MC102QR - Algoritmos e Programação de Computadores

Lab 7: Homem-aranha sem casa

Prazo: 22 de Maio de 2022.
Peso na nota: 3 (7,44%)

Peter Parker e Ned conseguiram resolver a situação com o Duende Verde, porém Peter precisou fazer alguns sacrifícios que fizeram com que ele perdesse todas as ligações com seus entes queridos e também a ajuda financeira da Tia May. Basicamente, Peter perdeu tudo, está morando de aluguel e agora precisa trabalhar para seguir com sua vida universitária que está se iniciando.

Para conseguir algum dinheiro, Peter viu uma oportunidade de trabalho em um jornal sensacionalista chamado Clarim Diário. Sua tarefa consistia em vender fotos de crimes da vizinhança combatidos por ele mesmo, o Homem-Aranha. A ideia que ele teve foi tirar fotos dos crimes, fotos do Homem-Aranha e de tudo que envolvia esses cenários.

Ao colocar essa ideia em prática, Peter encontrou alguns problemas para organizar suas fotos e suas tarefas. Havia várias fotos repetidas, fotos com erros de digitalização e fotos que poderiam ser descartadas. Além de tudo isso, Peter estava com muitas tarefas da universidade para serem feitas, impossibilitando que ele fizesse uma triagem manual destas fotos.

Para resolver essas situações de sua vida cotidiana, Peter pensou em automatizar seus processos utilizando o Python a seu favor. Existe uma lista de strings com os nomes dos arquivos de fotos que precisa ser tratada para que ele entregue o álbum ao jornal. Nessa lista de strings, cada item tem um prefixo que pode ser **"HA_"** ou **"CR_"** ou **"CC_"** representando, respectivamente, "HOMEM-ARANHA", "CRIMINOSO" e "CENA DO CRIME" que são as categorias que Peter determinou para as fotos. Seguido desse prefixo, a foto tem seu título.

Tarefa

Fazer um programa que cumpra as seguintes tarefas:

1 - Retornar qual elemento aparece mais vezes de forma consecutiva na lista, para ele ter um maior controle em fotos futuras. Por exemplo, na lista

```
['HA_SALTO DO POSTE DE ENERGIA', 'HA_SALTANDO', 'CR_DR
OCTOPUS', 'CR_DR OCTOPUS', 'CR_atacando Civis', 'CC_Foto
Da Paisagem', 'CC_AJUDANDO idosa', 'CC_AJUDANDO idosa',
'CC_AJUDANDO idosa']
```

o elemento "`CC_AJUDANDO idosa`" aparece 3 vezes de forma consecutiva.

Não haverá empate de quantidades de itens nessa tarefa.

2 - Contar quantos elementos únicos existem nessa lista, para verificar a quantidade de fotos únicas;

- A posição da primeira vez em que o item aparece é a que precisa ser preservada

3 - Remover todas as ocorrências de determinado elemento na lista, excluindo fotos que precisam ser descartadas ou que estão com erro;

4 - Excluir todos os elementos que se repetem na lista, deixando somente um elemento de cada item;

Terminadas estas tarefas, Peter precisa tratar os títulos dos arquivos de foto que estão nessa lista. É preciso:

- Remover todos os espaços (' ') existentes em cada string da lista e substituí-los por '-'.
- Deixar a string de cada item da lista em letra minúscula **exceto** o prefixo. Por exemplo: a string "*CC_foto-Do-CRIME-na-Ponte*" seria convertida em "*CC_foto-do-crime-na-ponte*";

- Categorizar as fotos em "HOMEM-ARANHA", "CRIMINOSO", "CENA DO CRIME" de acordo com os prefixos citados anteriormente. Assim, será necessário separar a lista em 3 listas finais de acordo com estas 3 categorias.

Entrada

O código vai receber como entrada:

- Uma lista única de strings em que os itens da lista estão separados por “, ” (vírgula + espaço) e o item a ser removido será o último, separado por “/ ” (barra + espaço)

Saída

- Qual elemento aparece mais vezes de forma consecutiva na lista, seguido da quantidade de vezes que aparece
- Quantidade de elementos únicos na lista
- 3 listas (em linhas separadas) de acordo com as 3 categorias citadas no enunciado
    - caso não haja nenhum item em determinada categoria, a lista deverá ser impressa vazia na forma “[]”

Obs. 1: É preciso fazer o uso da estrutura “if __name__=="__main__":” do Python.
Obs. 2: Está **proibido** o uso das estruturas set e dict bem como do método ***count()*** e de bibliotecas externas (import) do Python.
Obs. 3: O formato das saídas deve seguir o formato dos exemplos.

Exemplos

Exemplo 1:

Entrada

```
HA_SALTO DO POSTE DE ENERGIA, HA_SALTANDO, CR_DR OCTOPUS, CR_DR
OCTOPUS, CR_atacando Civis, CC_Foto Da Paisagem, CC_AJUDANDO
idosa, CC_AJUDANDO idosa, CC_AJUDANDO idosa/ CC_Foto Da Paisagem
```

Saída

```
CC_AJUDANDO idosa 3
6
```

```
['HA_salto-do-poste-de-energia', 'HA_saltando']
['CR_dr-octopus', 'CR_atacando-civis']
['CC_ajudando-idosa']
```

—----------

Exemplo 2:

Entrada

```
CR_atingido pela tampa de bueiro, HA_TOMANDO UM-GOLPE, HA_TOMANDO
UM-GOLPE, HA_TOMANDO UM-GOLPE, HA_TOMANDO UM-GOLPE, HA_TIMES
SQUARE, HA_SALTANDO, CR_DR OCTOPUS, HA_TIMES SQUARE, CR_DR
OCTOPUS, CR_atingido pela tampa de bueiro 2, CC_POLICIA CHEGANDO,
HA_TIMES SQUARE, HA_ATAQUE/ HA_TIMES SQUARE
```

Saída

```
HA_TOMANDO UM-GOLPE 4
8
['HA_tomando-um-golpe', 'HA_saltando', 'HA_ataque']
['CR_atingido-pela-tampa-de-bueiro', 'CR_dr-octopus',
'CR_atingido-pela-tampa-de-bueiro-2']
['CC_policia-chegando']
```

—----------

Exemplo 3:

Entrada

```
CC_FIM DO EXPEDIENTE, HA_ENTRANDO NO TUNEL DE TREM, HA_TOMANDO
UM-GOLPE, HA_TOMANDO UM-GOLPE, HA_TOMANDO UM-GOLPE, HA_TIMES
SQUARE, HA_TUNEL DE TREM, HA_TUNEL DE TREM, CR_preso pela
policia, HA_TIMES SQUARE, CR_preso pela policia, CR_preso pela
policia, CR_preso pela policia, CR_preso pela policia, CR_preso
pela policia, HA_TIMES SQUARE, HA_ATAQUE/ CC_FIM DO EXPEDIENTE
```

Saída

```
CR_preso pela policia 5
7
['HA_entrando-no-tunel-de-trem', 'HA_tomando-um-golpe',
'HA_times-square', 'HA_tunel-de-trem', 'HA_ataque']
['CR_preso-pela-policia']
[]
```

Submissão da tarefa

Submeter no *CodePost,* na tarefa com o nome de *Lab 7 - Listas e Strings,* o arquivo:

⇒ **lab7.py**: Arquivo onde deverá ser implementada a tarefa.

Após o prazo estabelecido para a atividade, será aberta uma tarefa *Lab 7 - Segunda Chance*, com prazo de entrega até o fim do semestre.