Tarefa de Laboratório 04 - Funções - Parte 1

Suporte para recuperação de senha de um terminal

Em um mundo em que a tecnologia estava avançando cada vez mais, a necessidade de ter um sistema eletrônico para a garantir a segurança da sua residência tornou-se real. A MAGAMI é uma empresa pioneira em fornecer todo um arsenal para a defesa de sua propriedade: travas eletrônicas, redes que capturam invasores, gás sonífero e, na pior das hipóteses, trancamento total da residência.

A oferta foi logo aceita pelo mercado, com um número altíssimo de compradores interessados, mas, em menos de uma semana, diversos relatos sobre os próprios moradores das residências sendo alvo das defesas instaladas, com várias equipes técnicas da MAGAMI tendo que ser despachadas para resolver os problemas. Após uma análise dos fatores, descobriu-se que o erro decorria do sistema inicial **não possibilitar** que o usuário errasse a senha mestre para desativação das defesas.

Após uma análise que levou em consideração tanto os fatores humanos (de esquecer a senha mestre, por exemplo) quanto ainda de manter a segurança, a MAGAMI optou em contratar você para realizar uma atualização do sistema que permitirá que o usuário erre a senha algumas vezes antes das defesas serem ativadas. Além disso, para facilitar que o proprietário lembre sua senha, uma dica deverá ser exibida a cada tentativa que indica a **semelhança** da senha digitada com a original. Essa semelhança é dada pelo *número de dígitos que estão na mesma posição da senha original*.

Por exemplo, se quisermos comparar a semelhança entre os números `2345678` e `2795273`, temos:

```
Número 1: 2 3 4 5 6 7 8
Número 2: 2 7 9 5 2 7 3
          ^     ^   ^
```

A semelhança seria `3` pois existem 3 dígitos na mesma posição em ambos os números.

Descrição da entrada

Seu programa será composto pela configuração inicial do sistema e pelo reconhecimento da senha pelo usuário. A entrada é dividida da seguinte forma:

- A primeira linha é composta por dois números inteiros positivos: a **senha mestre** e o **número de tentativas para descobrí-la (T)**, separados por um espaço em branco. Considere que **T** sempre será maior que 0.
- As próximas entradas são compostas por, no máximo, **T** linhas correspondentes às entradas de um usuário tentando adivinhar a senha (uma tentativa por linha). O número de dígitos de uma entrada do usuário **não** necessariamente corresponderá ao número de dígitos da senha mestre.

Um exemplo de entrada seria o seguinte (EX1):

```
1234 4
4324
1252
1234
```

Outro exemplo pode ser (EX2):

```
159 2
2222
789
```

Descrição da saída

A saída do seu programa deverá considerar as respostas a *cada uma* entradas das senhas pelo usuário. Devem obedecer ao seguinte formato:

Caso o usuário erre a senha, uma mensagem de três linhas deve ser impressa:

- **Senha incorreta**
- **Semelhanca:** Informar a semelhança da senha informada em relação com a senha mestre. Isso só pode ser possível caso **o número de dígitos de ambas senhas informadas e mestre seja igual**. Caso isso não ocorra, a mensagem "**Erro: quantidade de digitos incongruente**" deve ser informada como semelhança (sem as aspas).
- **Tentativas restantes:** Informar o número de tentativas restantes pelo usuário.
- Caso o usuário erre a última tentativa, a mensagem "**Tentativas esgotadas. Acionando defesas...**" também deve ser informada em uma nova linha (sem aspas).

Caso o usuário acerte a senha, somente deve ser impresso:

- **Senha reconhecida. Desativando defesas...**

A cada mensagem de saída para uma entrada, seu programa deve pular uma linha.

A saída do EX1 seria:

```
Senha incorreta
Semelhanca: 1
Tentativas restantes: 3

Senha incorreta
Semelhanca: 2
Tentativas restantes: 2

Senha reconhecida. Desativando defesas...
```

Já a saída do EX2 seria:

```
Senha incorreta
Semelhanca: Erro: quantidade de digitos incongruente
Tentativas restantes: 1

Senha incorreta
Semelhanca: 1
Tentativas restantes: 0

Tentativas esgotadas. Acionando defesas...
```

Importante: *Inteiros e zeros à esquerda*

No dia-a-dia estamos acostumados a representar números inteiros com zeros à esquerda em algumas situações. No entanto, uma representação com números assim em Python está incorreta. Você pode verificar isso pela `shell`:

```
>>> a = 7
>>> a
7
>>> b = 07
  File "<stdin>", line 1
    b = 07
        ^
SyntaxError: leading zeros in decimal integer literals are not permitted; use an 0o prefix for octal integers
```

Uma `string` representando um número com zero à esquerda é permitido porque o Python trata como `string` e não `int`. Se você converter tal `string` em `int`, o Python automaticamente elimina os zeros à esquerda:

```
>>> a = '007'
>>> a
'007'
>>> b = int(a)
>>> b
7
```

Repare que, de modo geral, uma senha não teria problemas em ter zeros à esquerda, podendo ser inclusive `0000` por exemplo. No entanto, para evitar problemas com a resolução desta tarefa, considere que **nenhuma senha terá o dígito 0 em qualquer posição** ou seja, a senha mestre e as digitadas pelo usuário **só irão conter dígitos de 1 a 9**.
