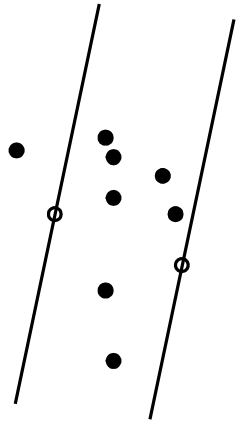
Задача A. Точки

Имя входного файла:	a.in
Имя выходного файла:	a.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	64 мегабайта

На плоскости задано N точек. Кроме того, на плоскости заданы две *базовые* точки.



Напишите программу, которая находит максимальное количество точек, которые попадут в *полосу* образованную парой параллельных прямых произвольно проведенных через базовые точки. Базовые точки не нужно включать в сумму точек. Если точка лежит на прямой – ее необходимо включить в сумму.

Формат входных данных

Первая строка входного файла содержит одно целое число N ($0 \le N \le 10\,000$) – количество точек. Вторая строка содержит координаты двух базовых точек в формате $x_1\ y_1\ x_2\ y_2$. Каждая из последующих N строк содержит координаты точки плоскости в формате $x\ y$. Координаты точек – целые числа, по модулю не превышающие 10 000. Базовые точки отличаются, по крайней мере, одной координатой.

Формат выходных данных

Единственная строка выходного файла должна содержать целое число – найденное максимальное количество точек, которые попадут в полосу, образованную оптимально проведенными параллельными прямыми через базовые точки.

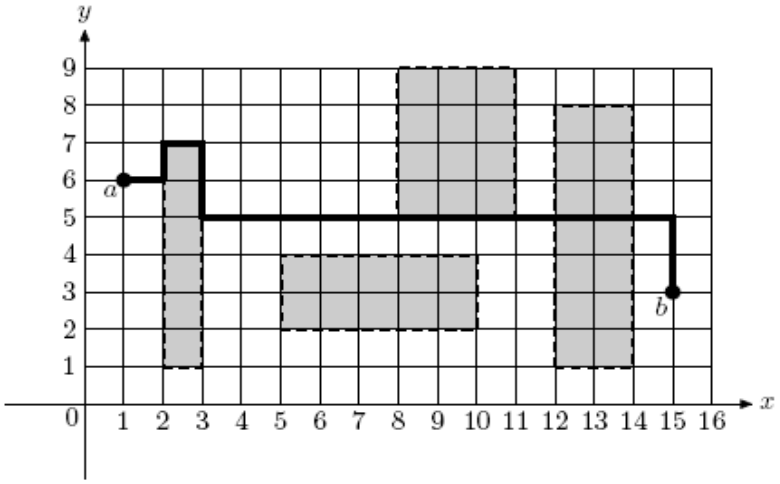
Пример

a.in	a.out
4 0 0 50 0 0 -50 -1 0 50 0 100 50	3

Задача B. Проезд через мегаполис

Имя входного файла:	b.in
Имя выходного файла:	b.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	64 мегабайта

Мегаполис будущего представляет собой прямоугольную сетку из улиц. Улицы параллельны осям координат и пересекаются в точках с целыми координатами x и y. Чтобы проехать от перекрестка A с координатами X_a, Y_a до перекрестка B с координатами X_b, Y_b необходимо проехать $|X_a-X_b| + |Y_a-Y_b|$ блоков. Обычно, время проезда одного блока составляет 10 минут, так что кто угодно способен быстро вычислить время своей поездки от A до B. Однако пробки, возникающие в мегаполисе, делают задачу вычисления минимального времени поездки очень сложной, ее предстоит решить вам.



Пробка в мегаполисе представляют собой прямоугольник, определенный координатами левого нижнего и правого верхнего угла. Время проезда одного блока в пробке задается для каждой из пробок отдельно. Пробка существует на всех улицах, находящихся внутри прямоугольного блока. Иногда пробку быстрее объехать, но иногда лучше все же ехать через пробку. Оба этих случая представлены в примере и на рисунке.

Формат входных данных

Первая строка входного файла содержит четыре целых числа X_a, Y_a, X_b, Y_b – координаты начального и конечного перекрестков. Вторая строка содержит одно целое число N ($1 \le N \le 1000$) - количество пробок в мегаполисе. Следующие N строк описывают пробки. Каждая пробка описывается пятью числами: $X_{1,i}$, $Y_{1,i}$, $X_{2,i}$, $Y_{2,i}$ и T_i, где первые 4 числа — координаты левого нижнего и правого верхнего углов пробки, а T_i ($10 \le T_i \le 10^8$) – время проезда одного блока в этой пробке. Все координаты во входных данных могут принимать значения от 0 до 10^8 включительно. Области пробок не пересекаются и не касаются. Начальный и конечный перекрестки различны и не находятся ни в пробке, ни на ее границе.

Формат выходных данных

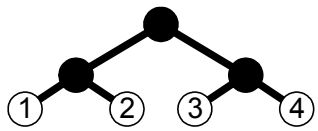
Выведите одно число — минимальное время поездки.

Пример

b.in	b.out
1 6 15 3 4 2 1 3 7 44 5 2 10 4 33 8 5 11 9 22 12 1 14 8 11	192

Задача C. Катакомбы

Имя входного файла: c.dXX
Имя выходного файла: c.sXX



Пиратские катакомбы на острове Сокровищ были вырыты по следующему принципу. После скрытого входа расположена пещера, из которой выходят два туннеля – налево и направо. Каждый из туннелей заканчивается пещерой, из которой так же выходит два туннеля, и т.д. Длина каждого туннеля равна единице. *Конечные* пещеры, которые находятся на расстоянии D от входа, не имеют дальнейших выходов. Никакие туннели между собой не пересекаются, и не ведут к одной пещере. Число D называют *глубиной* катакомб.

В каждой из конечных пещер спрятан один *сундук с сокровищами*. Перед прибытием на остров Капитана Джека Воробья пираты решили переместить эти сундуки в соответствии с последними указаниями Капитана. Пираты нарисовали план катакомб и пронумеровали конечные пещеры слева направо. Потом для каждого сокровища был установлен номер пещеры, в которой он должен оказаться перед прибытием Капитана. После перемещения в каждой пещере снова окажется только один сундук.

Чтобы обеспечить безопасность сокровищ, пираты могут только обменивать между собой сундуки из двух пещер. Только после окончания одного обмена можно начинать другой. Необходимо найти наименьшее суммарное расстояние, которое пиратам необходимо будет нести сундуки, чтобы разместить их требуемым образом.

По предоставленным входным файлам, которые содержат описание пещеры с сокровищами, создайте соответствующие выходные файлы, которые содержат минимальное суммарное расстояние, которое пиратам придется нести сундуки, и последовательность обменов.

Формат входных данных

Вам предоставлено 10 входных файлов, которые имеют названия C.D01, C.D02,…, C.D10, в таком формате. Первая строка содержит одно целое число D – глубину катакомб. Вторая строка содержит 2^D разных целых чисел от 1 до 2^D. Каждое i-ое из них идентифицирует номер пещеры, в которую должен попасть сундук, находившийся сначала в пещере i.

Формат выходных данных

Создайте 10 файлов C.S01,…, C.S10. Эти файлы должны содержать ответы для соответствующих входных файлов. Первая строка файла должна содержать единственное целое число – минимальное суммарное расстояние, которое пройдут пираты с сокровищами. Вторая строка: целое число K – соответствующее количество обменов. Каждая последующая из K строк: два числа, которые есть номерами *пещер*, между которыми производится обмен. Обмены должны быть указаны в том порядке, в котором они должны производиться.

Пример

c.d00	c.s00
2 4 3 1 2	20 3 3 4 1 4 3 2

Например, можно проводить обмены таким образом. Сначала поменять местами сокровища в пещерах 3 и 4. Пройденное расстояние 4 (по 2 для каждого сундука). Потом поменять сокровища в пещерах 4 и 1, и 3-2. Расстояние в обоих случаях – 8. Таким образом – все встанут на свои места, а суммарное расстояние будет 20.