**Перевезення**

|  |  |
| --- | --- |
| **Input file name:** | transportation.in |
| **Output file name:** | transportation.out |
| **Time limit:** | 100 ms |
| **Memory limit:** | 256 M |

З пункту ***A*** в пункт ***В*** необхідно перевезти ***X*** тонн вантажу, для чого в пункті ***A*** знаходяться 2 вантажівки. Перша вантажівка може везти ***V1*** тонн вантажу і долає шлях від ***A*** до ***B*** (або назад) за час ***T1***, другa, відповідно, ***V2*** тонн вантажу і за час ***T2***. За який мінімальний час вантажівки перевезуть вантаж в пункт ***B***? Завантаження та розвантаження здійснюються тільки в кінцевих пунктах.

**Вхідні дані:** У єдиному рядку вхідного файлу знаходяться п'ять цілих чисел ***X, V1, T1, V2, T2 (1 ≤ X, V1, T1, V2, T2 ≤ 231-1)***.

**Вихідні дані:** У єдиному рядку вихідного файлу виведіть одне число - мінімальний час, за який буде перевезено вантаж.

**Приклад вхідних і вихідних даних:**

|  |  |
| --- | --- |
| **transportation.in** | **transportation.out** |
| 10 2 3 5 4 | 12 |
| 100 2 3 5 4 | 105 |

**Підготовка до олімпіади**

|  |  |
| --- | --- |
| **Input file name:** | olympiad.in |
| **Output file name:** | olympiad.out |
| **Time limit:** | 100 ms |
| **Memory limit:** | 256 M |

Вимушені канікули Степана продовжуються, а тому підготовка до олімпіади триває. Сьогодні він зареєструвався на сайті з величезною кількістю різнотипних задач. Але особливість цього сайту в тому, що після реєстрації користувачеві недоступна жодна із задач. Проте відомо скільки нових задач стане відкрито користувачу на i-ий день після реєстрації.

Степан вирішив кожного дня розв’язувати по одній ще не розв’язаній задачі, звичайно, якщо така задача існує, інакше ж буде відпочивати. Якщо в якийсь із днів хлопцю доступні більше однієї задачі, то розв’язання деяких із них Степан перенесе на наступні дні, так, щоби кожного дня він розв’язував не більше однієї задачі, а переноси були як можна коротші.

Нехай, наприклад, на 5 день стануть доступні чотири нові задачі. Тоді одну з них Степан розв’яже п’ятого дня, а розв’язання інших трьох перенесе на 6, 7 і 8 дні. А якщо виявиться, що на 7 день стане відкритою ще одна задача, то її розв’язанням олімпіадник займеться тільки 9 дня.

Напишіть програму, яка, знаючи кількість нових задач, що стануть доступним у кожен з наступних ***N*** днів, визначить, в які дні Степан працюватиме, а в які відпочиватиме.

**Вхідні дані:** У першому рядку записане одне ціле число ***N*** – кількість днів, про які відомо, як працюватиме сайт.

У другому рядку знаходяться ***N*** невід’ємних цілих чисел – для кожного дня вказано, скільки нових задач стане доступно на сайті в цей день.
Гарантується, що ***1 ≤ N ≤ 100000***, і що сума всіх чисел у другому рядку не перевищує 100000.

**Вихідні дані:** Виведіть один рядок, що складатиметься із символі «+» або «-». Символом «+» позначайте день, в який Степан розв’язуватиме задачу, а «-» – відпочиватиме. Виведіть як мінімум ***N*** символів – по одному на кожен день, про який відома робота сайту. Але якщо розв’язання задач доводиться переносити на дні після ***N***-го (що дозволяється), то виведіть більшу кількість символів – до останнього дня, в який буде розв’язана задача. Символи розділяйте пропусками.

**Приклад вхідних і вихідних даних:**

|  |  |
| --- | --- |
| **olympiad.in** | **olympiad.out** |
| 50 3 0 0 0 | - + + + - |
| 100 4 0 2 0 0 0 0 1 0 | - + + + + + + - + - |
| 30 3 0 | - + + + |

**Спільний дільник**

|  |  |
| --- | --- |
| **Input file name:** | divisor.in |
| **Output file name:** | divisor.out |
| **Time limit:** | 100 ms |
| **Memory limit:** | 256 M |

Степан вивчив на уроці математики найбільший спільний дільник двох чисел. Степан дуже талановитий хлопчик і відразу зрозумів, що до чого. Він моментально порахував найбільший спільний дільник чисел ***A*** і ***B***. Вчителька математики одразу придумала для Степана нову задачу - для даних чисел ***A*** і ***B*** знайти таке додатне ціле число ***D***, що ***A*** ділиться на ***D***, ***B*** ділиться на ***D***, а сума цифр числа ***D*** максимальна.
Допоможіть Степану знайти спільний дільник чисел ***A*** і ***B*** з найбільшою сумою цифр.

**Вхідні дані:** У єдиному рядку вхідного файлу знаходяться два цілих числа ***A, B (1 ≤ A, B ≤ 109)***.

**Вихідні дані:** У єдиному рядку вихідного файлу виведіть спільний дільник чисел ***A*** і ***B*** з найбільшою сумою цифр. Якщо відповідей декілька, можете вивести будь-яку.

**Приклад вхідних і вихідних даних:**

|  |  |
| --- | --- |
| **divisor.in** | **divisor.out** |
| 10 20 | 5 |

**Щасливі квитки**

|  |  |
| --- | --- |
| **Input file name:** | tickets.in |
| **Output file name:** | tickets.out |
| **Time limit:** | 100 ms |
| **Memory limit:** | 256 M |

Степан працює головним економістом у фірмі, що володіє кількома маршрутками. Нещодавно в його фірмі був проведений ряд заходів щодо захисту квитків на маршрутки від підробок. Серед інших заходів було змінено систему номерів квитків. Тепер на кожному квитку надруковано ***N*** чисел, кожне в діапазоні від 1 до ***N***; тобто мається всього ***NN*** різних варіантів квитка.

В результаті автобуси цієї фірми стали користуватися ще більшою популярністю, ніж раніше. З метою отримати новий квиток в якості сувеніра люди стали їздити в цих маршрутках, навіть якщо їм треба було їхати в інший бік. Навіть у вихідні маршрутки стали ходити битком набитими. Як головний економіст фірми, Степан вирішив, що така популярність маршруток створює можливість отримання додаткового прибутку. У результаті були підвищені ціни на квитки, що, правда, не привело до серйозного зменшення пасажиропотоку.

Крім того, за повідомленнями кондукторів маршруток, серед пасажирів почалася боротьба за щасливі квитки. Правда, стандартні способи визначення "щасливості" квитка не працюють за нової системою номерів. Тому, спеціально за вказівкою Степана, кондуктори маршруток провели дослідження, яке показало, що пасажири придумали нове визначення щасливого квитка. А саме, якщо позначити числа, надруковані на квитку, як ***m(1), m(2), ..., m(N)***, то квиток щасливий тоді і тільки тоді, коли ***m(m(i)) = m(i)*** для кожного ***i*** від 1 до ***N***.

Наприклад, при ***N=2*** з 4 можливих квитків три є щасливими, і тільки квиток 2 1 - не щасливий (оскільки, наприклад, ***m(m(2))=m(1)=2 ≠ m(2)=1***); при ***N=4***, наприклад, квитки 1 1 3 1 і 1 2 3 4 є щасливими, а 2 2 2 3 - ні (оскільки, ***m(m(4))=m(3)=2 ≠ m(4)=3***).

Степан вирішив, що можна отримати додатковий дохід, продаючи щасливі квитки з націнкою. Тепер для складання економічного плану йому потрібно знати, скільки існує щасливих квитків при даному ***N***. Допоможіть йому.

**Вхідні дані:** У першому рядку одне число ***N (1 ≤ N ≤ 30)***.

**Вихідні дані:** Виведіть у вихідний файл одне число - кількість можливих щасливих квитків при даному ***N*** без ведучих нулів по модулю 109+7.

**Приклад вхідних і вихідних даних:**

|  |  |
| --- | --- |
| **tickets.in** | **tickets.out** |
| 2 | 3 |

**Аквапарк**

|  |  |
| --- | --- |
| **Input file name:** | aquapark.in |
| **Output file name:** | aquapark.out |
| **Time limit:** | 300 ms |
| **Memory limit:** | 128 M |

У новому аквапарку є ***N*** басейнів, заповнених водою. Деякі пари басейнів з'єднані трубами, всього ***М*** труб. Система труб збудована так, щоб з будь-якого басейна у будь-який інший можна було при потребі перегнати воду, можливо, скориставшись іншими басейнами як проміжними. На кожній трубі стоїть програмований насос, який здатний перекачувати у будь-який бік довільно заданий потік води. Крім того, в деякі басейни додатково тече вода з відомою швидкістю, а з деяких вода витікає - також з відомою швидкістю.

Ваше завдання - написати програму, яка відрегулює насоси так, щоб рівень води в басейнах підтримувався постійним.

**Формат вхідних даних:** У першому рядку вказано кількість басейнів ***N (1 ≤ N ≤ 100)***. Далі йдуть ***N*** рядків, що містять інформацію про басейни. Для кожного басейну, починаючи з 1-го і закінчуючи ***N***-им, вказана різниця між додатковим потоком, що тече до басейну і потоком, що витікає - ціле число, що не перевищує за модулем 30 000 (додатне число відповідає потоку що тече у басейн, від'ємне - потоку, що витікає).

На наступному рядку вхідного файлу знаходиться число труб ***M (0 ≤ M ≤ 4950)***. У наступних ***M*** рядках знаходиться опис труб: для кожної труби вказані два числа - номери басейнів, які труба з'єднує. Між кожною парою басейнів є не більше однієї труби, труби не можуть з'єднувати басейн з самим собою.

**Формат вихідних даних:** Якщо рішення існує, виведіть у вихідний файл ***M*** рядків, по числу на рядок. Для кожної труби в тому ж порядку, як у вхідному файлі, має бути зазначено ціле число - потік води через неї. Додатним числом позначається потік від басейну, зазначеного першим у вхідному файлі, до другого; від'ємним - зворотний потік. Потоки у вихідному файлі не повинні перевищувати 2 000 000 000.

Якщо можливо кілька рішень, можна виводити будь-яке з них.

У разі, коли рішення не існує, вихідний файл повинен містити один рядок 'Impossible'.

**Приклад вхідних і вихідних даних:**

|  |  |
| --- | --- |
| **aquapark.in** | **aquapark.out** |
| 4-100-10010020041 24 31 34 1 | Impossible |
| 4-100-10010010041 24 31 34 1 | 1000-100100 |