Centro de Educação Superior a Distância do
Estado do Rio de Janeiro – CEDERJ

Curso de Tecnologia em Sistemas de Computação – TSC

EAD-05.009 Fundamentos de Programação

Caderno de Exercícios
Aula 5

(Vetor, Matriz, String (Cadeia de Caracteres) e Tuplas)

Professores

Dante Corbucci Filho
Leandro A. F. Fernandes

Instruções

- Utilize Python 3 e a IDE PyCharm na elaboração de soluções para os problemas propostos;
- A entrada de cada problema deve ser lida da entrada padrão (teclado);
- A saída de cada problema deve ser escrita na saída padrão (tela);
- Siga o formato apresentado na descrição da saída, caso contrário não é garantido que a saída emitida será considerada correta;
- Na saída, toda linha deve terminar com o caractere `'\n';`
- Utilize o URI Online Judge (http://www.urionlinejudge.com.br) e submeta sua solução para correção automática.

Referências Autorais

Os exercícios apresentados nesta lista foram extraídos do URI Online Judge (http://www.urionlinejudge.com.br). Acesse a URL apresentada abaixo do título de cada problema para proceder com a correção automática de sua solução e, também, para consultar a autoria do enunciado.

Problema A: Quantas substrings?

https://www.urionlinejudge.com.br/judge/pt/problems/view/1530

Inicialmente, há uma string vazia. Seu programa deve realizar dois tipos de instruções:

1. Adicionar um caractere entre 'a' e 'z' ao final da string.
2. Calcular quantas substrings diferentes a string possui.

Por exemplo, a string "aba" possui 5 substrings diferentes: "a", "ab", "aba", "b", "ba".

Entrada

A entrada é composta por vários casos de teste. Cada caso de teste consiste de uma linha contendo uma sequência com até 2×10^5 caracteres. Cada caractere representa uma instrução que deve ser feita. Um caractere entre 'a' e 'z' indica que deve ser realizado uma instrução do tipo 1 com esse caractere. Um caractere '?' representa uma instrução do tipo 2.

Saída

Para cada instrução do tipo 2, imprima uma linha contendo o número de substrings diferentes que a string possui.

Exemplo

Entrada	Saída
aba? ?z?z?z? abc?abc?	5 0 1 2 3 6 15

Problema B: Matriz de quadrados

https://www.urionlinejudge.com.br/judge/pt/problems/view/1578

Atrapalhilton é um estudante muito dedicado, embora muito, muito atrapalhado. Na semana passada, seu professor de Matemática, o Sr. Sabetudilton, recomendou à classe uma lista de exercícios sobre matrizes. Atrapalhilton, aplicado como é, decidiu fazer os exercícios no mesmo dia, tão logo chegou em casa. O enunciado de um dos exercícios dizia:

Calcule o quadrado de cada uma das matrizes abaixo…

No entanto, Atrapalhilton fez uma baita duma confusão. Para ele, o quadrado de uma matriz quadrada *A* é a matriz dos quadrados dos valores da matriz *A*. Por exemplo, o quadrado da matriz

$$\begin{matrix} 1 & 3 \\ 5 & 7 \end{matrix}$$

para ele não é

$$\begin{matrix} 16 & 24 \\ 40 & 64 \end{matrix}$$

mas

$$\begin{matrix} 1 & 9 \\ 25 & 49 \end{matrix}$$

Atrapalhilton conseguiu calcular o "quadrado" da primeira matriz, da segunda, da terceira e percebeu que já estava muito tarde, que não ia conseguir terminar de calcular os "quadrados" de todas as *N* matrizes da lista. Então, decidiu escrever um programa que fizesse o serviço para ele.

Entrada

A primeira linha da entrada é constituída por um único inteiro positivo *N* ($N \leq 100$), o qual designa o número de matrizes cujos "quadrados" ainda não foram calculados. Em seguida ocorre a descrição de cada uma das *N* matrizes. A primeira linha da descrição de uma matriz consiste de um único inteiro *M* ($1 \leq M \leq 20$), o qual representa o número de linhas e o número de colunas da matriz. Seguem, então, *M* linhas, cada uma com *M* inteiros a_{ij} ($0 \leq aij \leq 2^{32} - 1, 1 \leq i, j \leq M$), os quais correspondem às células da matriz, de modo que valores consecutivos numa mesma linha são separados por um espaço em branco.

Saída

Imprima o "quadrado" de cada matriz da entrada, conforme o que Atrapalhilton entende pelo "quadrado" de uma matriz. Antes de imprimir cada "quadrado", imprima a linha "Quadrado da matriz #*X*:" (sem as aspas), para ajudar Atrapalhilton a não se perder na hora de passar a limpo os resultados para o caderno. Comece a contagem em $X = 4$, afinal, Atrapalhilton já calculou os "quadrados" das 3 primeiras matrizes. Adicione tantos espaços em branco à esquerda de cada valor quanto necessários para

que os valores de uma mesma coluna fiquem todos alinhados à direita, de modo que haja ao menos um valor em cada coluna não precedido por espaços em branco além do espaço em branco obrigatório que separa colunas consecutivas. Imprima também uma linha em branco entre "quadrados" de matrizes consecutivas.

Exemplo

Entrada	Saída
1 2 7 12 1024 1	Quadrado da matriz #4: 49 144 1048576 1

Problema C: Frequência de letras

https://www.urionlinejudge.com.br/judge/pt/problems/view/1255

Neste problema estamos interessados na frequência das letras em uma dada linha de texto.

Especificamente, deseja-se saber qual(is) a(s) letra(s) de maior frequência do texto, ignorando o "case sensitive", ou seja maiúsculas ou minúsculas (sendo mais claro, "letras" referem-se precisamente às 26 letras do alfabeto).

Entrada

A entrada contém vários casos de teste. A primeira linha contém um inteiro *N* que indica a quantidade de casos de teste. Cada caso de teste consiste de uma única linha de texto. A linha pode conter caracteres "não letras", mas é garantido que tenha ao menos uma letra e que tenha no máximo 200 caracteres no total.

Saída

Para cada caso de teste, imprima uma linha contendo a(s) letra(s) que mais ocorreu(ocorreram) no texto em minúsculas (se houver empate, imprima as letras em ordem alfabética).

Exemplo

Entrada	Saída
3 Computers account for only 5% of the country's commercial electricity consumption. Input frequency letters	co inptu e

Problema D: Criptografia

https://www.urionlinejudge.com.br/judge/pt/problems/view/1024

Solicitaram para que você construísse um programa simples de criptografia. Este programa deve possibilitar enviar mensagens codificadas sem que alguém consiga lê-las. O processo é muito simples. São feitas três passadas em todo o texto.

Na primeira passada, somente caracteres que sejam letras minúsculas e maiúsculas devem ser deslocadas 3 posições para a direita, segundo a tabela ASCII: letra 'a' deve virar letra 'd', letra 'y' deve virar caractere '|' e assim sucessivamente. Na segunda passada, a linha deverá ser invertida. Na terceira e última passada, todo e qualquer caractere a partir da metade em diante (truncada) devem ser deslocados uma posição para a esquerda na tabela ASCII. Neste caso, 'b' vira 'a' e 'a' vira '`'.

Por exemplo, se a entrada for “Texto #3”, o primeiro processamento sobre esta entrada deverá produzir “Wh{wr #3”. O resultado do segundo processamento inverte os caracteres e produz “3# rw{hW”. Por último, com o deslocamento dos caracteres da metade em diante, o resultado final deve ser “3# rvzgV”.

Entrada

A entrada contém vários casos de teste. A primeira linha de cada caso de teste contém um inteiro N ($1 \leq \boldsymbol{N} \leq 1 * 10^4$), indicando a quantidade de linhas que o problema deve tratar. As N linhas contém cada uma delas M ($1 \leq \boldsymbol{M} \leq 1 * 10^3$) caracteres.

Saída

Para cada entrada, deve-se apresentar a mensagem criptografada.

Exemplo

Entrada	Saída
4 Texto #3 abcABC1 vxpdylY .ph vv.xwfxo.fd	3# rvzgV 1FECedc ks. \n{frzx gi.r{hyz-xx

Problema E: Sequência de Threebonacci

https://www.urionlinejudge.com.br/judge/pt/problems/view/1739

Um número pertence à sequência de Threebonacci caso pertença à sequência de Fibonacci (assuma que o primeiro termo da série é o 1) e atenda pelo menos um dos últimos critérios abaixo:

1. A representação do número possui pelo menos um dígito 3.
2. O número é múltiplo de 3.

Entrada

Cada caso de teste contém um inteiro N ($1 \leq N \leq 60$). A entrada termina com o fim de arquivo (EOF).

Saída

Para cada caso de teste imprima uma linha contendo o N-ésimo termo da série de Threebonacci.

Exemplo

Entrada	Saída
1 3	3 21

Problema F: DNA storage?

https://www.urionlinejudge.com.br/judge/pt/problems/view/1814

Ms. Dolejškova está atualmente interessada no problema das árvores filogenéticas, e trabalhando, portanto, com N cadeias de DNA. Para simplificar o trabalho, Ms. Dolejškova resolveu trabalhar apenas com cadeias gênicas de comprimento M (isto é, todas as cadeias possuem exatamente M bases nitrogenadas).

Um subproblema interessante envolve o armazenamento das N cadeias em disco. Até o momento, Ms. Dolejškova está utilizando um esquema ingênuo que requer $N \times M$ caracteres, além dos delimitadores. Isto é, todas as sequências são gravadas dentro de um arquivo texto, sequencialmente. Mr. Chuchle, um colega de departamento e especialista em técnicas de armazenamento, sugeriu uma alternativa que pode ser mais econômica.

Segundo Mr. Chuchle, é possível armazenar uma cadeia juntamente com informações que permitam transformá-la em outras. Mais especificamente, considere duas cadeias de DNA D_1 = ACTA e D_2 = AGTC, onde A, C, G, T representam as bases nitrogenadas adenina, citosina, guanina e timina, nesta ordem. Observe que é possível transformar D_1 em D_2 trocando-se as bases nitrogenadas C e A das posições 2 e 4 de D_1 para G e C, respectivamente. Considere agora uma terceira cadeia D3 = CGTC. E necessária apenas uma modificação para transformar D_2 em D_3 e são necessárias três modificações para transformar D_1 em D_3. Logo, é vantajoso permitir a transitividade das modificações entre as cadeias.

Ms. Dolejškova observou rapidamente que, se as cadeias envolvidas forem muito diferentes entre si, este esquema de armazenamento alternativo não oferece ganhos. Assim, em vez de adotá-lo prontamente, ela solicitou a você que construa um programa que recebe as N cadeias, e determina o número mínimo de transformações que devem ser gravadas (além de uma cadeia) para que seja possível, no futuro, obter-se novamente as N cadeias originais. Baseado no resultado fornecido por seu programa, Ms. Dolejškova vai decidir qual dos esquemas deve utilizar em cada instância de dados que tiver.

Entrada

Seu programa deve estar preparado para trabalhar com diversas instâncias. Cada instância tem a estrutura que segue. Na primeira linha são fornecidos dois inteiros N e M ($0 \leq N \leq 100$ e $1 \leq M \leq 1000$) que representam, nesta ordem, o número de cadeias de DNA e o comprimento delas. Nas próximas N linhas são fornecidas as N cadeias, uma por linha, sem espaços adicionais. Cada cadeia é uma sequência de caracteres tomada sobre o alfabeto $\Sigma = \{A, C, G, T\}$. Um valor $N = 0$ indica o final das instâncias e não deve ser processado.

Saída

Para cada instância solucionada, você deverá imprimir um identificador Instancia H em que H é um número inteiro, sequencial e crescente a partir de 1. Na linha seguinte, você

deve imprimir o número mínimo de transformações que devem ser gravadas para esta instância. Uma linha em branco deve ser impressa após cada instância.

Exemplo

Entrada	Saída
3 4 ACTA AGTC CGTC 4 5 AAAAA ATATA ATCTA AACAA 0 0	Instancia 1 3 Instancia 2 4

Problema G: Descobrindo uma matriz

https://www.urionlinejudge.com.br/judge/pt/problems/view/1791

Uma matriz é uma Matriz de Potências se atende 3 pré-requisitos:

1. É uma matriz quadrada.
2. A primeira coluna é formada apenas por 1's.
3. Para todo elemento (i, j) com $j > 1, (i, j) = (i, 2)^{j-1}$ e (i, j) é diferente de zero.

Por exemplo:

$$P = \begin{pmatrix} 1 & M[1][2]^1 & M[1][2]^2 & \cdots & M[1][2]^{n-1} \\ 1 & M[2][2]^1 & M[2][2]^2 & \cdots & M[2][2]^{n-1} \\ 1 & M[3][2]^1 & M[3][2]^2 & \cdots & M[3][2]^{n-1} \\ & \vdots & & \ddots & \vdots \\ 1 & M[n][2]^1 & M[n][2]^2 & \cdots & M[n][2]^{n-1} \end{pmatrix}$$

Sua tarefa é descobrir se uma matriz quadrada pode ser transformada em uma Matriz de Potências utilizando dois tipos de operações:

1. Troca(*X*, *Y*): Inverte as posições de todos os elementos das colunas *X* e *Y* da matriz.
2. Transposta(): A matriz é transposta.

Por exemplo:

$$P = \begin{pmatrix} 9 & 4 & 16 \\ 1 & 1 & 1 \\ 3 & 2 & 4 \end{pmatrix}$$

$$Transposta() = \begin{pmatrix} 9 & 1 & 3 \\ 4 & 1 & 2 \\ 16 & 1 & 4 \end{pmatrix}$$

$$Troca(1,2) = \begin{pmatrix} 1 & 9 & 3 \\ 1 & 4 & 2 \\ 1 & 16 & 4 \end{pmatrix}$$

$$Troca(2,3) = \begin{pmatrix} 1 & 3 & 9 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \end{pmatrix}$$

Logo *P* pode ser transformada em uma Matriz de Potência.

Entrada

A entrada consiste de múltiplas linhas. A primeira linha contém um inteiro C que indica o número de casos de teste. Em seguida, em cada caso de teste a primeira linha contém um inteiro N $(1 < N < 8)$ que indica o número de linhas e colunas da matriz, em seguida N linhas, cada uma com N inteiros D $(-50000 < D < 50000)$ representando os elementos da matriz .

Saída

Imprima em uma única linha para cada caso de teste a "Potencia" (sem aspas) caso a matriz possa ser transformada, ou "Nao Potencia" (sem aspas) caso contrário.

Exemplo

Entrada	Saída
3 3 16 1 4 1 1 1 9 1 3 3 25 36 9 1 1 1 5 6 3 3 9 35 25 3 6 5 1 1 1	Potencia Potencia Nao Potencia

Problema H: Branco e Preto

https://www.urionlinejudge.com.br/judge/pt/problems/view/1997

O famoso jogo Preto e Branco é um jogo individual que é jogado com um conjunto de fichas idênticas. Cada ficha tem duas faces com cores diferentes. Surpreendentemente, essas cores são preto e branco.

O jogo começa colocando *N* fichas formando uma única linha. Existe um objetivo que é uma dada sequência de *N* cores preto ou branco. Em um único movimento, o jogador pode escolher um grupo de fichas consecutivas e inverter a sua cor, em outras palavras, para cada ficha no grupo, a cor que estava voltada para cima, esta voltada para baixo, e a que estava voltada para baixo está virada para cima. O jogo termina quando as cores voltadas para cima são iguais ao objetivo.

Barby acaba de descobrir este jogo e logo ela percebeu que você pode sempre ganhar invertendo cada ficha individualmente, se necessário. Para tornar o jogo mais desafiador para ela, ela queria ganhar no menor número possível de movimentos. Note que Barby apenas se preocupa com quantos movimentos ela faz, e não importa quantas fichas são invertidas em cada jogada. Para saber o quão bem Barby está jogando, ela lhe pediu para fazer um programa que, dada a posição inicial da ficha e o objetivo, mostra o menor número possível de movimentos para ganhar o jogo. Você vai dizer que não?

Entrada

A entrada contém vários casos de teste. Cada caso de teste é descrito em uma única linha que contém duas palavras não vazias *S* e *T* de igual tamanho e, no máximo, 500 caracteres cada. *S* indica a posição inicial da ficha, enquanto *T* representa o objetivo. Ambas as palavras contêm apenas letras maiúsculas "B" e "N", que representam, respectivamente, branco e preto. A última linha da entrada contém dois asteriscos ("*") separados por um espaço único e não deve ser processado como um caso de teste.

Saída

Para cada caso de teste de saída, imprima uma única linha com um inteiro representando a quantidade mínima de jogadas necessárias para passar as fichas da posição inicial *S* para o objetivo *T*.

Exemplo

Entrada	Saída
BBNBBNBBBB NNNNNBBNNB BNBNB NBNBN BNBN NBNB B B * *	3 1 1 0

Problema I: A lista

https://www.urionlinejudge.com.br/judge/pt/problems/view/2037

O Comitê Internacional de Xadrez Profissional organiza um torneio para jogadores avançados, com uma metodologia muito estranha. Como esperado, em cada jogo exatamente dois jogadores se enfrentam mutuamente, mas neste caso apenas um jogo ocorre de cada vez, porque existe apenas um tabuleiro de xadrez disponível. Depois de receber as inscrições dos competidores e atribuindo-lhes um número, a organização decide arbitrariamente quais jogos irão acontecer e em qual ordem. Cada concorrente pode enfrentar qualquer outro concorrente qualquer número de vezes, e é até possível que alguns concorrentes nunca joguem uns contra os outros. Assim que decidido todos os jogos a serem jogados, a organização distribui a cada competidor uma lista não-vazia de seus rivais, em ordem cronológica (ou seja, a ordem em que os jogos serão realizados).

Florência inscreveu em primeiro lugar, de modo que a ela foi atribuído o número 1. Depois de conversar um pouco com os outros concorrentes, ela percebeu que havia perdido sua lista de rivais. Ela não quer incomodar os organizadores do torneio, então ela pediu a todos os outros concorrentes para obter uma cópia de suas próprias listas de rivais, na esperança de que, com esta informação, ela seria capaz de reconstruir sua lista perdida. Florência não tem certeza se existe apenas um tipo de lista de rivais que é compatível com todas as listas copiadas que foram dadas a ela pelos outros concorrentes. No entanto, ela sabe que a lista que ela foi dada pelos organizadores do torneio é de fato única. Sua tarefa é ajudá-la a reconstruir esta lista.

Entrada

Cada caso de teste é descrito usando duas linhas. A primeira linha contém um único número inteiro N, que representa o número de competidores ($2 \leq N \leq 9$). Cada concorrente é identificado por um número inteiro diferente de 1 a N, e concorrente número 1 é sempre Florência. A segunda linha contém $N - 1$ strings não vazias Li de no máximo de 100 caracteres cada (para i = 2, 3, ..., N). A string Li é composta unicamente de dígitos entre 1 e N, excluindo o dígito i, e representa a lista de rivais do concorrente número i em ordem cronológica. Note que o número do competidor 1 aparece pelo menos uma vez em uma das listas dadas. Em cada caso de teste, existe uma lista única de rivais para a concorrente número 1, que é compatível com as outras listas de rivais. O final da entrada é indicada por uma linha que contém o número -1.

Saída

Para cada caso de teste, você deve imprimir uma única linha contendo uma string, representando a lista única de rivais da concorrente número 1 (Florência) que seja compatível com as listas dos rivais dos outros concorrentes. Os rivais indicados nessa lista devem aparecer em ordem cronológica.

Exemplo

Entrada	Saída
4 314 142 321 9 31 412 513 614 715 816 917 18 4 11111111111111111111111111111 4 3 -1	324 98765432 22222222222222222222222222222

Problema J: Sudoku

https://www.urionlinejudge.com.br/judge/pt/problems/view/1383

O jogo de Sudoku espalhou-se rapidamente por todo o mundo, tornando-se hoje o passatempo mais popular em todo o planeta. Muitas pessoas, entretanto, preenchem a matriz de forma incorreta, desrespeitando as restrições do jogo. Sua tarefa neste problema é escrever um programa que verifica se uma matriz preenchida é ou não uma solução para o problema.

A matriz do jogo é uma matriz de inteiros 9 x 9 . Para ser uma solução do problema, cada linha e coluna deve conter todos os números de 1 a 9. Além disso, se dividirmos a matriz em 9 regiões 3 x 3, cada uma destas regiões também deve conter os números de 1 a 9. O exemplo abaixo mostra uma matriz que é uma solução do problema.

Entrada

São dadas várias instâncias. O primeiro dado é o número $N > 0$ de matrizes na entrada. Nas linhas seguintes são dadas as N matrizes. Cada matriz é dada em 9 linhas, em que cada linha contém 9 números inteiros.

Saída

Para cada instância seu programa deverá imprimir uma linha dizendo "*Instancia K*", onde K é o número da instância atual. Na segunda linha, seu programa deverá imprimir "*SIM*" se a matriz for a solução de um problema de Sudoku, e "*NAO*" caso contrário. Imprima uma linha em branco após cada instância.

Exemplo

Entrada	Saída
2 1 3 2 5 7 9 4 6 8 4 9 8 2 6 1 3 7 5 7 5 6 3 8 4 2 1 9 6 4 3 1 5 8 7 9 2 5 2 1 7 9 3 8 4 6 9 8 7 4 2 6 5 3 1 2 1 4 9 3 5 6 8 7 3 6 5 8 1 7 9 2 4 8 7 9 6 4 2 1 5 3 1 3 2 5 7 9 4 6 8 4 9 8 2 6 1 3 7 5 7 5 6 3 8 4 2 1 9 6 4 3 1 5 8 7 9 2 5 2 1 7 9 3 8 4 6 9 8 7 4 2 6 5 3 1 2 1 4 9 3 5 6 8 7 3 6 5 8 1 7 9 2 4 8 7 9 6 4 2 1 3 5 *	Instancia 1 SIM Instancia 2 NAO