

## ALGORITMOS I

### 13ª LISTA DE EXERCÍCIOS

#### 01 MONOPÓLIO (OBI 2006)

Monopólio (conhecido no Brasil como Banco Imobiliário) é um dos jogos mais famosos do mundo, com 750 milhões de cópias vendidas. Durante o jogo, os jogadores podem comprar propriedades que estejam disponíveis, vendê-las para que elas voltem a ficar disponíveis, e cobrar aluguel pelo uso de uma determinada propriedade por outro jogador. O objetivo do jogo é acumular a maior quantidade de dinheiro possível.

O jogo é composto por um tabuleiro e um conjunto de cédulas de dinheiro. Três amigos, Dália, Elói e Félix, querem jogar uma partida de Monopólio, mas o irmãozinho menor de Dália escondeu as cédulas de dinheiro. Os três amigos decidiram jogar a partida assim mesmo, anotando em um papel todas as operações que ocorreram durante o jogo (compras, vendas e pagamentos de aluguéis). Assim que eles pararam de jogar, perceberam que levaria muito tempo para descobrir quanto dinheiro cada um acumulou. Eles então pediram sua ajuda para determinar esses valores.

#### Tarefa

Sua tarefa é escrever um programa que, a partir dos registros de jogadas realizados pelos três jogadores, determine a quantidade de dinheiro acumulada por cada um dos jogadores.

#### Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado). A primeira linha da entrada contém dois inteiros,  $I$  e  $N$  que indicam respectivamente as quantias de dinheiro que Dália, Elói e Félix possuem no início do jogo ( $1 \leq I \leq 1000000$ ) e o número de operações realizadas durante o jogo ( $1 \leq N \leq 10000$ ). Note que os três jogadores iniciam a partida com a mesma quantidade de dinheiro. Os jogadores são representados na entrada sempre pela letra inicial de seu nome ('D', 'E' ou 'F'). As  $N$  linhas contém as operações ocorridas durante o jogo. Cada linha pode ter um dos formatos abaixo:

- Compra — a letra C, seguida da letra inicial de um jogador J e de um inteiro X que representa o valor gasto por J na compra ( $0 < X \leq 1000000$ ).  
Exemplo: 'C D 1000'.
- Venda — a letra V, seguida da letra inicial de um jogador J e de um inteiro X que representa o valor recebido por J na venda ( $0 < X \leq 1000000$ ).  
Exemplo: 'V E 200'.
- Aluguel — a letra A, seguida da letra inicial de um jogador J que recebe o aluguel, da letra inicial do jogador K que paga o aluguel e de um inteiro X que representa o valor do aluguel ( $J \neq K$  e  $0 \leq X \leq 1000000$ ).  
Exemplo: 'A F D 500'.

Os valores intermediários e totais acumulados por cada jogador estão entre 0 e 1000000.

## Saída

Seu programa deve imprimir, na saída padrão, uma única linha composta de três inteiros que correspondem à quantidade de dinheiro acumulada por Dália, Elói e Félix, nesta ordem

## Exemplos

Entrada 1000 1 C D 500  Saída 500 1000 1000	Entrada 1000 3 C D 100 V E 200 A D F 1000  Saída 1900 1200 0	Entrada 10000 5 C D 5000 C E 3000 A D F 1000 V E 4000 A F E 1000  Saída 6000 10000 10000
--	---	---

## 02 UIQUIPÉDIA (OBI 2007)

A Uiquipédia (Wikipedia em inglês), fundada em 2001 por Jimmy Wales e Larry Sanger, é um site onde qualquer pessoa pode editar os artigos, fazendo correções ou ampliando seu conteúdo.

Uma das grandes vantagens da Uiquipédia sobre enciclopédias de papel é a facilidade de seguir referências; com um simples clique, é possível ir de um artigo para outro relacionado. Essas referências são chamadas de referências diretas. Também é possível navegar a Uiquipédia sequencialmente: cada artigo possui referência para o artigo anterior e para o posterior, na ordem alfabética. Essas referências são chamadas de referências sequenciais.

Por exemplo, um artigo para o termo “Elefante” pode ter uma referência direta para “Mamíferos” em seu texto, desta forma pode-se chegar de “Elefante” a “Mamíferos” em um clique. Observe que pode não existir a referência direta contrária, ou seja, de “Mamíferos” para “Elefante”. Adicionalmente se “Elevador” é o próximo artigo depois de “Elefante”, na ordem alfabética, pode-se ir com um clique de “Elefante” para “Elevador” e de “Elevador” para “Elefante”, pois há uma referência sequencial entre eles.

Paulo e André são dois amigos que contribuem para a Uiquipédia. Muitas vezes, André edita um artigo e quer que Paulo o ajude a revisar a modificação. A conexão de Paulo à Internet é discada, e por isso ele quer chegar na página que André editou usando o menor número de cliques possível, começando do artigo em que está, e navegando apenas por referências, diretas ou sequenciais.

## Tarefa

Escreva um programa que, dados todas as referências diretas existentes na Uiquipédia, a página onde Paulo está, e a página editada por André, determina de quantos cliques Paulo precisa, no mínimo, para ver a página que foi modificada por André, utilizando as referências diretas e sequenciais.

## Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado). A primeira linha contém um único inteiro,  $N$ , que é o número de referências da Uiquipédia ( $1 \leq N \leq 1.000$ ). As  $N$  linhas contém cada uma duas strings  $X$  e  $Y$ , separadas por um espaço, que são os nomes de duas páginas da Uiquipédia conectadas por uma referência direta (de  $X$  para  $Y$ ). Todo artigo existente na Uiquipédia aparece pelo menos uma vez na descrição das referências diretas, permitindo que as referências sequenciais sejam extraídas das informações dadas. Note que uma referência direta pode ligar duas páginas que estariam ligadas também por uma referência sequencial.

Depois da descrição das referências, há uma linha em branco, e a linha seguinte contém duas cadeias de caracteres,  $P$  e  $A$ , que são a página atual de Paulo e a página editada por André. O nome de cada página é limitado a 100 caracteres e contém somente letras maiúsculas, letras minúsculas e o símbolo '-'. Observe que na ordem alfabética o símbolo '-' é anterior às letras maiúsculas, que por sua vez são anteriores às letras minúsculas.

## Saída

Seu programa deve imprimir, na *saída padrão*, uma única linha, contendo um único inteiro, que diz o número mínimo de cliques que são necessários para ir da página atual de Paulo até a página editada por André. Sempre é possível navegar de um artigo a outro.

## Exemplos

<p>Exemplo de entrada</p> <p>3</p> <p>Pink_Floyd O_Lado_Escuro_Da_Lua</p> <p>Pink_Floyd O_Muro</p> <p>O_Muro Muro_de_Berlim</p> <p>O_Muro O_Lado_Escuro_Da_Lua</p>	<p>Exemplo de saída</p> <p>1</p>
<p>Exemplo de entrada</p> <p>4</p> <p>Chaves Quico</p> <p>Quico Chiquinha</p> <p>Professor_Girafales Dona_Florinda</p> <p>Chaves Dona_Clotilde</p> <p>Chaves Chiquinha</p>	<p>Exemplo de saída</p> <p>2</p>

## 03 TELEFONE (OBI 2008)

As primeiras redes públicas de telefonia foram construídas pela AT&T no começo do século XX. Elas permitiam que seus assinantes conversassem com a ajuda de uma telefonista, que conectava as linhas dos assinantes com um cabo especial.

Essas redes evoluíram muito desde então, com a ajuda de vários avanços tecnológicos.

Hoje em dia, essas redes atendem centenas de milhões de assinantes; ao invés de falar diretamente com uma telefonista, você pode simplesmente discar o número da pessoa desejada no telefone.

Cada assinante recebe um número de telefone — por exemplo, 55–98–234–5678. Qualquer pessoa que discar esse número consegue então falar com a pessoa do outro lado da linha. Os hifens no número de telefone são só para facilitar a leitura, e não são discados no telefone.

Para que fique mais fácil de se lembrar de um número de telefone, muitas companhias divulgam números que contêm letras no lugar de dígitos. Para convertê-los de volta para dígitos, a maioria dos telefones tem letras nas suas teclas:

1	2 A B C	3 D E F
4 G H I	5 J K L	6 M N O
7 P Q R S	8 T U V	9 W X Y Z
*	0	#

Ao invés de discar uma letra, disca-se a tecla que contém aquela letra. Por exemplo, se você quiser discar o número 0800–FALE–SBC, você na realidade discaria 0800–3253–722.

A sua avó tem reclamado de problemas de vista — em particular, ela não consegue mais enxergar as letrinhas nas teclas do telefone, e por isso queria que você fizesse um programa que convertesse as letras em um número de telefone para dígitos.

### Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado). A entrada é composta de apenas uma linha, contendo o número de telefone que deve ser traduzido. O número de telefone contém entre 1 e 15 caracteres, que podem ser letras de 'A' a 'Z' e hifens ('-').

### Saída

Seu programa deve imprimir, na saída padrão, uma única linha, contendo o número de telefone com as letras convertidas para dígitos. Hifens no número telefone devem ser mantidos no número de telefone de saída.

### Exemplos

Exemplo de entrada PIPOCA	Exemplo de saída 747622
Exemplo de entrada FALE-SBC	Exemplo de saída 3253-722

## 04 VESTIBULAR

A maioria das universidades brasileiras usa o *vestibular* para selecionar seus alunos. O vestibular consiste de uma ou mais provas sobre as matérias do Ensino Médio, visando avaliar os conhecimentos dos candidatos.

Um formato popular de prova de vestibular é a *prova objetiva*. Neste formato de prova, cada candidato deve escolher uma das cinco alternativas apresentadas pela questão como sendo a correta. Durante a correção dos cartões, cada questão onde a alternativa escolhida pelo candidato é a mesma do gabarito, ele ganha um ponto.

Alguns dos vestibulares mais concorridos do Brasil são disputados por dezenas de milhares de candidatos, e, por isso, geralmente usa-se uma folha de leitura óptica e um programa de computador para corrigir as provas de todos os candidatos e gerar a lista com suas pontuações.

Você trabalha no comitê responsável pelo vestibular em uma faculdade e deve escrever um programa que, dado o gabarito e as respostas de um dos candidatos, determina o número de acertos daquele candidato.

### Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado). A primeira linha da entrada contém um único inteiro  $N$  ( $1 \leq N \leq 80$ ), representando o número de questões na prova. A segunda linha da entrada contém uma cadeia de  $N$  caracteres, indicando o gabarito da prova. A terceira linha da entrada contém outra cadeia de  $N$  caracteres, indicando as opções marcadas pelo candidato. Ambas as cadeias contém apenas os caracteres 'A', 'B', 'C', 'D' e 'E' (sempre em letra maiúscula).

### Saída

Seu programa deve imprimir na saída padrão uma única linha contendo um único inteiro, indicando o número de acertos do candidato

### Exemplos

Exemplo de entrada 7 AEDBCCE ADDCCBE	Exemplo de saída 4

Exemplo de entrada 5 ABCDE ABCDE	Exemplo de saída 5
Exemplo de entrada 10 ABCDEABCDE BCDEABCDEA	Exemplo de saída 0

## 05 QUADRADINHO DE 8 (OBI 2013)

Fernando ficou sabendo de um novo jogo chamado quadradinho de 8. Nesse jogo, é apresentado ao jogador uma fileira de quadrados, um do lado do outro. Em cada quadrado há um número escrito. Veja abaixo um exemplo de fileira de quadrados:

3	4	6	0	2	9
---	---	---	---	---	---

Para ganhar, o jogador deve escolher alguns quadrados de forma que eles juntos formem apenas um retângulo contíguo e que a soma de seus números seja divisível por 8. Na fileira de quadrados acima, o jogador ganha se escolher os quadrados com os números 6, 0 e 2. O jogador perde se escolher os quadrados com 3, 4 e 9, apesar da soma ser divisível por 8, os quadrados não estão juntos, eles acabam formando dois retângulos separados.

Você deve estar pensando agora que Fernando quer sua ajuda para que você mostre a ele como ganhar o jogo, mas Fernando é um garoto muito esperto e sabe resolver o jogo rapidamente. Ele quer na verdade que você o ajude a descobrir de quantas formas é possível ganhar esse jogo.

### Entrada

A entrada possui duas linhas. A primeira linha contém apenas um inteiro  $N$  que indica o número de quadrados na fileira de um jogo. A segunda linha contém  $N$  inteiros indicando na ordem os números presentes nos quadrados da fileira de um jogo.

### Saída

Seu programa deve imprimir uma única linha, contendo apenas um inteiro, o número de maneiras de ganhar o jogo apresentado na entrada. Se não for possível que o jogador ganhe o jogo, imprima 0.

### Restrições

$$1 \leq N \leq 1000000$$

Os números nos quadrados são inteiros não negativos menores ou iguais a 1000.

### Exemplos

Entrada	Saída
---------	-------

6 3 4 6 0 2 9	3
Entrada 7 1 1 1 1 1 1 1	Saída 0
Entrada 5 8 0 8 0 8	Saída 15

## 06 TROCO (OBI 2013)

Você está num supermercado e está na fila do caixa para comprar alguns produtos. Assim que você termina de passar as compras pelo caixa, se lembra que tem várias moedas em seu bolso, algumas repetidas, e fica pensando se com elas dá para pagar exatamente o valor das compras (para assim se livrar destas moedas e ficar com os bolsos mais leves). Você consegue pagar o valor exato da conta usando estas moedas?

### Entrada

A primeira linha da entrada contém dois números inteiros  $V$  e  $M$ , indicando, respectivamente, o valor final da compra e o número de moedas que você tem em seu bolso. A segunda linha contém  $M$  números inteiros que descrevem o valor  $M_i$  de cada moeda existente em seu bolso.

### Saída

Seu programa deve imprimir apenas uma linha, contendo apenas um caractere: S caso seja possível pagar o valor exato da conta usando apenas suas moedas, ou N caso contrário.

### Restrições

$$1 \leq V \leq 10^5$$

$$1 \leq M \leq 10^3$$

$$1 \leq M_i \leq 10^5$$

### Exemplos

Entrada 16 4 25 10 5 1	Saída S

Entrada	Saída
20 4 25 10 5 1	N

## 07 LENÇOL (OBI 2013)

João dispõe de dois pedaços retangulares de tecido, e quer usá-los para fazer um lençol, também retangular, de dimensões  $A \times B$ . Se necessário, os dois pedaços retangulares podem ser unidos por uma costura, mas João quer que a costura seja paralela aos lados dos retângulos. Os cortes, se necessários, também devem ser paralelos aos lados dos retângulos.

Dadas as dimensões dos pedaços de tecido e do lençol, escreva um programa que determina se é possível João fazer o lençol com as dimensões desejadas

### Entrada

A entrada contém uma única linha, com seis inteiros  $A_1, B_1, A_2, B_2, A$  e  $B$ , representando, respectivamente, as dimensões dos dois retângulos disponíveis, e as dimensões do retângulo desejado.

### Saída

Seu programa deve imprimir uma única linha contendo um caractere  $S$  se é possível fazer o lençol, e  $N$  caso contrário.

### Restrições

$$1 \leq A_1, B_1, A_2, B_2, A, B \leq 10^6$$

### Exemplos

Entrada	Saída
4 2 3 5 4 4	S
Entrada	Saída
4 2 2 5 4 5	N
Entrada	Saída
1 2 3 5 5 2	S
Entrada	Saída



3 4 10 9 9 10	S
---------------	---

## 08 TORRE (OBI 2015)

No jogo de xadrez, a torre é uma peça que pode se mover para qualquer outra posição do tabuleiro na linha ou na coluna da posição que ela ocupa. O professor Paulo está tentando inventar um novo tipo de jogo de xadrez onde todas as peças são torres, o tabuleiro também é quadrado mas pode ter qualquer dimensão e cada posição do tabuleiro é anotada com um número inteiro positivo, como na figura abaixo.

	1	2	3	4	5	6
1	4	1	3	8	4	5
2	9	2	8	9	2	7
3	5	5	4	3	2	5
4	8	2	9	<span style="border: 1px solid black; padding: 2px;">1</span>	9	8
5	7	1	3	2	1	2
6	5	1	2	9	3	8

Ele definiu o **peso** de uma posição  $(i, j)$  como sendo a soma de todos os números que estejam na linha  $i$  com todos os números da coluna  $j$ , mas sem somar o número que está exatamente na posição  $(i, j)$ . Quer dizer, se uma torre estiver na posição  $(i, j)$ , o peso da posição é a soma de todas as posições que essa torre poderia atacar.

O professor Paulo está solicitando a sua ajuda para implementar um programa que determine qual é o peso máximo entre todas as posições do tabuleiro. No exemplo da figura acima, com um tabuleiro de dimensão seis (ou seja, seis linhas por seis colunas), o peso máximo é 67, referente à posição (4,4).

### Entrada

A primeira linha da entrada contém um inteiro  $N$ , representando a dimensão do tabuleiro. Cada uma das  $N$  linhas seguintes contém  $N$  inteiros positivos  $X_i$ , definindo os números em cada posição do tabuleiro.

### Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o peso máximo entre todas as posições do tabuleiro.

### Restrições

- $3 \leq N \leq 1000$
- $0 < X_i \leq 100$

### Informações sobre a pontuação

- Em um conjunto de casos de teste cuja soma é 60 pontos,  $N \leq 300$ .

### Exemplos

Entrada	Saída
6 4 1 3 8 4 5 9 2 8 9 2 7 5 5 4 3 2 5 8 2 9 1 9 8 7 1 3 2 1 2 5 1 2 9 3 8	67

## 09 LETRAS (OBI 2015)

Uma cadeia de caracteres é uma sequência de letras do alfabeto. Uma cadeia de caracteres crescente é uma sequência de letras onde a próxima letra (da esquerda para a direita) nunca ocorre antes no alfabeto do que a letra anterior. Por exemplo ABBD é crescente, enquanto ABBAD não é crescente.

Uma subsequência de uma cadeia de caracteres é uma cadeia de caracteres que pode ser obtida a partir da remoção de zero ou mais caracteres da cadeia de caracteres original. Por exemplo ANNA é uma subsequência de BANANAS. Outro exemplo seria ANNS, que é uma subsequência crescente de BANANAS.

Dada uma cadeia de caracteres  $S$ , escreva um programa para determinar o tamanho da maior subsequência de  $S$  que é uma cadeia de caracteres crescente.

### Entrada

A entrada consiste em uma única linha, contendo uma cadeia de caracteres  $S$ .

### Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o tamanho da maior subsequência de  $S$  que é uma cadeia de caracteres crescente.

### Restrições

- A cadeia de caracteres de entrada contém letras maiúsculas do alfabeto, de A até Z.
- $1 \leq \text{comprimento}(S) \leq 3 \times 10^5$ .

### Informações sobre a pontuação

- Em um conjunto de casos de teste valendo 20 pontos:  $\text{comprimento}(S) \leq 20$ .
- Em um conjunto de casos de teste valendo 30 pontos:  $\text{comprimento}(S) \leq 3000$ .

### Exemplos

Entrada	Saída
BANANAS	4

Entrada	Saída
AAXBXZZX	7
Entrada	Saída
AAA	3

## 10 IMPEDIMENTO! (OBI 2015)

A regra do impedimento no futebol pode parecer estranha, mas sem ela, se a gente pensar bem, o jogo ficaria muito chato! Ela funciona dadas as posições de três jogadores: L o jogador atacante que lança a bola; R o jogador atacante que recebe a bola; e D o último jogador defensor. E a regra vale somente se o jogador R está no seu campo de ataque; se o jogador R está no seu campo de defesa ou na linha divisória do meio campo, ele não está em impedimento. Neste problema o campo tem 100 metros de comprimento. Dadas as posições desses três jogadores, no momento exato do lançamento, haverá impedimento se e somente se a seguinte condição for verdadeira:

$$(R > 50) \text{ e } (L < R) \text{ e } (R > D)$$

A regra parece estranha, não é mesmo? Mas a gente nem precisa entender a lógica dela. O seu programa deve apenas determinar, dadas as três posições L,R e D, se há ou não impedimento, implementando exatamente a condição acima. A figura abaixo mostra um exemplo onde não há impedimento:

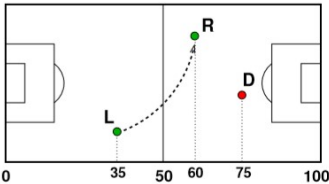
3 8

↺

esse momento se o jogador R está no seu campo de ataque; se o jogador R está no seu campo de defesa ou na linha divisória do meio campo, ele não está em impedimento. Neste problema o campo tem 100 metros de comprimento. Dadas as posições desses três jogadores, no momento exato do lançamento, haverá impedimento se e somente se a seguinte condição for verdadeira:

$$(R > 50) \text{ e } (L < R) \text{ e } (R > D)$$

A regra parece estranha, não é mesmo? Mas a gente nem precisa entender a lógica dela. O seu programa deve apenas determinar, dadas as três posições *L*, *R* e *D*, se há ou não impedimento, implementando exatamente a condição acima. A figura abaixo mostra um exemplo onde **não** há impedimento:



**Entrada**

A entrada é composta de apenas uma linha, contendo os três inteiros *L*, *R* e *D*.

**Saída**

Seu programa deve produzir uma única linha, contendo um único caractere, que deve ser "S" caso haja impedimento, ou "N" caso contrário.

**Restrições**

↻

Entrada

A entrada é composta de apenas uma linha, contendo os três inteiros L,R e D. Saída Seu programa deve produzir uma única linha, contendo um único caractere, que deve ser "S" caso haja impedimento, ou "N" caso contrário. Restrições •  $0 \leq L \leq 100$  •  $0 \leq R \leq 100$  •  $0 \leq D \leq 100$  Exemplos

Entrada 35 60 75

Saída N

Entrada 55 68 67

Saída S

Olimpíada Brasileira de Informática – OBI2015 3

Entrada 66 80 80

Saída N

Olimpíada Brasileira de Informática – OBI2015 4

Capitais

Nome do arquivo: capitais.c, capitais.cpp, capitais.pas, capitais.java, capitais.js ou capitais.py

A Linearlândia construiu uma rede de ferrovias de alta velocidade, ligando certos pares de cidades, de modo que: é possível viajar entre qualquer par de cidades usando apenas ferrovias; e há apenas um caminho de ferrovias (sequência de ferrovias) entre qualquer par de cidades. Existe muita disputa entre as capitais dos estados da Linearlândia e, por isso, ficou decidido que cada capital seria ligada por ferrovia a apenas uma outra cidade, e que toda cidade que não é capital seria ligada a outras cidades por duas ou mais ferrovias. Dessa forma, nenhuma viagem entre um par de capitais usando apenas ferrovias passa por uma terceira capital.

5

12

Vamos definir como distância-ferrovia entre duas cidades o número de ferrovias que é necessário utilizar para viajar entre essas duas cidades. Dada apenas a informação sobre quais pares de cidades estão ligados por uma ferrovia, você deve escrever um programa para computar a menor distância-ferrovia entre todos os pares de capitais. Na figura acima, há nove capitais e a menor distância-ferrovia entre qualquer par de capitais é 3, entre as capitais 5 e 12. Entrada A primeira linha da entrada contém um inteiro N, o número de cidades. As cidades são identificadas por inteiros de 1 a N. As N-1 linhas seguintes contém, cada uma, dois inteiros U e V, representando um par de cidades ligadas por uma ferrovia. Saída Seu programa deve produzir uma única linha, contendo um único inteiro, a menor distância-ferrovia entre todos os pares capitais. Restrições •  $2 \leq N \leq 105$  Informações sobre a pontuação • Em um conjunto de casos de teste cuja soma é 40 pontos,  $N \leq 104$ ; Exemplos Entrada 8 1 2 2 3 3 4 5 3 6 5 6 7 8 7 Saída 3

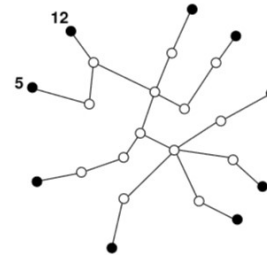
Entrada 2 1 2

Saída 1

## Capitais

Nome do arquivo: `capitais.c`, `capitais.cpp`, `capitais.pas`, `capitais.java`, `capitais.js` ou `capitais.py`

A Linearlândia construiu uma rede de ferrovias de alta velocidade, ligando certos pares de cidades, de modo que: é possível viajar entre qualquer par de cidades usando apenas ferrovias; e há apenas um caminho de ferrovias (sequência de ferrovias) entre qualquer par de cidades. Existe muita disputa entre as capitais dos estados da Linearlândia e, por isso, ficou decidido que cada capital seria ligada por ferrovia a apenas uma outra cidade, e que toda cidade que não é capital seria ligada a outras cidades por duas ou mais ferrovias. Dessa forma, nenhuma viagem entre um par de capitais usando apenas ferrovias passa por uma terceira capital.



Vamos definir como *distância-ferrovia* entre duas cidades o número de ferrovias que é necessário utilizar para viajar entre essas duas cidades. Dada apenas a informação sobre quais pares de cidades estão ligados por uma ferrovia, você deve escrever um programa para computar a menor distância-ferrovia entre todos os pares de capitais. Na figura acima, há nove capitais e a menor