**Universidade de Brasília**

Departamento de Ciência da Computação

Introdução à Ciência da Computação - 113913

Lista de Exercícios 6

Listas

**Observações:**

* As listas de exercícios serão corrigidas por um **corretor automático,** portanto é necessário que as entradas e saídas do seu programa estejam conforme o padrão especificado em cada questão (exemplo de entrada e saída). Por exemplo, a não ser que seja requisitado na questão, não use mensagens escritas durante o desenvolvimento do seu código como “Informe a primeira entrada”. Estas mensagens não são tratadas pelo corretor, portanto a correção irá resultar em **resposta errada**, mesmo que seu código esteja correto;
* As questões estão em **ordem de dificuldade.** Cada lista possui 7 exercícios, sendo 1 questão fácil, 3 a 4 médias e 2 a 3 difíceis;
* Leia com atenção e faça **exatamente** o que está sendo pedido.

Acesso Remoto

Arborilda é uma jovem dona de uma loja de jogos de mesa que tem crescido bastante nos últimos meses. Até então, Arborilda tem se organizado usando papel e lápis, mas é cada vez mais difícil executar as manipulações necessárias nas requisições de produtos com o crescente número de clientes a importunando.

Conhecendo sua reputação e seu conhecimento em Python, Arborilda pede a sua ajuda para ajudá-la a se organizar melhor.

A primeira coisa que Arborilda faz ao chegar ao trabalho de manhã cedo, é reler todas as requisições de produtos, da mais recente para a mais antiga.

Entrada

A primeira linha da entrada consiste de um inteiro N, com o número de requisições a serem processadas. As N seguintes linhas contêm, cada uma, uma string P, o nome do produto requisitado.

**Saída**

Seu programa deve imprimir uma única linha contendo os jogos da entrada, porém na ordem invertida, separados por vírgula e um único espaço.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  2  Dixit  Carcassone | **Saída**  Carcassone, Dixit |

|  |  |
| --- | --- |
| **Entrada**  4  Magic: The Gathering  Yu-Gi-Oh!  Pokémon: TCG  Cardfight!! Vanguard | **Saída**  Cardfight!! Vanguard, Pokémon: TCG, Yu-Gi-Oh!, Magic: The Gathering |

Bella e seus amigos

Bella é uma pessoa muito popular e agradável, e tem muitos, muitos amigos. Só que desde que voltou de viagem, seu amigo André tem sido bastante inconveniente e a tem perturbado bastante com piadas inapropriadas e invasivas.

Para resolver a situação, Bella decidiu que sempre que fosse para um evento ou festa, olharia primeiro a lista de convidados para saber se seu amigo André estaria presente. Mas como essas festas normalmente têm extensas listas de convidados, Bella está tendo problemas para verificá-las uma a uma manualmente.

Conhecendo a Bella e sabendo do seu dilema, você se prontificou para auxiliá-la, escrevendo um programa python que processa a lista dos convidados e a responde se é seguro ir.

Entrada

A primeira linha da entrada consiste em um inteiro C, o número de convidados da festa em questão. As próximas C linhas contêm, cada uma, uma string não-vazia, o primeiro nome de um convidado.

**Saída**

Seu programa deve imprimir uma única linha com “Cuidado!” ou “Seguro!”, se o André estiver na lista de convidados, ou não, respectivamente.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  4  André  George  Julia  Diego | **Saída**  Cuidado! |

|  |  |
| --- | --- |
| **Entrada**  5  Roberto  Alberron  Andrezildo  Abacatilson  Georgina | **Saída**  Seguro! |

Cake Store

Beijo da Vó é um loja de bolos que a dona Sílvia acabou de abrir e está fazendo uma promoção de inauguração.

Todo bolo que ela vende nos próximos três meses terá uma fatia premiada, e quem pegá-la ganha um carro.

Mas a dona Sílvia está tendo problemas de gerenciar quem pegou a fatia premiada, pois a clientela anda volumosa.

Sabendo das suas habilidades computacionais, a dona Sílvia pediu sua ajuda para escrever um programa que a responda, para cada bolo, quem pegou a fatia premiada e foi o fatídico ganhador de um carro zero.

Entrada

A primeira linha da entrada consiste em dois inteiros F e P, correspondentes ao número de fatias, e o índice (iniciando em zero) da fatia premiada.

As próximas F linhas contêm, cada uma, uma string não-vazia N e um inteiro E, o primeiro nome da pessoa e o índice da fatia escolhida, respectivamente.

Note que os índices das fatias são atualizados toda vez que alguém retira um pedaço.

**Saída**

Seu programa deve imprimir na tela o nome da pessoa que recolheu a fatia premiada.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  4 3  Roberto 2  Julia 2  Umbreon 0  Blackout 0 | **Saída**  Julia |

|  |  |
| --- | --- |
| **Entrada**  3 0  Abacatilson 0  Vigário 0  Joelma 0 | **Saída**  Abacatilson |

Déficit de Memória

André é uma criança perturbada que tem déficit de memória recente. Acontece que ele também tem uma quantidade enorme de brinquedos, e gosta de organizá-los das mais diversas formas.

Curioso com a sua doença e sua mania de arrumação, André inventou a seguinte brincadeira:

No início da semana, André escreve a configuração dos brinquedos na sua prateleira.  
Uma vez por dia, durante cinco dias, André vai até a sua prateleira e move um dos seus brinquedos de lugar zero ou mais posições, empurrando os outros conforme necessário.

Ao final dos cinco dias, André recupera sua anotação do início da semana e verifica o que tem de diferente para a configuração final.

Depois de esquecer de anotar a configuração inicial três vezes, André decide contratar um programador experiente para ajudá-lo na sua brincadeira, e entra em contato com você.

Entrada

A primeira linha da entrada contém um inteiro N, o número de brinquedos na prateleira.

A próxima linha contém N caracteres diferentes separados por espaço, os identificadores de cada um dos brinquedos na prateleira, em ordem.

As próximas 5 linhas contêm, cada uma, um caractere B, o brinquedo a ser movido; um caractere D, a direção em que ele será movido, podendo ser ‘E’ (para esquerda), ou ‘D’ (para direita); e um inteiro Q, a quantidade de espaços que o brinquedo será movido.

**Saída**

Seu programa deve imprimir uma única linha, contendo o número de brinquedos fora dos seus lugares, ao fim dos 5 dias.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  3  A B C  A D 2  A E 1  B D 1  A D 0  C E 0 | **Saída**  0 |

|  |  |
| --- | --- |
| **Entrada**  4  X Y Z W  X D 3  Y D 1  W E 1  Z E 0  W E 1 | **Saída**  4 |

Elastiman

Roberto é um game designer muito famoso na região onde mora por ter lançado alguns jogos de sucesso, mas nunca estabeleceu uma empresa de fato.

Há duas semanas, Roberto chegou em você com a sua mais nova ideia revolucionária de jogo, Elastiman! Sem saber a quem chamar para ajudá-lo a desenvolver, ele recorreu ao programador mais próximo, que aconteceu de ser você.

Seu trabalho na primeira versão do jogo é desenvolver a parte da engine que lida com gravidade.

Entrada

A primeira linha da entrada contém um inteiro N.

As próximas N linhas conterão uma matriz N x N, descrevendo o cenário atual:

cada `.` corresponde a um espaço vazio;

cada `x` corresponde a um bloco fixo, que não está sujeito à gravidade;

cada `o` corresponde a um bloco móvel, que está sujeito à gravidade.

**Saída**

Seu programa deve imprimir a tela após um ciclo do loop de jogo, após a execução da gravidade. Para simplificar sua vida, considere a gravidade uma força que só puxa um espaço para baixo por loop, não uma força física *de facto*.

Note que blocos móveis não podem atravessar blocos fixos.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  3  . . .  . o .  x x x | **Saída**  . . .  . o .  x x x |

|  |  |
| --- | --- |
| **Entrada**  3  . o .  . . .  x x x | **Saída**  . . .  . o .  x x x |

Florêncio Pede Ajuda

Florêncio é um jovem programador Ruby que está aprendendo Java e está tendo dificuldade em não somente achar a linguagem palatável, como se acostumar ao padrão camelCase.

Sabendo do versátil programador que é, Florêncio pede sua ajuda para fazer um conversor de snake\_case para CamelCase.

Entrada

A primeira e única linha da entrada consiste em uma palavra ou frase em snake\_case, ou seja, todas as letras minúsculas, separadas por *underscore* (\_).

**Saída**

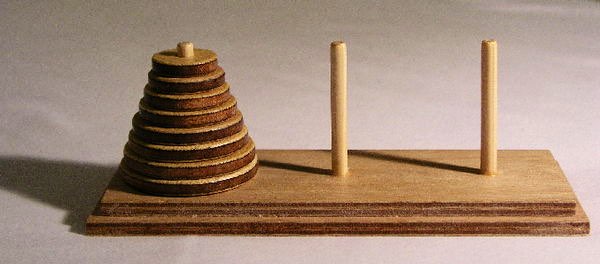
Seu programa deve imprimir uma única linha, contendo a entrada convertida para CamelCase, ou seja, todas as letras minúsculas, exceto a primeira letra de cada palavra.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  snake\_case | **Saída**  SnakeCase |

|  |  |
| --- | --- |
| **Entrada**  create\_underscored\_book\_cock\_tail | **Saída**  CreateUnderscoredBookCockTail |

Hanoi

A Torre de Hanoi é um jogo matemático que data de 1883, mas há lendas de sua existência desde a criação do mundo.

O objetivo do jogo é trazer todos os discos da haste esquerda para a haste direita seguindo as seguintes três regras simples:

* Apenas um disco pode ser movido por vez;
* Cada movimento consiste em retirar o disco que está mais acima em uma das hastes, e o colocar no topo de outra haste;
* Nenhum disco pode ser colocado sobre um disco menor.

Seu objetivo é criar um simulador da solução mais otimizada para este puzzle.

Entrada

A entrada consiste de apenas dois inteiros H e P, descrevendo o número de discos da torre de Hanoi e o número de passos desejados, respectivamente.

**Saída**

Seu programa deve simular a solução ótima do puzzle e parar após a execução de P passos. Ao final da execução, ele deve imprimir na saída padrão três inteiros, cada um descrevendo a quantidade de discos em cada torre após P passos.

**Exemplos**

|  |  |
| --- | --- |
| **Entrada**  4 3 | **Saída**  2 0 2 |

|  |  |
| --- | --- |
| **Entrada**  3 1 | **Saída**  2 0 1 |