Competidor(a): _____

Número de inscrição: _____–_____ *(opcional)*

OBI2020

Caderno de Tarefas

Modalidade **Programação** • **Nível Júnior** • Fase **Estadual**

24 de outubro de 2020

A PROVA TEM DURAÇÃO DE **2** HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 6 páginas (não contando a folha de rosto), numeradas de 1 a 6. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar a Internet, livros, anotações ou qualquer outro material durante a prova. É permitida a consulta ao *help* do ambiente de programação se este estiver disponível.
- As tarefas têm o mesmo valor na correção.
- A correção é automatizada, portanto siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa; em particular, seu programa não deve escrever frases como "Digite o dado de entrada:" ou similares.
- Não implemente nenhum recurso gráfico nas suas soluções (janelas, menus, etc.), nem utilize qualquer rotina para limpar a tela ou posicionar o cursor.
- As tarefas **não** estão necessariamente ordenadas, neste caderno, por ordem de dificuldade; procure resolver primeiro as questões mais fáceis.
- Preste muita atenção no nome dos arquivos fonte indicados nas tarefas. Soluções na linguagem C devem ser arquivos com sufixo *.c*; soluções na linguagem C++ devem ser arquivos com sufixo *.cc* ou *.cpp*; soluções na linguagem Pascal devem ser arquivos com sufixo *.pas*; soluções na linguagem Java devem ser arquivos com sufixo *.java* e a classe principal deve ter o mesmo nome do arquivo fonte; soluções na linguagem Python 2 devem ser arquivos com sufixo *.py2*; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py3*; e soluções na linguagem Javascript devem ter arquivos com sufixo *.js*.
- Na linguagem Java, **não** use o comando *package*, e note que o nome de sua classe principal deve usar somente letras minúsculas (o mesmo nome do arquivo indicado nas tarefas).
- Para tarefas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada tarefa.
- Ao final da prova, para cada solução que você queira submeter para correção, copie o arquivo fonte para o seu diretório de trabalho ou pen-drive, conforme especificado pelo seu professor.
- Não utilize arquivos para entrada ou saída. Todos os dados devem ser lidos da entrada padrão (normalmente é o teclado) e escritos na saída padrão (normalmente é a tela). Utilize as funções padrão para entrada e saída de dados:
  - em Pascal: *readln, read, writeln, write*;
  - em C: *scanf, getchar, printf, putchar*;
  - em C++: as mesmas de C ou os objetos *cout* e *cin*.
  - em Java: qualquer classe ou função padrão, como por exemplo *Scanner, BufferedReader, BufferedWriter* e *System.out.println*
  - em Python: *read,readline,readlines,input,print,write*
  - em Javascript: *scanf, printf*
- Procure resolver a tarefa de maneira eficiente. Na correção, eficiência também será levada em conta. As soluções serão testadas com outras entradas além das apresentadas como exemplo nas tarefas.

Dona Lesma

Nome do arquivo: "`lesma.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

Dona Lesma é esportista e aventureira e definiu como objetivo deste verão alcançar o topo do muro do jardim em que vive. A cada dia, valente e metodicamente ela sobe exatamente uma certa distância (sempre a mesma a cada dia). Mas a cada noite enquanto dorme Dona Lesma escorrega para baixo uma outra distância (sempre a mesma a cada noite)...

Dadas a altura do muro, a distância que ela sobe a cada dia e a distância que ela desce a cada noite, ajude Dona Lesma a calcular quantos dias ela levará para chegar ao topo do muro.

Entrada

A primeira linha contém um inteiro A, a altura do muro. A segunda linha contém um inteiro S, distância que Dona Lesma sobe a cada dia. A terceira linha contém um inteiro D, a distância que Dona Lesma escorrega para baixo a cada noite.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número de dias que Dona Lesma demorará para chegar ao topo do muro.

Restrições

- $1 \leq A \leq 10000$
- $1 \leq D < S \leq 10000$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
4 2 1	3

Exemplo de entrada 2	**Exemplo de saída 2**
12 5 2	4

Exemplo de entrada 3	**Exemplo de saída 3**
10000 100 50	199

Palavras Cruzadas

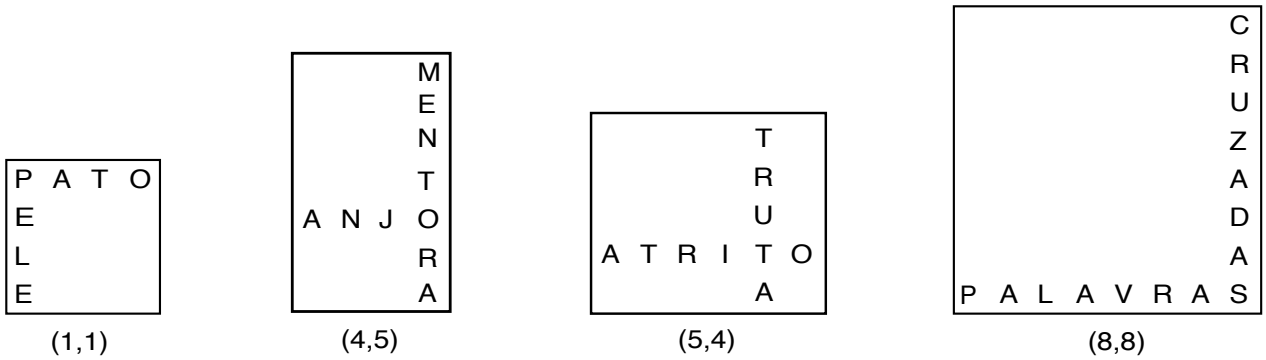
Nome do arquivo: "`cruzadas.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

Você já deve ter tentado completar um jogo de palavras cruzadas. Nesse jogo, o jogador deve descobrir um conjunto de palavras através de dicas fornecidas juntamente com um retângulo dividido em quadrados de mesmo tamanho, sendo a maoria quadrados em branco e alguns quadrados pretos. Cada linha e cada coluna formada pelos quadrados em branco deve ser preenchida por uma palavra, com uma letra em cada quadrado em branco. As palavras das linhas são chamadas de palavras horizontais, as palavras das colunas são chamadas palavras verticais. As palavras horizontais *cruzam* com as palavras verticais em uma letra comum às duas palavras, vindo daí o nome do jogo.

Nesta tarefa, dadas uma palavra horizontal e uma palavra vertical, você deve encontrar a *melhor* letra de cruzamento entre elas, que é definida como a letra que produz

1. o cruzamento mais à direita possível na palavra horizontal;
2. se há mais de uma possibilidade de cruzamento com a regra (1), a melhor letra de cruzamento é definida como a que produz o cruzamento mais abaixo possível na palavra vertical.

A figura abaixo ilustra alguns exemplos. Entre parênteses estão os índices da melhor letra de cruzamento. O índice de uma letra é a posição que ela ocupa na palavra, iniciando com 1 (primeira letra).



Entrada

A primeira linha da entrada contém a palavra horizontal. A segunda linha da entrada contém a palavra vertical.

Saída

Seu programa deve produzir uma única linha, contendo apenas dois inteiros descrevendo a melhor letra de cruzamento. O primeiro número deve ser o índice da letra de cruzamento na palavra horizontal, o segundo número deve ser o índice da letra de cruzamento na palavra vertical. Se não há letra de cruzamento possível, os dois inteiros devem ser iguais a -1.

Restrições

- As palavras são compostas por letras maiúsculas não acentuadas, com comprimento de pelo menos uma letra e de no máximo 1000 letras.

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 20 pontos há exatamente uma possibilidade de cruzamento possível.
- Para um conjunto de casos de testes valendo 80 pontos adicionais não há restrições adicionais.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
PATO PELE	1 1

Exemplo de entrada 2	**Exemplo de saída 2**
ANJO MENTORA	4 5

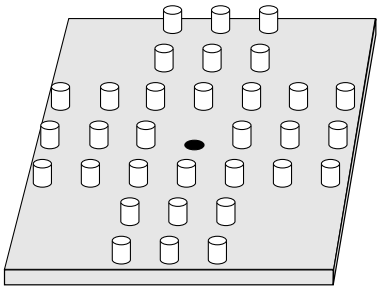
Exemplo de entrada 3	**Exemplo de saída 3**
MENTORA ANJO	7 1

Exemplo de entrada 4	**Exemplo de saída 4**
URUBU POLIVALENTE	-1 -1

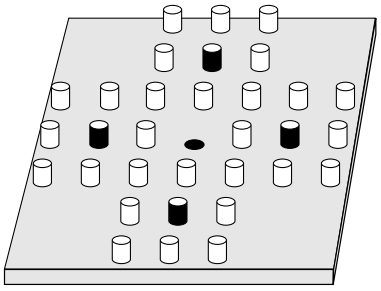
Jogo dos Pinos

Nome do arquivo: "`pinos.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

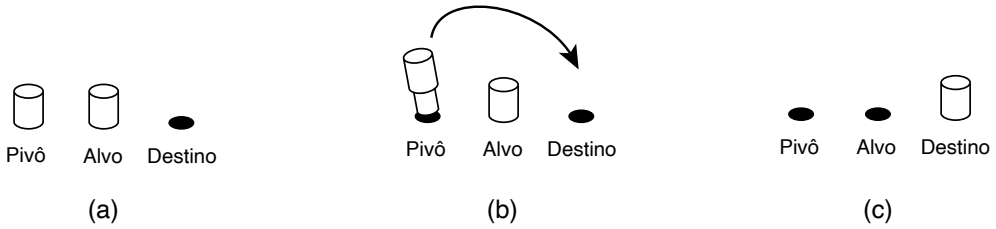
O Jogo dos Pinos é um quebra-cabeças que utiliza pinos e um tabuleiro com furos em forma de cruz. Inicialmente há apenas um furo vago, no centro do tabuleiro, e todos os outros furos contém um pino como mostra figura abaixo.



O objetivo do jogo é remover os pinos do tabuleiro de forma que reste apenas um pino. Para remover um pino é necessário fazer um *movimento válido*, que é definido da seguinte maneira. O jogador deve escolher um pino, chamado *pivô*, e uma das quatro direções (acima, abaixo, esquerda, direita) de tal forma que o pivô tenha um outro pino, chamado *alvo*, como vizinho imediato na direção escolhida e que o pino alvo seja seguido, também na direção escolhida, por um furo vago (chamado de *destino).* A figura abaixo mostra os quatro possíveis pivôs da configuração inicial do jogo.



O jogador pode então fazer o pino pivô pular sobre o pino alvo e ocupar o furo destino, removendo o pino alvo do tabuleiro. A figura abaixo mostra um exemplo (a) antes, (b) durante e (c) depois de um movimento válido.



Dada uma configuração de pinos em um tabuleiro, escreva um programa para determinar o número de movimentos válidos possíveis na configuração dada.

Entrada

A entrada é composta por sete linhas, cada linha com exatamente sete caracteres. A linhas são identificadas por números de 1 a 7. Os dois primeiros caracteres e os dois últimos caracteres das linhas 1, 2, 6 e 7 são '`-`' (hífen). Todos os outros caracteres são ou '`o`' (letra o minúscula) representando um pino, ou '`.`' (ponto) representando um furo.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número de movimentos válidos na configuração da entrada.

Restrições

- A seção Entrada descreve as restrições.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**

`--ooo--`
`--ooo--`
`ooooooo`
`ooo.ooo`
`ooooooo`
`--ooo--`
`--ooo--`
	`4`

Exemplo de entrada 2	**Exemplo de saída 2**

`--.o.--`
`--o.o--`
`....o..`
`....o..`
`o.o.o..`
`--o.o--`
`--o.o--`
	`2`