хочет давать несколько заданий какому-либо ученику. Помогите ему озадачить как можно больше учеников.

Формат входных данных

В первой строке дано три целых числа: A — количество учеников, B — количество заданий и M — количество совместностей ($1 \leqslant A, B \leqslant 100$, $0 \leqslant M \leqslant AB$).

В остальных M строках описаны совместности. Каждая из этих строк содержит два целых числа x и y и означает, что ученику под номером x подходит задание под номером y. ($1 \leqslant x \leqslant A$, $1 \leqslant y \leqslant B$).

Гарантируется, что каждая совместность описана ровно один раз.

Формат выходных данных

В первой строке нужно вывести целое число K — максимальное количество озадаченных учеников ($0 \leqslant K \leqslant \min(A, B)$). В остальных K строках нужно вывести назначение заданий ученикам. В каждой строке нужно записать два целых числа u и v, означающих, что ученику под номером u нужно выдать задание под номером v ($1 \leqslant u \leqslant A$, $1 \leqslant v \leqslant B$).

Пример

input.txt	output.txt
6 5 9 1 1 2 1 2 2 3 4 2 3 4 4 5 4 5 5 6 5	4 1 1 3 4 5 5 2 2

Задача 6. Самоизоляция

Источник:	основная I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Вовочка наконец понял, что коронавирус — это не шутка, и поспешно уехал на дачу, пока не начался зомбипокалипсис. Поначалу это принесло ему облегчение, но теперь он боится, что зомби проникнут на дачу ночью и съедят его мозги. От этой мысли он даже спать перестал! Теперь он хочет устроить полную самоизоляцию, так чтобы никто не мог дойти до его дачи из города.

На дачу из города можно попасть только через лес, в которой есть много тропинок, соединяющихся в разных полянках. Вовочка уверен, что благоразумные зомби будут ходить только по тропинкам, ведь в лесу уже появились клещи! Для каждой тропинки он посчитал, сколько грамм в тротиловом эквиваленте нужно, чтобы привести её в негодность.

Определите точный план, как отсечь дачу от города, потратив на это минимум взрывчатки.

Формат входных данных

В первой строке входного файла записано два целых числа N и M — количество полянок ($2 \leqslant N \leqslant 100$) и количество тропинок ($0 \leqslant M \leqslant \frac{N \cdot (N-1)}{2}$) соответственно. Во второй строке указаны два целых числа S и T — номер полянки, где расположена дача, и номер полянки, выходящей в город ($1 \leqslant S \neq T \leqslant N$). Полянки пронумерованы целыми числами от 1 до N.

По полянкам можно ходить в обоих направлениях. Гарантируется, что каждую пару полянок непосредственно соединяет не более одной тропинки, и что не бывает тропинки, которая начинается и заканчивается в одной и той же полянке.

Далее идёт M строк, описывающих тропинки, каждая из которых содержит по три целых числа: u, v и c — номера полянок, соединённых тропинкой ($1 \leqslant u, v \leqslant N$), и сколько грамм взрывчатки нужно для её уничтожения ($1 \leqslant c \leqslant 10^6$) соответственно.

Формат выходных данных

В первой строке выведите два целых числа: A — какое минимальное количество грамм взрывчатки потребуется и K — сколько тропинок для этого надо взорвать ($0 \leqslant K \leqslant M$). В остальных K строках нужно вывести тропинки, которые нужно привести в негодность, по одной тропинке в строке. Для каждой из этих тропинок выведите номера полянок, которые она соединяет.

Пример

input.txt	output.txt
5 7 1 5 1 2 2 2 5 5 1 3 6 3 4 2 4 5 10 3 2 3 2 4 1	7 3 1 2 3 4 2 3

Задача 7. Sentinel: Descendants in Time

Источник:	основная II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

В этой задаче предлагается решить головоломку из Myst-подобной компьютерной игры "Sentinel: Descendants in Time". Головоломка отлично показано в этом видео. Ближе к концу видео становится ясно, что игрок решает головоломку рекурсивным перебором. При этом используются осмысленные отсечения (и даже некоторые приоритеты).

Есть M лампочек и N пультов, на каждом пульте K переключателей. На каждой лампочке нужно добиться напряжения в ровно L единиц. Изначально все переключатели выключены, и напряжение на лампочках нулевое.

На каждом пульте требуется включить **ровно один** переключатель (все остальные должны быть выключены). При включении любого переключателя напряжение на некотором подмножестве лампочек вырастает на единицу. Это подмножество своё для каждого переключателя, и дано во входном файле.

Требуется определить для каждого пульта, какой переключатель нужно на нём включить, чтобы добиться в точности требуемого напряжения на всех лампочках.

Формат входных данных

В первой строке дано четыре целых числа: N — количество пультов, K — количество переключателей на каждом пульте, M — количество лампочек и L — требуемое напряжение на каждой лампочке ($1 \leqslant N, K \leqslant 10$, $3N \leqslant M \leqslant 100$, $0 \leqslant L \leqslant N$).

Далее описано влияние всех переключателей на все лампочки. Записано $N \cdot K$ строк, по M символов в каждой. Если j-ый переключатель на i-ом пульте увеличивает напряжение на t-ой лампочке, то t-ый символ в строке под номером $(i \cdot K + j)$ равен `'X'` (большая буква «икс»), а если нет — то `'.'` (символ точки).

Могут быть добавлены дополнительные пустые строки в описании.

Формат выходных данных

Если искомого решения не существует, нужно вывести слово `NO`, и больше ничего не выводить.

Если решение существует, то нужно вывести слово `YES` в первую строку, и решение в остальных N строках. В каждой i-ой из этих строк нужно вывести номер переключателя, который нужно включить на i-ом пульте (переключатели нумеруются с единицы).

Если решений несколько, можно вывести любое из них.

Комментарий

В подобных задачах почти бессмысленно оценивать время работы асимптотически — сильно многое зависит от качества составления тестов. В данной задаче тесты сгенерированы случайным образом без попытки кого-то специально подловить.

Пример

input.txt	output.txt
4 8 20 3 XXXX.X.XXXXXXX.XXX.. XXXX.XX.XXX.XX.XX.XX X.XXXX.XXX.XX.XX.XXX .XX.XXXXXX.XXX.XXX.X XXX.X.XX.XXXXXX.XX.X X.XXX..X.XXX.XXXXXXX X.XXXXXXXX.XX.X..XX XXX.XX.XXXX.XXXX.X.X XX.XXX.XX.XX.XXXXX.X XXXXX.X.XX.XXX.XX.XX XX.XXXX..XXX.X.XXXXX XXX.XXX.XXXX.XXXX.X. XX.XXX.X.XXX.XXXX.XX XX.XXXXXX.XXXX..XXX. X.XX.XXX.XXXXX.XXX.X .XXXXXX.XXXX.X.XXX.X XXXX..XX.XXXX.XXXX.X XX.XXXXXX.X.XXX.XXX. XX.XXX.XXX.X.XX.XXXX XX.XXX.XXXXXX.XX.XX. X.XXX..XXXX.XXXXX.XX X.XX.XXXX.XXX.XXXXX. XXX.XX..XXXXX.XXXX.X X.XX.XXX.X.XXX.XXXXX .XXX.XXXXXX.XX.XX.XX X.XXX.XX.XXXXXX.XXX. XX.XXXXX.XXX.XXX..XX .XXXX.XXXX.XXXX.X.XX .XXXXX.XXX.XX.X.XXXX XX.XXXX.XXX.XXX.X.XX XXXXX.XX..XX.X.XXXXX X.XXX.XX.XXXXX.XXX.X	YES 8 3 6 4
1 1 5 1 X.X.X	NO

Пояснение к примеру

Первый пример в точности соответствует ситуации из игры, если пульты и переключатели пронумеровать слева направо, а лампочки на мосте пронумеровать сверху вниз — сначала левую половину, потом правую. Приведено единственное решение этой задачи, к которому и приходит игрок в видео.

Задача 8. Рюкзак

Источник:	основная II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Имеется N предметов, для каждого предмета известны его вес и его стоимость. Хочется унести набор предметов максимальной суммарной стоимости. Однако унести предметы можно только в рюкзаке. Рюкзак один, и он выдерживает вес не более W.

Требуется определить, какие предметы надо забрать, так чтобы рюкзак выдержал их суммарный вес, и чтобы их суммарная стоимость была максимальна.

Формат входных данных

В первой строке дано два целых числа: N — количество предметов и W — какой максимальный вес выдерживает рюкзак ($1 \leqslant N \leqslant 200$, $1 \leqslant W \leqslant 5 \cdot 10^4$).

Далее идёт N строк, которые описывают предметы. В каждой строке записано два целых числа: w_i — вес предмета и c_i — стоимость предмета ($1 \leqslant w_i \leqslant W$, $1 \leqslant c_i \leqslant 10^6$).

Формат выходных данных

На выход нужно вывести оптимальное решение.

В первую строку нужно вывести три целых числа: K — количество взятых предметов, $\overline{W}$ — суммарный вес этих предметов и $\overline{C}$ — суммарная стоимость этих предметов. Во второй строке нужно вывести K целых чисел — номера взятых предметов в любом порядке. Предметы нумеруются в порядке их записи во входных данных, начиная с единицы.

Если решений несколько, можно вывести любое из них.

Пример

input.txt	output.txt
5 20 5 10 4 12 6 3 3 8 6 20	4 18 50 1 2 4 5

Задача 9. Судоку

Источник:	повышенной сложности II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Предлагается написать программу, которая будет автоматически решать судоку.

Судоку — это головоломка на клеточном поле размера 9 на 9. Изначально какие-то клетки поля пустые, а в каких-то клетках записаны цифры (в пределах от 1 до 9). Требуется в каждую пустую клетку вписать цифру (от 1 до 9), так чтобы было верно:

1. В каждой строке поля каждая цифра встречается ровно один раз.
2. В каждом столбце поля каждая цифра встречается ровно один раз.
3. В каждом блоке поля каждая цифра встречается ровно один раз.

Заметим, что поле можно разбить на 9 строк, на 9 столбцов, или на 9 квадратных блоков размера 3 на 3. В каждой из этих частей ровно 9 клеток, и в них должны быть записаны в точности все цифры от 1 до 9.

Формат входных данных

В первой строке дано одно целые число N — количество полей судоку, которые нужно решить ($1 \leqslant N \leqslant 100$). Далее приведено N полей, каждое поле — отдельная головоломка.

Каждое поле описывается в виде девяти строк, в каждой строке девять символов. Символ может быть либо цифрой от 1 до 9, либо точкой (точка обозначает пустую клетку). Поля **могут** быть отделены друг от друга пустыми строками.

Гарантируется, что на каждом поле существует решение. Скорее всего, оно также будет единственным.

Формат выходных данных

Для каждого записанного во входных данных поля нужно вывести решение. Решения можно отделять друг от друга дополнительными пустыми строками.

Каждое решение — это девять строк по девять символов в каждой. Все символы должны быть цифрами от 1 до 9.

Если решений на головоломку несколько, можно вывести любое из них.

Пример

input.txt	output.txt
3 ..3.2.6.. 9..3.5..1 ..18.64.. ..81.29.. 7.......8 ..67.82.. ..26.95.. 8..2.3..9 ..5.1.3.. ..1..7.9. 59..8...1 .3.....8. .....58.. .5..6..2. ..41..... .8.....3. 1...2..79 .2.7..4.. 48.3..... ....71 .2....... 7.5....6. ...2..8.. ....1.76... 3.....4.. ....5....	483921657 967345821 251876493 548132976 729564138 136798245 372689514 814253769 695417382 861357294 597482361 432619785 916275843 358964127 274138956 789541632 143826579 625793418 487312695 593684271 126597384 735849162 914265837 268731549 851476923 379128456 642953718

Комментарий

Подсказка:

1. Клетки можно перебирать в разном порядке.
2. Попробуйте решить несколько простых судоку самостоятельно.

Задача 10. Наибольшая возрастающая подпоследовательность+

Источник:	повышенной сложности II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Дан массив из N чисел. Нужно найти в этом массиве такую подпоследовательность, что:

1. Числа этой подпоследовательности строго возрастают (слева направо).
2. Количество элементов в этой подпоследовательности максимально возможное.

Формат входных данных

В первой строке дано одно целое число N — размер массива ($1 \leqslant N \leqslant 100\,000$). Во второй строке записано N знаковых 32-битных целых чисел через пробел.

Формат выходных данных

В первую строку нужно вывести целое число K — количество элементов в искомой подпоследовательности. Саму подпоследовательность нужно ввести в оставшихся K строках. Каждый элемент подпоследовательности следует выводить в формате "`A[i] = k`", где i — индекс элемента (нумеруя с единицы), а k — значение элемента. Естественно, элементы подпоследовательности нужно выводить в порядке возрастания.

Если решений несколько, можно вывести любое из них.

Пример

input.txt	output.txt
12 18 3 18 5 7 10 5 18 20 19 7 18	6 A[2] = 3 A[4] = 5 A[5] = 7 A[6] = 10 A[8] = 18 A[9] = 20

Комментарий

Один из возможных способов решения:

1. Если отсортировать все значения в последовательности и пронумеровать их по порядку, то можно заменить каждый элемент последовательности на его номер. После такого преобразования все элементы будут в диапазоне от 0 до $N - 1$.
2. На каждом шаге динамического программирования нужно выбрать максимальный известный результат среди тех элементов, которые больше/меньше некоторого порога. Этот выбор максимума сводится к выбору максимума на отрезке, и его можно ускорить с помощью блоков.

Задача 11. Рёберно непересекающиеся пути

Источник:	повышенной сложности I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды*
Ограничение по памяти:	16 мебагайт

Дан граф с выделенными вершинами s и t. Требуется найти максимальное возможное количество путей из s в t по рёбрам графа, такое что никакое ребро не используется в этих путях дважды.

Формат входных данных

В первой строке дано три целых числа: N — количество вершин, M — количество рёбер и O — флаг ориентированности графа ($1 \leqslant N \leqslant 3\,000$, $0 \leqslant M \leqslant 10\,000$, $O \in \{0, 1\}$). Если число O равно 0, то граф неориентированный, и каждое ребро можно использовать в путях в любую сторону. Если O равно 1, то граф ориентированный, и каждое ребро можно использовать только в одну сторону.

Во второй строке дано два целых числа: s — номер вершины, в которой должны начинаться пути, и t — номер вершины, в которой должны заканчиваться пути ($1 \leqslant s \neq t \leqslant N$).

В остальных M строках описаны рёбра графа. Каждая из этих строк содержит два целых числа: a — номер вершины, в которой начинается ребро, и b — номер вершины, в которой заканчивается ребро ($1 \leqslant a \neq b \leqslant N$). Если граф неориентированный (т.е. $O = 1$), то это ребро двустороннее.

В графе могут быть кратные рёбра.

Формат выходных данных

В первой строке нужно вывести целое число K — максимальное количество рёберно непересекающихся путей из s и t. Далее должно быть K блоков, каждый из который описывает отдельный путь.

В первой строке каждого блока должно быть целое число Q — количество рёбер в пути. В следующих Q строках должны быть описаны эти рёбра в порядке их следования в пути в направлении от s к t. Каждое ребро описывается тремя целыми числами: u — номер начальной вершины, i — номер ребра и v — номер конечной вершины ($1 \leqslant u \neq v \leqslant N$, $1 \leqslant i \leqslant M$). Рёбра графа нумеруются в порядке их описания во входных данных. В неориентированном графе вершины u и v нужно выдавать так, что путь проходит через ребро uv в направлении от u к v.

Пример

input.txt	output.txt
8 15 0 1 4 2 3 3 5 4 5 1 8 8 2 3 8 3 4 3 6 4 8 6 4 6 8 1 6 7 6 8 7 1 7	3 2 1 4 8 8 9 4 3 1 12 6 6 8 3 3 7 4 3 1 15 7 7 13 6 6 10 4
2 5 1 2 1 2 1 1 2 2 1 2 1 1 2	3 1 2 1 1 1 2 3 1 1 2 4 1

Задача 12. Вершинно непересекающиеся пути

Источник:	повышенной сложности I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды*
Ограничение по памяти:	16 мебагайт

Дан граф с выделенными вершинами s и t. Требуется найти максимальное возможное количество путей из s в t по рёбрам графа, такое что никакая вершина не используется в этих путях дважды (за исключением первой и последней вершин путей).

Формат входных данных

В первой строке дано три целых числа: N — количество вершин, M — количество рёбер и O — флаг ориентированности графа ($1 \leqslant N \leqslant 3\,000$, $0 \leqslant M \leqslant 10\,000$, $O \in \{0, 1\}$). Если число O равно 0, то граф неориентированный, и каждое ребро можно использовать в путях в любую сторону. Если O равно 1, то граф ориентированный, и каждое ребро можно использовать только в одну сторону.

Во второй строке дано два целых числа: s — номер вершины, в которой должны начинаться пути, и t — номер вершины, в которой должны заканчиваться пути ($1 \leqslant s \neq t \leqslant N$).

В остальных M строках описаны рёбра графа. Каждая из этих строк содержит два целых числа: a — номер вершины, в которой начинается ребро, и b — номер вершины, в которой заканчивается ребро ($1 \leqslant a \neq b \leqslant N$). Если граф неориентированный (т.е. $O = 1$), то это ребро двустороннее.

В графе могут быть кратные рёбра. Гарантируется, что в графе нет прямых рёбер между s и t.

Формат выходных данных

В первой строке нужно вывести целое число K — максимальное количество вершинно непересекающихся путей из s и t. Далее должно быть K блоков, каждый из который описывает отдельный путь.

В первой строке каждого блока должно быть целое число Q — количество рёбер в пути. В следующих Q строках должны быть описаны эти рёбра в порядке их следования в пути в направлении от s к t. Каждое ребро описывается тремя целыми числами: u — номер начальной вершины, i — номер ребра и v — номер конечной вершины ($1 \leqslant u \neq v \leqslant N$, $1 \leqslant i \leqslant M$). Рёбра графа нумеруются в порядке их описания во входных данных. В неориентированном графе вершины u и v нужно выдавать так, что путь проходит через ребро uv в направлении от u к v.

Пример

input.txt	output.txt
12 19 1 12 11 12 1 12 2 12 3 12 4 1 5 1 6 6 2 2 7 3 7 3 8 4 7 5 11 5 9 6 9 7 10 8 10 8 11 9 11 10 11	3 3 12 1 1 1 5 5 5 12 11 4 12 2 2 2 8 7 7 15 10 10 19 11 3 12 3 3 3 10 8 8 17 11
4 7 0 1 4 1 2 2 4 4 3 3 1 3 2 3 2 2 3	2 2 1 1 2 2 2 4 2 1 4 3 3 3 4