Задача 1 Реклама.

В одному гіпермаркеті вирішили час від часу транслювати рекламу нових товарів. Для того, щоб скласти оптимальний розклад реклами, керівництво гіпермаркету наказало провести дослідження: на протязі дня для кожного покупця, який відвідав гіпермаркет, зафіксувати час, коли він зайшов до гіпермаркету та час, коли він звідти вийшов.

В припущенні, що такий розклад відвідування збережеться і надалі, менеджеру по рекламі було доручено скласти розклад трансляції реклами з дотриманням таких вимог:

1. кожен відвідувач має почути рекламне оголошення не менше двох разів;
2. два оголошення не можуть транслюватись одночасно;
3. постільки продавці вимушені постійно чути рекламні оголошення, то загальна
кількість оголошень повинна бути мінімальною.

Необхідно допомогти менеджеру та розробити програму для складення розкладу трансляції рекламних роликів. Рекламні оголошення можна починати транслювати тільки в цілі моменти часу. Рахується, що кожен рекламний ролик завершується до початку наступного цілого моменту часу. Якщо рекламне оголошення транслюється в момент часу, коли покупець входить в гіпермаркет або з нього виходить, то покупець встигає почути це оголошення.

**Вхідні дані.**

Перший рядок вхідного файлу REKLAMA.IN містить число N - кількість відвідувачів за день (1 < N < 3000). Наступні N рядків містять пари натуральних чисел аі, В] , що вказують, відповідно, на момент часу, коли 1-ий покупець зайшов в гіпермаркет та момент часу, коли він вийшов з гіпермаркету (0 < аі < ВІ< 100).

**Вихідні дані.**

Перший рядок вихідного файлу REKLAMA.OUT повинен містити отриману кіль­кість рекламних роликів. Далі - в зростаючому порядку моменти часу, коли потрібно транслювати рекламні оголошення. Якщо є декілька рішень, вказати довільне.

**Приклад.**

REKLAMA.IN

5

1 10

1012

1 10

1 10

2324

REKLAMA.OUT

5

5 10 12 23 24

2.Точки (2011)

На площині задано N точок.

Завдання Напишіть програму points, яка знайде суму квадратів відстаней між всіма парами точок.

Вхідні дані Перший рядок вхідного файлу points.dat містить єдине натуральне число N (1 ≤ N ≤ 100 000) – кількість точок. Наступні N рядків містять по два цілих числа X і Y (‑10 000 ≤ X, Y ≤10 000) – координати точок.

Вихідні дані Єдиний рядок вихідного файлу points.sol має містити єдине ціле число – суму квадратів відстаней між всіма парами точок.

Оцінювання Щонайменше у 50% тестів будуть виконуватись додаткові обмеження 1 ≤ N ≤ 1000, ‑1000 ≤ X, Y ≤1000. Щонайменше у 70% тестів будуть виконуватись додаткові обмеження 1 ≤ N ≤ 20 000.

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| points.dat | points.sol |
| 41 1-1 -11 -1-1 1 | 32 |

**3. Прямокутники (1977)**

На площині задано многокутник. Треба написати програму *RECT*.\*, що визначає прямокутник мінімальної площі, який містить в собі заданий многокутник. Наприклад, для многокутника:

 відповідним прямокутником буде:

Вхідні дані: Вхідний файл *RECT.DAT* містить в 1-му рядку ціле число *N* - кількість вершин многокутника (3 ≤ *N* ≤ 3000), в наступних *N* рядках - по два дійсних числа *X*i,*Y*i - координати вершин многокутника в порядку їх обходу за годинниковою стрілкою.

Вихідні дані: Вихідний файл *RECT.SOL* повинен містити 5 рядків: в першому рядку число *S* - площа прямокутника, а в наступних 4-х рядках - пари координат *X*i *Y*i вершин прямокутника в порядку їх обходу (в довільному напрямку).

Технічні умови.

1. Всі координати у вхідному і вихідному файлах подаються у вигляді дійсних чисел в форматі, що обробляється стандартними функціями введення-виведення.

2. Рекомендований тип даних для координат - *Real* в Pascal і *float* в C та C++.

3. Оптимальну площу і координати прямокутника треба визначити з точністю до 10-5.

Приклад введення і виведення

|  |  |
| --- | --- |
| *RECT.DAT* | *RECT.SOL* |
| 6 | 32.0 |
| 0.0 0.0 | 4.0 4.0 |
| 3.0 2.0 | 0.0 8.0 |
| 4.0 4.0 | 4.0 -4.0 |
| 5.0 2.0 | 0.0 0.0 |
| 8.0 0.0 |  |
| 4.0 1.0 |  |

4. Багатокутники (2002)

На площині задана така множина з *N* багатокутників, що виконуються наступні умови:

1. ніякі два багатокутника не мають спільних точок;
2. для кожного *i*–го багатокутника існує *Pi* багатокутників, всередині яких він знаходиться, і
*N*-1-*Pi* багатокутників, котрі знаходяться всередині нього, 0≤*Pi*≤*N*-1.

Завдання

Напишіть програму POLYGON, яка для кожного багатокутника видає кількість багатокутників, всередині яких він знаходиться.

Вхідні дані

Перший рядок вхідного файлу POLYGON.DAT містить ціле число *N* — кількість багатокутників, 3≤*N*≤10000. Наступні *N* рядків файлу описують *N* багатокутників. (*i*+1)–ий рядок файлу описує *i*–ий багатокутник. Перше ціле число *Ci* — кількість вершин багатокутника, 3≤*Ci*≤20. Наступні *Ci* пар чисел — координати вершин багатокутника у порядку його обходу. Координати вершин — цілі числа, що належать діапазону від -2 000 000 000 до 2 000 000 000.

Вихідні дані

Єдиний рядок вихідного файлу POLYGON.SOL повинен містити N чисел: *i*–те число рядка повинно бути *Pi* — кількість багатокутників, всередині яких знаходиться *i*–ий багатокутник.

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| POLYGON.DAT | POLYGON.SOL |
| 33 -2 1 8 9 12 13 7 5 6 3 7 44 4 3 7 7 9 3 1 2 | 0 2 1 |

4. Зала Круглих Столів (2004)

Єдиний спосіб потрапити до Зали Круглих Столів – пройти через Колонний Коридор. Стіни Коридору зображаються на карті прямими лініями, які паралельні вісі OY системи координат. Вхід в коридор знаходиться знизу, а вихід з Коридору до Зали – зверху. В Коридорі є циліндричні (на карті круглі) Колони однакового радіуса R.

Завдання

Напишіть програму TABLE, що за інформацією про розміри Коридору, та розміщення Колон визначає діаметр найбільшого з Круглих Столів, який можна пронести через такий Коридор, зберігаючи поверхню Стола горизонтальною.

Вхідні дані

У першому рядку вхідного файлу TABLE.DAT задані два числа XL та XR – x-координати лівої та правої стін Коридору. У другому рядку знаходиться ціле число R (1≤R≤1 000 000) – радіус усіх Колон. У третьому – ціле число N (1≤N≤200), що задає кількість Колон. Далі йдуть N рядків, в кожному з яких по два числа – x- та y-координати центра відповідної Колони.

Всі вхідні координати – цілі числа, що за модулем не перевищують 1 000 000.

Вихідні дані

Єдиний рядок вихідного файлу TABLE.SOL має містити єдине число – шуканий діаметр найбільшого Столу. Діаметр потрібно виводити з точністю 3 знаки після десяткової крапки (навіть у випадку, якщо він виявиться цілим). Якщо не можна пронести жодного Столу, то відповідь має бути: 0.000

Точність 3 знаки після крапки, за звичайними правилами заокруглення, означає, що відповідь, яка видається у вихідний файл, повинна відрізнятися від точної не більше ніж на 5⋅10-4 (тобто на 0.0005). Наприклад, якщо точна відповідь 1.234567, то у файлі повинно знаходитися число 1.235. Якщо точна відповідь 5.0005, то необхідно заокруглювати у більшу сторону, тобто у файл необхідно видати 5.001

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| TABLE.DAT | TABLE.SOL |
| 0 903410 1070 1050 5010 90 | 47.000 |

**5.Скелі (2005)**

На планеті Олімпія робітники будують нову дамбу. Частина площини, де проводяться будівельні роботи, має вигляд прямокутника розміром 1 x L метрів, на якому введено координати, як показано на рисунку.

Для підняття ландшафту використовуються спеціально розроблені магічні імпульсатори. Якщо магічний імпульсатор з силою H поставити в точку з *X*-координатою p, то в кожній точці q відрізку [p–H;p] на вісі *X* рельєф піднімається на q–p+H метрів по всій його ширині (тобто для довільного Z від 0 до 1), а в кожній точці q відрізка [p;p+H] рельєф піднімається на H+p–q метрів по всій його ширині, в інших точках ландшафт залишається незмінним (див. рисунок).

Під час будівництва робітники час від часу цікавляться об’ємом дамби, що знаходиться вище деякого прямокутника.

Завдання

Напишіть програму ROCKS, що допоможе робітникам у їхніх обрахунках.

Вхідні дані

В першому рядку вхідного файлу ROCKS.DAT містяться два цілих числа: N – кількість операцій, що будуть виконувати робітники (1≤N≤100000), та L – довжина прямокутника (1≤L≤100000).

В наступних N рядках містяться описи операцій: перше число рядка – номер операції, де „1” означає, що робітники збираються поставити магічний імпульсатор, „2” – робітники бажають дізнатися деякий об’єм. Якщо операція має код „1”, то далі йдуть два цілих числа p та H (0≤p≤L; 1≤H≤L), тобто імпульсатор сили H ставлять на позицію p (на вісі X). Якщо операція має код „2”, то далі йдуть два цілих числа A та B (0≤A<B≤L); це означає, що робітники бажають дізнатися об’єм частини дамби, яка знаходиться над прямокутником від A до B по вісі X, та від 0 до 1 по вісі Z.

Вихідні дані

Створіть вихідний файл ROCKS.SOL у якому для кожної операції, вказаної у вхідному файлі, виведіть рядок з наступною інформацією.

Якщо операція є „1”, то виведіть число „-1” без лапок. Якщо операція є „2”, то виведіть число, яке дорівнює об’єму частини дамби, що знаходиться над прямокутником від A до B по вісі X, та від 0 до 1 по вісі Z, як це показано на рисунку.

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| ROCKS.DAT | ROCKS.SOL |
| 2 131 7 52 5 9 | -116 |

Розділення багатокутників (2007)

На площині задано дві фігури, що обмежені опуклими багатокутниками. Фігури розташовані таким чином, що їх вершини не співпадають, а контури мають рівно 2 точки перетину.

Довільним чином розділимо площину прямою. Будемо вважати, що півплощина з одного боку прямої відповідає першій фігурі, а з іншого боку – другій фігурі. Визначимо поняття дефекту розділення – сума площі першої фігури, що розташована на півплощині другої фігури, та площі другої фігури, що розташована на півплощині першої фігури. З двох можливих відповідностей півплощин до фігур оберемо таку відповідність, де значення дефекту менше.

Наприклад, на рисунку зображена пряма, що задає певне розділення багатокутників. Оцінка дефекту цього розділення (два випадки відповідності): (D)+(C+E) та (A+D)+(B+C). З рисунку, D+C+E менше, отже, загалом, оцінка розбиття дефекту розділення утвореного цією прямою є D+C+E.

Завдання

Напишіть програму DIVIDE, що за заданими двома багатокутниками знаходить пряму, що утворює розділення з найменшим дефектом.

Вхідні дані

Перший рядок вхідного файлу DIVIDE.DAT містить одне ціле число N (3≤N≤20) – кількість вершин першого багатокутника. Наступні N рядків містять пари цілих чисел – координати вершин першого багатокутника у порядку обходу. (N+2)-й рядок файлу містить число M (3≤M≤20) – кількість вершин другого багатокутника. Наступні M рядків містять його координати задані таким же чином, як і для першого багатокутника. Координати точок додатні та не перевищують 1000.

Вихідні дані

Єдиний рядок вихідного файлу DIVIDE.SOL має містити дві пари чисел – координат двох точок, що однозначно задають розділення (пряму) з найменшим дефектом. Числа потрібно виводити у порядку: x1 y1 x2 y2. Координати потрібно виводити з точністю 10-3. Координати точок мають бути додатні та не більші за 1000.

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| DIVIDE.DAT | DIVIDE.SOL |
| 32 13 34 135 23 24 3 | 2 5 4 1 |

6. Точки (2008)

На площині задано N точок. Окремо на площині задані дві базові точки.

Завдання

Напишіть програму POINTS, що знаходить максимальну кількість точок, що потраплять у смугу створену парою паралельних прямих довільно проведених через базові точки. Базові точки не потрібно включати до суми точок. Якщо точка лежить на прямій – її потрібно врахувати у сумі.

Вхідні дані

Перший рядок вхідного файлу POINTS.DAT містить одне ціле число N (0≤N≤10 000) – кількість точок. Другий рядок містить координати двох базових точок у форматі x1 y1 x2 y2. Кожен з наступних N рядків містить координати точки площини у форматі x y. Координати точок – цілі числа, що за модулем не перевищують 10 000. Базові точки відрізняються принаймні однією координатою.

Вихідні дані

Єдиний рядок вихідного файлу POINTS.SOL має містити ціле число – знайдену максимальну кількість точок, що потраплять у смугу, яка буде утворена оптимально проведеними паралельними прямими через базові точки.

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| POINTS.DAT | POINTS.SOL |
| 40 0 50 00 -50-1 050 0100 50 | 3 |

7. Кола (2009)

На площині задано N різних кіл. Два кола перетинаються, якщо мають хоча б одну спільну точку.

Завдання

Напишіть програму CIRCLES, що за координатами центрів кіл та їх радіусами знайде пару кіл, що перетинається.

Вхідні дані

В першому рядку вхідного файлу CIRCLES.DAT міститься ціле число N (1≤N≤10 000) . В кожному з наступних N рядків міститься три натуральних числа *X, Y, R* менших за 10 000, що задають координати центру кола *(X, Y)* та його радіус *R*.

Вихідні дані

Єдиний рядок вихідного файлу CIRCLES.SOL має містити пару номерів кіл, що перетинаються, або єдине число 0, якщо жодні два кола не перетинаються. Кола нумеруються відповідно до порядку у вхідному файлі, починаючи з 1 до *N*. Якщо існує декілька пар кіл, що перетинаються, виведіть будь-яку з них. Елементи пари можуть бути виведені в довільному порядку.

Приклад вхідних та вихідних даних

|  |  |
| --- | --- |
| CIRCLES.DAT | CIRCLES.SOL |
| 55 10 46 20 310 15 312 8 213 13 1 | 5 3 |

## Задача 7. Лесной пожар

*Максимальная оценка 35 баллов*

В МЧС поступило сообщение о возможном лесном пожаре в заданном квадрате тайги. Для поиска места возгорания было послано *N* самолетов. Однако ни один из экипажей пожар не обнаружил.

Известно, что с самолета видна полоса тайги, границы которой находятся на расстоянии 50 км справа и слева от той линии на поверхности Земли, над которой пролетает самолет (см. рисунок), причем точки, находящиеся на расстоянии ровно 50 км от этой линии, все еще видны. Донесение с каждого самолета содержало информацию о том, в каких двух различных точках (*xb*, *yb*) и (*xe*, *ye*) самолет входил в заданный квадрат и покидал его соответственно. Между этими точками самолет двигался строго по прямой.

**Требуется** написать программу, которая определит, действительно ли весь заданный квадрат тайги был просмотрен с самолетов. Если это не так, то программа должна находить координаты какой-нибудь точки, находящейся внутри или на границе квадрата и не попавшей ни в одну из просмотренных полос.

**Входные данные**

Входной файл с именем fire.in состоит из *N* +2 строк.

В первой строке записано натуральное число *L* - размер заданного квадрата тайги в километрах (0<*L<=*1000). Во второй строке - натуральное число *N* (1*<=N<=*100) - количество самолетов. В каждой из последующих *N* строк записано донесение с самолета - четыре вещественных координаты *xb*, *yb*, *xe*, *ye*. Координаты заданы в километрах. Стороны квадрата тайги параллельны осям координат, его левый нижний угол находится в точке с координатами (0, 0), а правый верхний - в точке (*L*, *L*).

**Выходные данные**

Выходной файл с именем fire.out должен содержать одну строку. Если заданный квадрат был просмотрен полностью, то эта строка должна состоять из слова OK, написанного заглавными латинскими буквами. В противном случае в этой строке должны быть записаны через пробел координаты *x* и *y* какой-либо точки, которая не попала ни в одну из просмотренных полос. Координаты нужно выводить в километрах с ошибкой не более одного метра\*).

Пример входного файла

245
1
26.1 0 193.568 245

Пример выходного файла для приведенного примера входного файла

155.123 100

\*) В одном километре 1000 метров. //примечание автора ;-)