

### Задача A1. Стари карти

Картите, които не са в електронен вид вече се смятат за стари. Те съдържат вярна и може би пълна информация, но за какво може да се използва карта, която дори няма търсачка? А без автоматично намиране на пътища, картата е напълно неизползваема за съвременните потребители. За нещастие, старите карти представят данните в много неудобен вид на огромна картина, която е пълна с излишна и неизползваема информация. Единствената съществена информация са градовете и пътищата между тях. Известно е, че между всеки два града съществува или директен път, или път, минаващ през няколко други града. Всеки път има фиксирана дължина, като между два града може да има най-много един пряк път. Ако такъв съществува в едната посока, то съществува и пряк път със същата дължина в другата посока.

Единствената надеждна информация, която може да се вземе в електронен вид от стара карта, е таблица с минималните разстояния между всеки два града. Градовете са безкрайно малки и когато преминават през град, се счита, че изминават 0 км.

Входните данни започват с ред, съдържащ числото  $N$  – броя на градовете ( $3 \leq N \leq 500$ ). На следващите  $N$  реда има по  $N$  числа.  $j$ -тото число на  $i$ -тия ред е разстоянието от град  $i$  до град  $j$  и нека го означим с  $D_{ij}$  ( $1 \leq i, j \leq N$ ). Знаем, че  $0 < D_{ij} = D_{ji} \leq 1000000$ . Също така  $D_{ii} = 0$ .

Вашата програма **MAPS** трябва да отпечата описание на всички преки пътища, съществуващи на картата, ако е известно, че техният брой е минимален. Описанието на всеки път трябва да бъде отпечатано на отделен ред и трябва да съдържа три числа, разделени с интервал – двата града, свързани с пътя и разстоянието между тях. Тъй като старата карта е надежден източник на информация – задачата винаги ще има решение.

Примерен вход 1:

```
3
0 10 20
10 0 30
20 30 0
```

Примерен изход 1:

```
1 2 10
1 3 20
```

Примерен вход 2:

```
3
0 10 20
10 0 25
20 25 0
```

Примерен изход 2:

```
1 2 10
1 3 20
2 3 25
```

### Решение:

Ако разгледаме градовете като върхове, а пътищата като ребра, можем да решим задачата в термините на крайни неориентирани графи.

В задачата разполагаме с матрица на разстоянията между всяка двойка върхове в граф  $G$ , който обаче е неизвестен. Нека означим тази матрица с  $D(G)$ . От нея елементарно може да се построи пълен граф  $P$ , чиято матрица  $D(P) = D(G)$ , като между всеки два върха поставим ребро с дължина равна на разстоянието между тях в матрицата. Първо забелязваме, че  $G$  е подграф на  $P$  – т.е. те имат едни и същи върхове и ако едно ребро е от  $G$ , тогава то е и от  $P$ . Това е така, защото за ребро от  $G$ ,  $(a, b)$ , свързващо върха  $a$  с върха  $b$  е изпълнено, че разстоянието между двата върха е точно равно на дължината на това ребро и следователно това е ребро и от  $P$  (ако допуснем, че реброто е по-късо, то разстоянието би било по-малко, а ако допуснем, че реброто е по-дълго, тогава то би било излишно и това противоречи с минималния брой ребра в  $G$ ).

Нека разгледаме алгоритъм на Дейкстра в пълния граф  $P$  с начален връх  $i$ . На всяка стъпка от алгоритъма избираме оня връх  $j$ , за който разстоянието от  $i$  до  $j$  е най-малко измежду разстоянията от  $i$  до върховете, които не са избрани досега. Заедно с такъв връх  $j$ , ние избираме и ребро, което е последното в най-късия път от  $i$  до  $j$ . Нека на всяка стъпка се стремим това ребро да е минимално, т.е. ако има няколко пътя до  $j$ , да избираме този, за който последното ребро е минимално. Съвкупностите от тези ребра образуват покриващо дърво на графа  $P$ , което ще означим с  $D(i)$ .

Твърдим, че графа  $G$  е обединение на дърветата  $D(i)$  за всеки връх  $i$ .

Доказателство: Тъй като върховете на дърветата и  $G$  съвпадат, разглеждаме само ребрата. Нека реброто  $r(a, b)$  е от  $D(i)$  за някое  $i$  и дължината му е  $len$ . Тогава разстоянието между  $a$  и  $b$  е равно на  $len$  (дължината е по-малка или равна на  $len$ , но не може да е по-малка, тъй като пътят до  $b$  би бил по-къс, без да се включва  $r(a, b)$ ). Тогава върховете  $a$  и  $b$  в  $G$  са свързани или с  $r(a, b)$ , или с някакъв друг път със същата дължина. Но ако са свързани с път, то последното ребро в този път е по-късо от  $r(a, b)$ , което е противоречие с минималността на последното ребро за пътя от  $i$  до  $b$ .

Нека  $r(a, b)$  е ребро от  $G$ , с дължина  $lenr$ . Нека разгледаме  $D(a)$ . Реброто  $r(a, b)$  или е ребро от  $D(a)$ , или в  $D(a)$  има друг път между  $a$  и  $b$ . Нека допуснем, че има друг път  $a, x_1, x_2, \dots, b$  с дължина  $len$ . От това, че  $D(a)$  е построено с алгоритъм за минимална дължина от  $a$  до всеки връх следва, че  $len \leq lenr$ . Но от първата част на доказателството всяко от ребрата  $r(a, x_1), r(x_1, x_2), \dots, r(x_n, b)$  е от  $G$ . Тогава в  $G$  има път с дължина  $len$  ( $len \leq lenr$ ), свързващ  $a$  и  $b$ . Тогава реброто  $r(a, b)$  е излишно, защото не променя разстоянието между  $a$  и  $b$  в  $G$ , както и кое да е друго разстояние в  $G$ . Това е в противоречие с минималния брой ребра в  $G$ . Оттук следва, че  $r(a, b)$  участва в  $D(a)$ .

С това доказахме, че графът  $G$  е обединение на дърветата  $D(i)$  за всеки връх  $i$ . В доказателството съществено използвахме и факта, че дължините на ребрата са положителни.

## Задача A2. Цифри

Дадено е число  $N$ , записано в  $K$ -ична бройна система,  $2 \leq K \leq 36$ . Главните латински букви A, B, C, D, E, F, ... означават цифри, съответно със стойности 10, 11, 12, 13, 14, 15, ... . Дадена е и една цифра  $M$  (число между 0 и 9, или главна латинска буква, такава че цифрата която тя означава е по-малка от  $K$ ). Да се напише програма **DIGITS**, която намира колко пъти цифрата  $M$  се използва за записването на всички числа от 1 до  $N$  в  $K$ -ична бройна система.

На първия ред от входния файл са записани числото  $R$  ( $1 \leq R \leq 1000000$ ) (което задава дължината на числото  $N$ ), цифрата  $M$  (записана с десетично число от 0 до  $K - 1$ ) и числото  $K$ , всичките разделени с интервал. На следващия ред се намира числото  $N$ , записано в  $K$ -ична бройна система. То се състои от последователни цифри, без интервал между тях.

На единствения ред в изходния файл програмата трябва да изведе търсения брой, записан в  $K$ -ична бройна система.

В около 40% от тестовете  $M$  ще е равно на 0.

В около 20% от тестовете  $R \leq 6$ .

В около 50% от тестовете  $R \leq 5000$ .

Примерен вход:

6 11 36

GWFERZ

Съответният изход:

3CMATS

Примерен вход:

6 0 10

107351

Съответният изход:

49517