Задача A. Футбол, визы и TopCoder

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

Решение о том, что чемпионат Европы по футболу 2012 года проводят Польша и Украина, было принято как раз во время отбора на TopCoder Open — 2007.

Двое участников, собирающиеся на онсайт-раунд TCO-2012, но пока что не оформившие документы, запланировали в тот день идти в американское посольство за визой. Однако, услышав по радио новость про чемпионат Европы, каждый из них, будучи заядлым болельщиком, решил «сделать крюк» и перед посещением посольства зайти в туристические агенства — каждый в своё — и заранее заказать себе билеты на будущий чемпионат. Первый и второй участники вышли из своих домов, расположенных в точках B и D соответственно, одновременно. Первый участник направился в турагентство, расположенное в точке C, второй — в турагентство, расположенное в точке E. После посещения турагентств каждый участник направляется в посольство США, которое в целях уменьшения опасности террористической атаки находится в точке A с координатами (0,0). Считая, что между любыми двумя точками участники движутся по прямой, скорость их движения одинакова, а потеря времени в промежуточных точках нулевая (зашёл, оставил заявку и вышел), определить, кто из участников первым придёт в посольство.

Формат входного файла

Во входном файле заданы 8 целых неотрицательных чисел B_x, B_y, C_x, C_y, D_x, D_y, E_x, E_y, не превосходящих 30000 — координаты пунктов B, C, D, E соответственно.

Формат выходного файла

В выходной файл выведите одно слово FIRST, SECOND или BOTH (последнее — если оба участника пришли одновременно).

Пример

walk.in	walk.out
0 1 2 1 2 2 2 0	SECOND
2 3 2 0 4 1 0 1	BOTH
2 2 1 1 10 10 3142 2718	FIRST

Задача B. Хогвартс и TCHS

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

Во время трансляции очередного полуфинала TopCoder High School в Хогвартсе неожиданно отключился Интернет. Это было тем более обидно, что впервые за всю историю Хогвартса ученик этой школы вышел в финал личного соревнования по программированию. Однако присутствовавший на соревнованиях в качестве зрителя Гарри Поттер сумел наладить по магическим каналам передачу Рону Уизли информации о результатах участников по каждой из задач. К сожалению, Гарри, не очень хорошо разобравшийся в формуле проведения соревнований, забыл указать стоимость каждой задачи. Вам необходимо написать программу, которая из колонки результатов участников по одной задаче вычисляет возможные значения стоимости этой задачи, или сообщает, что таких значений нет (при передаче вполне может произойти сбой — Вольдеморт не дремлет).

Согласно правилам TopCoder, по задаче стоимостью M (M — натуральное число, кратное 50 и не превосходящее 1000) возможно набрать любое количество очков, меньшее M и не меньшее $0.3M$, если задача решена, и 0, если задача не решена. От Вас требуется вычислить, сколько очков могла бы стоить задача, результаты участников по которой Вам даны, или определить, что данные противоречивы.

Формат входного файла

В первой строке входного файла задано неотрицательное число $N < 10^5$ — количество участников, по которым у Вас есть информация. Далее в последующих N строках идут результаты участников, по одному в строке. Каждый результат представляет собой неотрицательное вещественное число, не превосходящее 1000 и заданное с 2 знаками после десятичной точки.

Формат выходного файла

В выходном файле выведите в порядке возрастания все возможные для данного набора результатов значения M. В случае, если таких значений нет, выведите строку «Impossible».

Пример

tchs.in	tchs.out
4 201.00 123.45 0 90.00	250 300
2 1000.00 1.00	Impossible

Задача C. Таблицы Руматы Эсторского — ЭКСПЕРИМЕНТАЛЬНАЯ

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

Часто финалы TopCoder проходят в Лас-Вегасе. В качестве ознакомления с местной спецификой на этот раз участникам было предложено сыграть в покер на игровом автомате. Один из участников заявил, что у него с собой есть таблицы, которые по пришедшему раскладу определяют, какое количество карт необходимо поменять для максимизации ожидаемого выигрыша (то есть для максимизации суммы по всем исходам произведений выигрыша в каждом исходе на вероятность данного исхода). Таблицы эти приписываются легендарному Румате Эсторскому, который с их помощью регулярно обыгрывал других благородных донов. Проверить огромные таблицы «на глазок» трудно даже участнику онсайт-раунда TopCoder, а вот подсчитать количество раскладов, в которых необходимо менять определённое число карт, вполне возможно. Именно этим Вам и предстоит заняться.

Установленные в данном казино автоматы устроены следующим образом.

Используется стандартная колода из 52 карт (4 масти по 13 карт в каждой масти, внутри каждой масти карты по достоинству расположены в порядке возрастания следующим образом: 2,3,4,5,6,7,8,9,10,валет,дама,король,туз).

Сначала на руки игроку случайным образом сдаётся расклад из 5 карт. Игрок может поменять от 0 до 5 карт на соответствующее количество карт, которые выбираются случайным образом из оставшейся колоды. Полученный в результате расклад оценивается так:

- Если в раскладе есть 2 валета, 2 дамы, 2 короля или 2 туза, то выплачивается выигрыш за «пару».
- Если в раскладе есть 2 пары карт одного достоинства (например, две двойки и две тройки), то выплачивается выигрыш за «две пары».
- Если в раскладе есть 3 карты одного достоинства (например, три девятки), то выплачивается выигрыш за «тройку».
- Если в раскладе 5 карт, идущих подряд, например: (2 червей, 3 треф, 4 червей, 5 пик, 6 бубен) или (8 пик, 9 пик, 10 пик, валет червей, дама треф), то выплачивается выигрыш за «стрит». При этом тузы могут считаться либо как карта, идущая за королём (как обычно), либо как карта, идущая перед двойкой, но не одновременно (то есть «циклического» стрита не бывает).
- Если в раскладе все 5 карт принадлежат к одной масти (например, тройка, семёрка, туз, валет и дама пик), то выплачивается выигрыш за «флеш».
- Если в раскладе 3 карты одного достоинства и 2 карты другого (например, три валета и две тройки), то выплачивается выигрыш за «фулл хаус».
- Если в раскладе 4 карты одного достоинства (например, все четыре туза), то выплачивается выигрыш за «каре».
- Если в раскладе выполняются условия для «стрита» и «флеша» (например, 7 пик, 8 пик, 9 пик, 10 пик, валет пик), то выплачивается выигрыш за «флеш-стрит».
- Если расклад состоит из туза, короля, дамы, валета и десятки одной масти, то выплачивается выигрыш за «флеш-рояль».

При этом всегда выплачивается только один выигрыш — за наиболее дальнюю в списке комбинацию (например, при трёх двойках и двух тузах выплачивается только выигрыш за «фулл хаус», но не выигрыш за «тройку», «пару» или «две пары»).

Если в раскладе нет ни одной из присутствующих в списке комбинаций, то выигрыш не выплачивается.

Автоматы различаются системами выплат — векторами из 9 целых чисел, задающими коэффициент выигрыша за «пару», «две пары», «тройку», «стрит», «флеш», «фулл хаус», «каре», «флеш-стрит» и «флеш-рояль». Всего бывает 11 типов автоматов: с векторами (1,2,3,4,6,9,25,50,800), (1,2,3,4,6,9,25,50,940), (1,2,3,4,5,9,25,50,800), (1,2,3,4,6,8,25,50,800), (1,2,3,4,5,8,25,50,800), (1,2,3,4,5,7,25,50,800), (1,2,3,4,5,6,25,50,800), (1,2,2,4,6,9,30,125,1000), (1,2,2,4,6,9,30,100,1000), (1,2,3,4,5,6,25,50,1000), (1,2,2,4,6,9,30,100,500).

Ваша задача — по заданной системе выплат (одной из перечисленных 11) вывести количество раскладов, в которых наибольший ожидаемый выигрыш достигается без обмена карт, с обменом одной карты, с обменом двух карт, с обменом трёх карт, с обменом четырёх карт, с обменом пяти карт. Если в некотором раскладе для двух случаев с разным числом меняемых карт наибольший ожидаемый выигрыш одинаков, то выбирается вариант с меньшим числом оставляемых себе карт.

Формат входного файла

Во входном файле заданы 9 целых чисел, задающих коэффициенты за выигрыш по комбинациям в соответствии с условием задачи. Гарантируется, что наборы чисел будут соответствовать одному из описанных в условии типов автоматов.

Формат выходного файла

Шесть целых чисел, обозначающих количество раскладов, в которых для достижения наибольшего ожидаемого выигрыша необходимо менять 0 карт, менять одну карту, менять 2 карты, менять 3 карты, менять 4 карты и менять все 5 карт соответственно.

Пример

table.in	table.out
1 2 3 4 6 9 25 50 800	18864 292800 147528 1651440 403968 84360

Задача D. Памятные подарки

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

Дарёному коню в брюхо не смотрят...

Улисс

Корпорация «МИФ» стала спонсором очередного Single Round Match.

— Да, это не TopCoder Open, но это способ заявить о себе среди программистов. Мы должны запомниться не только суммой призовых: как раз этим участников TopCoder удивить стало сложно, но и призами, уникальными для каждого участника, — заявил Ааз.

— Участников же много... Откуда же взять уникальные призы? — поинтересовался Скив.

— А очень просто. Я уже заказал призы у одного умельца, который только недавно пришёл на Базар-На-Деве. Призы представляют собой полый куб со стороной N, грань которого состоит из N^2 маленьких квадратиков. Каждый квадратик может быть или золотым, или серебряным, или просто содержать инкрустацию с логотипом нашей корпорации. При этом для всех граней всей партии призов логотипы размещены на одних и тех же клетках — для запоминаемости.

— На одних и тех же — это как?

— Разумеется, с точностью до поворота грани. Если тебе и это непонятно — объясняю: для любых двух граней любых двух кубиков в заказе существует поворот одной из граней, при котором положение клеток с нашим логотипом на обеих гранях совпадает. Вы спросите, а в чём же уникальность такого приза? Отвечаю — в комбинации золотых и серебряных пластин на клетках, оставшихся незаполненными логотипами нашей компании. Два разных кубика из одной партии невозможно совместить, как их ни крути.

— А как доставлять призы участникам? Если кубик целиком — то придётся возить воздух со стороной N, — поинтересовался Гвидо.

— И тут всё продумано, — ответил Ааз. — Для удобства перевозки кубик можно разобрать на отдельные грани, запаковать, а потом собрать произвольным образом, но так, чтобы ни для какой грани внутренняя (то есть обращённая изначально к центру кубика) сторона не стала внешней (то есть видной снаружи при полностью собранном кубике).

— А не получится так, что два участника получат такие наборы, покрутят-покрутят грани, да и соберут два одинаковых кубика? — спросил Скив.

— Если кто-то думает, что я не продумываю всё до конца, то он путает меня с кем-то другим. Согласно заказу, два кубика считаются существенно различными, только если один из них нельзя получить из другого с помощью описанной выше операции демонтажа-монтажа.

— Так сколько всего будет таких кубиков? — поинтересовался Нунцио.

— Сейчас наши программисты подсчитают и выдадут максимальный объём партии. Я понимаю, что не во всех измерениях так хорошо умеют работать с большими числами, как на Извре, так что ответ они выдадут по модулю... ну, скажем, $10^9 + 7$.

В итоге решить данную задачу было поручено Вам.

Формат входного файла

В первой строке входного файла задано одно число $1 \leqslant N \leqslant 1000$ — размер кубика в единичных клетках. Далее, в последующих N строках длины N, задано расположение логотипов корпорации «МИФ» на грани кубика: 1, если соответствующая клетка занята логотипом, и 0, если на клетке располагается золотая или серебряная пластина.

Формат выходного файла

Одно число P — количество существенно различных призов для данного N, взятое по модулю $10^9 + 7$.

Пример

souvenir.in	souvenir.out
3 1 0 0 1 1 1 0 0 1	5005

Задача E. Муки творчества

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

Журналисту одного из популярных спортивных изданий Берляндии поручили написать статью про недавно прошедший SRM, в котором берляндские участники заняли несколько мест в первой десятке, установив новый национальный рекорд. Дойдя до фразы «Борьба была...» журналист посмотрел в таблицу и задумался.

Таблица результатов SRM состояла из четырёх колонок. Для каждого участника были заданы результаты по каждой из трёх задач и итоговый результат. Итоговый результат участника складывается из суммы очков, набранных по задачам, и очков, набранных в челлендж-фазе. Во время челлендж-фазы участники могут просматривать решения друг друга. В случае, если участник считает, что решение другого участника неверно, он может сделать челлендж — то есть предложить тест, на котором, по его мнению, решение сработает неверно. Если челлендж оказался успешным (на данном тесте решение выдало неверный результат), то результат автора данного решения по соответствующей задаче становится равным 0, а к результату автора теста прибавляется 50 очков. Если челлендж оказался безуспешным, результат автора решения не изменяется, а от результата автора теста отнимается 25 очков. При этом 0 очков по задаче можно получить и на другой стадии (например, не отослав решения по задаче). Также считается, что каждый участник может сделать челлендж на решение любого другого участника (то есть все участники находятся в одной «комнате»).

Для написания статьи журналисту хочется знать максимальное и минимальное количество челленджей, которое возможно для заданной таблицы результатов.

Формат входного файла

В первой строке входного файла задано неотрицательное целое число $N \leqslant 10^5$ — количество участников SRM. Далее в последующих N строках идут результаты участников по окончании SRM, заданные четырьмя числами: результатом по первой, второй и третьей задачам соответственно и итоговой суммы очков, набранной участником. Результат по каждой из задач задан неотрицательным вещественным числом, не превосходящим 1000, заданым с 2 знаками после десятичной точки. Сумма, набранная участником, задана вещественным числом, по модулю не превосходящим 3000 и заданным с 2 знаками после десятичной точки.

Формат выходного файла

В выходной файл выведите два целых неотрицательных числа P и Q — наименьшее и наибольшее возможное количество сделанных в данном SRM челленджей соответственно.

Пример

report.in	report.out
2 100.00 200.00 300.00 600.00 100.00 200.00 0.00 300.00	0 3
2 100.01 200.02 300.03 650.06 200.00 300.00 0.00 500.00	1 1

Задача F. Нестабильность

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

Один из постоянных участников SRM, выступающий крайне нестабильно, решил проанализировать причины своих неудач. Исследовав ряд факторов, он предположил, что его результат зависит от функция $f(x)$, которая представляет собой сумму N слагаемых типа $sin(a_i * \pi * x)$, где все a_i — целые числа. Для того, чтобы проверить своё предположение, участник попросил Вас найти минимальный период этой функции по заданным a_i.

Формат входного файла

В первой строке входного файла задано число $1 \leqslant N \leqslant 100000$ — количество функций в сумме. Во второй строке идут N целых чисел a_i, по модулю не превосходящих 1000 — коэффициенты для каждой функции. При этом хотя бы одно из чисел a_i не равно нулю.

Формат выходного файла

Минимальный период функции $f(x)$ с 3 точными знаками после десятичной точки.

Пример

period.in	period.out
3 1 2 3	2.0000
1 3	0.66666

Задача G. Зацепление

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

Он плоский, как куб...

четырёхмерная поговорка

Перед Single Round Match, спонсируемым какой-нибудь компанией, обычно проходит общение с представителем компании-спонсора. Так что перед SRM, спонсируемым корпорацией «МИФ», надо было выбрать одного из членов корпорации, который бы сумел адекватно рассказать о задачах, которые могут стоять перед программистами, работающими на корпорацию «МИФ». По предложению Ааза представителем был назначен именно он.

И вот открылся чат. Ааз довольно бойко рассказал о перспективах сочетания магии и современных компьютерных технологий, о возникающих перед корпорацией задачах прогнозирования в экономике, да и о финансовом могуществе корпорации. Когда он коснулся «самой актуальной задачи — автоматизации работы с измерениями», один из участников задал вопрос по поводу примера задач, возникающих в этом случае. Ааз задумался на минуту и выдал следующую задачу: в 4-мерном пространстве расположены поверхность 3-мерного тетраэдра, заданного своими вершинами, и окружность, заданная тремя точками. Требуется узнать, зацеплены ли они друг за друга. Подумав ещё какое-то время, он объявил, что желающие работать в корпорации должны решить эту задачу в качестве тестовой, и решать её они будут наравне с нынешними программистами корпорации «МИФ».

Так что Вам тоже придётся решать эту задачу. Хорошо хоть, что ограничения для Вас выбраны не изврские: гарантируется, что тетраэдр и окружность невырождены, окружность не лежит в гиперплоскости тетраэдра и что окружность и поверхность тетраэдра не имеют общих точек.

Напомним, что окружность и поверхность тетраэдра считаются зацепленными, если нельзя непрерывно переместить окружность в бесконечность, таким образом, чтобы она во время перемещения ни в какой момент не имела общих точек с поверхностью тетраэдра.

Формат входного файла

В первых трёх строках входного файла заданы по четыре числа — координаты трёх точек на окружности (в каждой строке расположены координаты одной точки в порядке x, y, z, t), в следующих четырёх строках в таком же формате заданы координаты вершин тетраэдра. Координаты всех семи точек заданы не более, чем с 3 знаками после десятичной точки и по модулю не превосходят 1000.

Формат выходного файла

В выходной файл выведите слово YES, если заданные поверхность тетраэдра и окружность зацеплены, и NO, если они не зацеплены.

Пример

zacepl.in	zacepl.out
1 1 1 0 -1 -1 -1 0 0 0 0 1.732 0 0 0 0 5 0 0 0 0 5 0 0 0 0 5 0	YES

Задача H. Заявки на билеты

Имя входного файла:	`tickets.in`
Имя выходного файла:	`tickets.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайт

Получилось так, что финал очередного онсайт-соревнования TopCoder оказался расположен близко по времени к розыгрышу Кубка Векуа. Организаторы Кубка приняли решение централизованно закупить билеты для делегаций, отправляющихся на онсайт-соревнование. К сожалению, удобных прямых рейсов в день отъезда нет, и участники приняли решение лететь с пересадкой.

Из аэропорта Тбилиси в день отъезда участников Кубка Векуа летят рейсы по N направлениям, подходящим для последующей пересадки к месту проведения онсайт-финала. Рейсы обслуживаются лайнерами максимальной вместимости k. При этом получилось так, что расстояние до конечных пунктов примерно одно и то же, и затраты авиакомпании на рейс по любому направлению одинаковы и равны s. Цена билета на любой рейс по i-му направлению равна c_i, при этом цены билетов подобраны так, что при половинной заполняемости самолёта ($k/2$ для чётных k и $(k-1)/2$ для нечётных) авиакомпания как минимум компенсирует затраты на рейс. В авиакассы на этот день поступило l заявок от делегаций, участвующих в Кубке. В j-й заявке указан номер p_j направления, по которому летит делегация и количество q_j участников в делегации. При этом отправлять всю делегацию одним рейсом не обязательно.

Авиакомпания имеет достаточное количество самолётов, чтобы удовлетворить все заявки, назначив требуемое количество рейсов по нужным направлениям. Однако руководство авиакомпании решило максимизировать прибыль, полученную от перевозок в этот день, передав часть поданных заявок другим авиакомпаниям. Ваша задача — написать программу, которая бы определяла, какие заявки нужно передать конкурентам для того, чтобы максимизировать прибыль, полученную компанией от перевозок пассажиров в этот день (сумма проданных по заявкам билетов минус суммарные затраты на все рейсы). Заявки можно передавать другой авиакомпании только полностью; заполнять самолёты сверх максимальной вместимости нельзя.

Формат входного файла

В первой строке входного файла заданы 4 целых числа: N, l, k, s ($1 \leqslant N \leqslant 100$, $2 \leqslant l \leqslant 1000000$, $2 \leqslant k \leqslant 400$, $1 \leqslant s \leqslant 1000$), где N — количество направлений, l — количество поданных заявок, k — количество посадочных мест в самолёте, s — затраты на один рейс. Во второй строке идут N целых чисел c_i — цены билетов на i-е направление. При этом каждое из c_i, с одной стороны, не превосходит s, а с другой — подобрано так, чтобы компания не несла убытков при половинной заполняемости самолёта. В следующих l строках заданы заявки на билеты от делегаций. В $j+2$-й строке входного файла j-я заявка задана двумя целыми числами $1 \leqslant p_j \leqslant m$ — номером направления, по которому летит j-я делегация и $1 \leqslant q_j \leqslant 1000$ — количество участников j-й делегации.

Формат выходного файла

В выходной файл выведите одно целое число — требуемую максимальную прибыль авиакомпании от перевозки пассажиров по данным заявкам с учётом возможной передачи части заявок конкурентам.

Пример

tickets.in	tickets.out
3 2 10 30 7 10 8 1 9 3 13	77

Задача I. Побег

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

Однажды журналист одной популярной газеты, написавший статью про результаты TopCoder Collegiate Challenge и «чудо-программисте из МГУ», содержавшую крайне мало фактов и чересчур много домыслов по поводу «хакеров» и «вирусов», случайно зашёл в Главное Здание МГУ и был там опознан участниками соревнований по спортивному программированию.

После небольшой гонки журналист оказался на шпиле Главного Здания МГУ, расположенном в точке A. Путь к отступлению был отрезан разъярёнными программистами... И тут журналист заметил, что там и тут в воздухе летают утки. А, как известно, журналист популярной газеты может легко прыгнуть на утку и чувствовать себя там, как на твёрдой почве. Журналист может перепрыгнуть с башни или утки на другую башню или утку, если расстояние между ними не превосходит предельной длины его прыжка L. Утки летают примерно на одной и той же высоте, так что перепадами высоты можно пренебречь. Цель журналиста — добраться до Останкинской башни, расположенной в точке B. Утки могут летать по прямой или висеть неподвижно (это такая особенная порода). Кривизной территории Москвы, а также линейными размерами башен и уток можно пренебречь. Скорость прыжка бесконечно большая. Между двумя прыжками должно проходить не менее t секунд, чтобы журналист успел отдохнуть.

Требуется определить, при каком минимальном L (предельной длине прыжка) журналист сможет добраться до Останкинской башни.

Формат входного файла

В первой строке входного файла задано $0 \leqslant t \leqslant 100$ — время, требуемое журналисту на отдых, заданное с 3 знаками после десятичной точки. Во второй — четыре числа, по модулю не превосходящие 1000 и заданные максимум с 3 знаками после десятичной точки: координаты шпиля Главного Здания МГУ $A_x,\ A_y$ и Останкинской телебашни $B_x,\ B_y$ соответственно. В третьей строке задано целое число $0 \leqslant N \leqslant 100$ — количество уток. Далее, в последующих N строках, для каждой из уток указаны 4 числа: начальное положение утки $-1000 \leqslant x_i,\ y_i \leqslant 1000$, заданное с 3 знаками после десятичной точки и координаты вектора её скорости $-100 \leqslant V_{ix},\ V_{iy} \leqslant 100$, заданные с 4 знаками после десятичной точки.

Формат выходного файла

В выходной файл выведите минимальную предельную длину прыжка с точностью 0.001.

Пример

run.in	run.out
0 0 0 10 10 1 -5 -5 1 1	0.000
25 0 0 10 10 1 -5 -5 1 1	14.142