Maratona Take

15ª Semana de Ciência & Tecnologia CEFET-MG

Questões Elaboradas por:

- André Rodrigues da Cruz (http://lattes.cnpq.br/3346655862356439)
- Felipe Duarte dos Reis (http://lattes.cnpq.br/4965215490981944)
- Lucas Dutra Marioza dos Santos (https://www.linkedin.com/in/lucasmarioza/)
- Rodrigo Rodrigues de Novaes Junior (http://lattes.cnpq.br/5192856393208593)

Boca Configurado por:

- Felipe Duarte dos Reis
- Andrei Rimsa Álvares (http://lattes.cnpq.br/0531001850708530)

Belo Horizonte
25 de Outubro de 2019

Instruções

1. **Sobre a prova:** Este caderno de prova contém 6 questões. As páginas estão numeradas, a contar desta de 1 à 9. Verifique se o seu caderno está completo. A prova tem duração de 3 horas, com 30 minutos o placar será congelado. Com 15 minutos o juiz não responderá mais as submissões, deixando o resultado final para o encerramento da competição. O time terá a sua disposição um computador por dupla. Não é permitida a utilização do computador ao lado;

2. **Sobre a nomenclatura dos arquivos:** Suas soluções **devem** estar nomeadas LETRA.ext, onde LETRA é a letra, em caixa alta, que identifica o problema, e "ext" a extensão correspondente a linguagem escolhida (c, cpp, java, py). Em Java, **deve** haver uma classe pública, com o nome do arquivo. A linha "package" no cabeçalho do arquivo deve ser OMITIDA;

3. **Sobre a entrada:** Ela deve ser lida da entrada padrão. Atente-se para a formatação de cada problema;

4. **Sobre a saída:** A saída deve ser escrita na saída padrão. Atente-se para a formatação, quantidade de espaços, pontuação, quebra de linha, etc;

5. **Sobre o tempo de execução:** O tempo de execução está assinalado em cada problema, para a linguagem C, C++. Para Java/Python, considere o tempo de execução 3 vezes maior;

6. **Sobre a submissão dos problemas:** A submissão deve ser realizada pelo sistema web BOCA, no host, com o login e senha fornecido pelos juízes. Não tente submeter programas maliciosos para travar o juiz, isso acarretará em eliminação da competição e, para membros da instituição sede, demais sanções cabíveis.

7. **Sobre a consulta:** É permitida a consulta a qualquer material impresso, desde que você tenha começado a prova com ele. Não é permitido o uso de qualquer aparelho eletrônico ao longo da prova. O descumprimento destas normas acarreta em eliminação da competição.

8. **Demais orientações:** Caso tenha alguma dúvida ao longo da competição, você pode perguntar ao fiscal que estará na sala. Você também pode pedir um esclarecimento aos juízes através da aba *Clarifications* no sistema, que responderão o mais rápido possível. Se você leu as orientações da prova até aqui, escreva um comentário no cabeçalho da sua primeira submissão, com o número "42".

9. **Boa sorte e divirta-se!**

A - Padrão Textual

A.[c/cpp/java/py]
Limite de tempo: 1 segundo
Autor: André Rodrigues da Cruz

Matheus Pink é um estagiário cuja fama se alastrou rapidamente dentro da empresa *Take* por colaborar bastante na criação de soluções que facilitam a comunicação entre empresas e pessoas. Segundo o que contam nos corredores da empresa, o Pink Boy (alcunha recebida pelos colegas de serviço) é fascinado por programação dinâmica. É comum nos papos do café da tarde ouvir dele: "*Eu curto muito resolver problemas dos quais a solução ótima final é computada de outras ótimas, de subproblemas que compõem o problema original, calculadas e memorizadas previamente. Evitar recálculo é o que há! Assim eu evito a fadiga.*"

Pois bem… essa reputação chegou aos ouvidos do chefe dele, o Jefinho, que tinha planos em mente sobre uma nova funcionalidade do BLiP: uma ferramenta para pesquisar palavras que têm certos padrões nos diálogos dos chatbots. Basicamente são apresentados uma frase de um diálogo e um padrão dado por uma cadeia de caracteres com letras latinas minúsculas, '?' para representar uma única letra qualquer e '*' para representar uma sequência de quaisquer letras (podendo ser vazia). Por exemplo, suponha a frase "Ela ama amora com ameixa agora". Para o padrão "a*a" as palavras "ama", "amora", "ameixa" e "agora" devem ser consideradas. Já para o padrão "a?a", apenas a palavra "ama" deve ser acatada.

Pink Boy recebeu a missão de desenvolver tal solução. Entretanto, ele precisa de sua ajuda, pois está muito atarefado também com as apresentações da Semana de Ciência e Tecnologia e da Mostra Específica de Trabalhos e Aplicações do CEFET-MG.

Entrada
A entrada possui duas linhas. Na primeira há uma frase com até 255 caracteres (letras e espaços). Na segunda linha há uma cadeia de caracteres que representa o padrão de interesse.

Saída
A saída possui uma linha com o número de palavras da frase que combina com o padrão desejado.

Exemplos

Entrada	Saída
o rato roeu a roupa do rei de roma na ratoeira r*t*	2
ana e andreia gostam de anagrama e antologia a*a	4
bia lia e tia riam quando o gato caiu na pia ?ia*	5
acho que o anciao apaixona atrasado ante o anuncio a*i?o	1

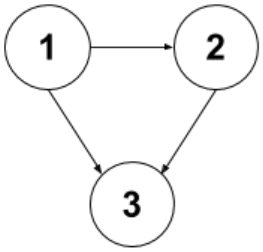
B - Cadeado morto

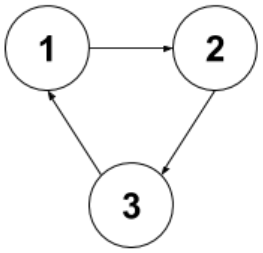
B.[c/cpp/java/py]
Limite de tempo: 1 segundo
Autores: Felipe Duarte dos Reis, Lucas Marioza

O BLiP é uma plataforma muito complexa: faz centenas de milhares de trocas de mensagens por segundo, escritas no banco, análise de texto, e suas redes neurais formigam a todo vapor. Para cada uma dessas tarefas existe uma *thread*, e todas elas compartilham recursos. Calma! Você não vai precisar escrever um programa multithread para resolver este problema, jovem afoito!

O que ocorre é que o Mestre Pacheco, que trabalhou no desenvolvimento do BLiP conhece de cór e salteado todas as *threads* do programa, e as relações de dependência entre elas. Ele sabe que só existe uma instância de um determinado recurso, e que essa instância só pode ser acessada por uma *thread* por vez. Quando uma *thread* terminou de utilizá-lo, o recurso passa a *thread* seguinte que espera por ele.

O time de desenvolvedores da Take quer saber se um par de *thread* podem se bloquear mutuamente por tempo indeterminado, a espera de recurso. No exemplo abaixo, à esquerda, a seta de 1 para 2 indica que a *thread* 1 espera por um recurso que está com a *thread 2*. Neste caso, elas não se bloqueiam mutuamente pois 1 e 2 estão esperando por um recurso que está com a *thread* 3, que por sua vez não espera por ninguém e pode executar. Quando ela terminar, a 2 pode executar, e por fim a 1. No caso da direita elas podem se bloquear, uma vez que 1 espera por 2, que está esperando por 3, que espera por 1, e nenhuma delas pode prosseguir a execução.





Como o Mestre Pacheco está viajando, ele deixou este problema para você resolver!

Entrada

A entrada é composta de dois números inteiros N, ($2 \leq N \leq 5000$) e M ($N \leq M \leq 2 \cdot N$), seguidos de M pares u e v, com $1 \leq u, v \leq N$, onde a *thread* u espera por um recurso vindo da *thread* v.

Saída

Você deve imprimir uma linha contendo a mensagem **CADEADO MORTO** caso as *threads* possam ficar esperando por tempo indeterminado, e **SEM CADEADO** caso contrário.

Exemplos

Entrada	Saída
4 4 1 2 2 3 1 4 3 4	SEM CADEADO
4 4 1 2 2 3 3 4 4 1	CADEADO MORTO
3 2 1 2 2 3 1 3	SEM CADEADO

C - Meu primeiro bot

C.[c/cpp/java/py]
Limite de tempo: 1 segundo
Autores: Felipe Duarte dos Reis

Hoje é o primeiro dia do Mourinho, aluno de engenharia de computação no CEFET-MG, como estagiário na Take. Ele já recebeu um projeto de um cliente, que precisa de um bot simples, mas o menino Mourinho está completamente perdido. O BLiP é uma poderosa plataforma para desenvolver chatbots, mas ela é também muito grande, e o Mourinho está desesperado lendo a documentação e pediu sua ajuda para escrever o código principal do bot. Pra sua sorte, a tarefa não é difícil.

Entrada

A entrada é composta por uma cadeia de caracteres S, |S| < 100.

Saída

A saída é composta pela cadeia de caracteres lida pela entrada, invertida.

Exemplos

Entrada	Saída
`Bom dia`	`aid moB`
`Bem vindo a Maratona Take`	`ekaT anotaraM a odniv meB`
`Boa sorte`	`etros aoB`

D - Vetores de Palavras

D.[c/cpp/java/py]
Limite de tempo: 1 segundo
Autores: Felipe Duarte dos Reis, Rodrigo Novaes

Ninha é uma engenheira do time de Inteligência Artificial da Take. Uma das suas áreas de atuação é o processamento de linguagem natural, na qual é preciso tornar grandes entradas de texto em algo que possa ser entendido e processado pelo computador.

O gerente da Ninha pediu a ela um modelo de linguagem que sirva de entrada para uma rede neural artificial. Que tal ajudá-la?

Dado um vocabulário V e conjunto de N textos $T_1, T_2, ..., T_N$, o vetor de palavras P_i é definido como a quantidade de ocorrências de cada palavra $j \in V$, $1 \leq i \leq N$. De maneira simplificada, P_{ij} representa quantas vezes j ocorreu em T_i.

A Ninha sabe que a vida não é tão gentil com ela, portanto não é esperado que V lhe seja conhecido enquanto converte sua lista de textos em vetores de palavras.

Entrada

A entrada é composta por um inteiro N, que representa a quantidade de textos a serem lidos. A seguir vêm N linhas, cada uma contendo uma quantidade variável de palavras s separadas por um espaço. Os textos são compostos apenas por caracteres alfabéticos da tabela ASCII e são delimitados por uma quebra de linha.

Restrições:

$1 \leq N \leq 1000$

$1 \leq |s| \leq 20$

Saída

A saída é composta pelo vocabulário V, precedida pela expressão "vocabulario: ". Cada palavra deve ser exibida em ordem alfabética e separada por uma vírgula. A seguir, N linhas deverão conter a saída "texto i: ", $1 \leq i \leq N$, na qual a quantidade de ocorrências de cada palavra do vocabulário é contabilizada para o texto correspondente. Cuidado com a ordem alfabética e com espaços adicionais ao fim de cada linha!

Exemplos

Entrada	Saída
2 take on me take me on	vocabulario: me, on, take texto 1: 1, 1, 1 texto 2: 1, 1, 1
2 certo eh certo errado eh errado	vocabulario: certo, eh, errado texto 1: 2, 1, 0 texto 2: 0, 1, 2

E - Onde vamos parar?

E.[c/cpp/java/py]
Limite de tempo: 1 segundo
Autores: Felipe Duarte dos Reis

O time de IA da Take está a todo vapor, e solicitou mais uma demanda para você. Será que você é capaz de ajudá-los?

Um Bot no BLiP possui um estado, e a cada comando que ele recebe, acontece uma transição para outro estado. O problema é que os comandos são digitados pelo usuário no celular, que é frequentemente sabotado pelo corretor ortográfico. Por exemplo: o usuário pode digitar 'abacate', o corretor corrigir pra 'abacaxi' logo antes do envio, e o Bot não vai entender o comando. Bom, isso é fácil de resolver, certo? Para esse comando basta trocar 2 caracteres: 'x' por 't' e 'i' por 'e'. Mas como para cada estado, vários comandos são possíveis, e talvez alguns sejam até parecidos.

Para treinar uma daquelas redes neurais muito complexas para tentar fazer previsões sobre os *inputs* dos usuários, o pessoal do time de IA quer saber, dado uma lista de comandos e as transições de estados, qual o estado final que tem o menor custo total de mudanças na lista de comandos. Veja que para a equipe, substituir, remover ou adicionar um caractere ao comando tem custo unitário.

Entrada

A entrada é composta por três inteiros N ($2 \leq N \leq 1000$), M ($N \leq M \leq N^2$) e L ($3 \leq L \leq 100$), seguidos de M triplas i j e s ($1 \leq i, j \leq N$ *e* $|s| \leq 10$) tal que existe uma transição do estado i para o estado j com o comando s. Por fim, haverá L cadeias de caracteres alfabéticos com no máximo 10 caracteres, separadas por espaços. Considere 1 o vértice inicial.

Saída

A saída é composta por dois números inteiros separados por um espaço, a quantidade total de modificações necessárias para chegar seguido estado final. Caso não seja possível executar essa lista de comandos, a saída deve ser -1.

Exemplos

Entrada	Saída
4 5 4 1 2 chocolate 2 3 morango 1 3 abacate 3 4 abacaxi 4 1 batido abacat tomate abacati chocoliate	13 2
4 5 4 1 2 abacate 2 3 morango 1 3 abacate 3 4 abacaxi 4 1 batido	0 1

abacate morango abacaxi batido	
4 4 3 1 2 a 1 3 a 2 4 a 3 4 a a b c	-1

F - Risômetro

F.[c/cpp/java/py]
Limite de tempo: 1 segundo
Autores: Felipe Duarte dos Reis, Rodrigo Novaes

O Matheus Pink, o mais novo engenheiro de software da *Take*, é um menino divertido e descontraído. Nos *happy hours* da empresa, ele se diverte contando piadas para os colegas de trabalho. Na verdade, somente ele se diverte, porque a maioria ri muito pouco das piadas dele.

Pensando em melhorar a qualidade das piadas, e com a cabeça de um recém formado engenheiro de computação, ele quer criar um chatbot para contar piadas no chat da empresa e medir o efeito delas nas pessoas. Ele leu em um artigo que o professor Rogerinho, doutor em IAPR (Inteligência Artificial e Piadas Ruins), passou para ele que quanto maior o período da risada, mais engraçada a piada.

Como o jovem Matheus Pink está muito ocupado terminando o TCC, ele pediu a sua ajuda para medir o período das risadas que o Bot contador de piadas recebeu.

Entrada

A entrada é composta por uma cadeia de caracteres alfabéticos S, $1 \leq |S| \leq 256$.

Saída

A saída é composta por um inteiro T, que representa o maior período da risada. Para uma risada aperiódica, a saída deve ser 0.

Exemplos

Entrada	Saída
huehuehue	3
kkkkkkkkk	1
hiuasoiajshaisduh	0