MC102 - Algoritmos e Programação de Computadores

#ilovepython ;-)

Submissões no período de 06 a 12 de julho só serão computadas para alun*s em recuperação de Média dos Laboratórios.

Em textos disponíveis na Internet é muito comum o uso de hashtags e emoticons. Um pesquisador da área de linguística está interessado em medir o quanto o uso destes elementos é importante para a interpretação destes textos. Para isso ele irá entregar os textos originais para um grupo de leitores e os mesmos textos sem emoticons e hashtags para outro grupo. Finalmente, ele irá comparar as impressões dos grupos sobre os textos.

Sua tarefa será escrever a versão inicial de um programa em Python para auxiliar o pesquisador a filtrar os textos. Adotaremos as seguintes regras simplificadas para a classificação dos elementos:

- **Palavra:** sequência de letras (sem acento).
  Exemplos: `UNICAMP algoritmos programacao`
- **Número:** sequência de dígitos precedidos ou não do sinal negativo (`-`). Identificaremos números inteiros, não tratando números racionais, reais, complexos ou com pontos indicando a separação em grupos de três dígitos.
  Exemplos: `2020 -100000`
- **Hashtag:** caractere `#` seguido de letras.
  Exemplos: `#python #unicamp #amoviajar`
- **Emoticon:** sequência de um ou mais caracteres que não se enquadra nas descrições anteriores. Emoticons são compostos principalmente por caracteres de pontuação, mas

podem conter letras, números ou serem precedidos por `#`.
Exemplos: `:-)` `#:-)` `:D` `<3`

Descrição da entrada

Nesta versão inicial, a entrada virá pré-processada em uma linha contendo uma sequência de elementos separados por espaços em branco. Veja o exemplo abaixo:

```
#ilovepython <3 amo programar :-)
```

Descrição da saída

A saída conterá listas dos elementos classificados, respeitando a ordem da entrada. Para a entrada usada como exemplo a saída será:

```
Palavra(s): amo programar
Numero(s):
Hashtag(s): #ilovepython
Emoticon(s): <3 :-)
```

Cuidado com os espaços em branco!!!!

Nenhum espaço em branco deve ser escrito ao final das listas, caso contrário o SuSy acusará um erro na comparação entre o resultado do seu programa e a saída esperada. Observe abaixo quais são os caracteres que devem ser escritos para o exemplo utilizado anteriormente:

```
Palavra(s): amo programar
Numero(s):
Hashtag(s): #ilovepython
Emoticon(s): <3 :-)
```

Testes para o SuSy

Esta tarefa contém 7 testes abertos e 3 testes fechados. Todos os elementos a serem processados estão de acordo com as regras apresentadas na primeira seção.

arq01.in	`#UNICAMP 1966 artes ciencias cultura @>--;--`

arq01.res	`Palavra(s): artes ciencias cultura`
	`Numero(s): 1966`
	`Hashtag(s): #UNICAMP`
	`Emoticon(s): @>--;--`

arq02.in	`Python if elif else True False while for def`

arq02.res	`Palavra(s): Python if elif else True False while for def`
	`Numero(s):`
	`Hashtag(s):`
	`Emoticon(s):`

arq03.in	`-10 -5 0 5 10 15 20`

arq03.res	`Palavra(s):`
	`Numero(s): -10 -5 0 5 10 15 20`
	`Hashtag(s):`
	`Emoticon(s):`

arq04.in	`#tbt #melhordodia #ilovepython`

arq04.res	`Palavra(s):`
	`Numero(s):`
	`Hashtag(s): #tbt #melhordodia #ilovepython`
	`Emoticon(s):`

arq05.in	`:-) :) #:-) @:-) 8-) :-D =-) :-( >:-( ;-) :-/`

arq05.res	`Palavra(s):`
	`Numero(s):`
	`Hashtag(s):`
	`Emoticon(s): :-) :) #:-) @:-) 8-) :-D =-) :-( >:-( ;-) :-/`

arq06.in	`feliz :-) superfeliz :-D triste :-( bravo >:-( piscando ;-) perplexo :-/`

arq06.res	`Palavra(s): feliz superfeliz triste bravo piscando perplexo`
	`Numero(s):`
	`Hashtag(s):`
	`Emoticon(s): :-) :-D :-( >:-( ;-) :-/`

arq07.in	`#AmoChocolate cacau 70 amargo #foradieta :D`

arq07.res	`Palavra(s): cacau amargo`
	`Numero(s): 70`
	`Hashtag(s): #AmoChocolate #foradieta`
	`Emoticon(s): :D`

Releia, se necessário, as instruções para fazer os testes em Testes com o SuSy.

Dicas de Python 3 para esta tarefa:

- Para ler a linha com os elementos a serem classificados e montar uma lista de strings podemos utilizar a função `split()`:

  ```
  lista = input().split()
  ```

- Para inserir um elemento em uma lista, utilize a função `append()`:

  ```
  lista_palavras.append(palavra)
  ```

- Para operar com o primeiro elemento de uma string `s` escreva `s[0]`.
- Para operar com uma substring criada a partir da remoção do primeiro elemento da string `s` escreva `s[1:]`.
- Para verificar se uma string `s` contém apenas dígitos use a função `s.isdigit()`.

- Para verificar se uma string `s` contém apenas letras use a função `s.isalpha()`.

Orientações para submissão

Veja aqui a página de submissão da tarefa. O arquivo a ser submetido deve se chamar **`lab08.py`**. No link Arquivos auxiliares há um arquivo `aux08.zip` que contém todos os arquivos de testes abertos e seus respectivos resultados compactados.

Utilize o sistema SuSy com o mesmo login e senha que você utiliza para fazer acesso ao sistema da DAC. Se você não estiver inscrito corretamente, envie email para islene@ic.unicamp.br.

O limite máximo será de 30 submissões. Serão considerados os resultados da última submissão.

O peso desta tarefa é 3.

O prazo final para submissão é 31/05/2020.

A nota desta tarefa é proporcional ao número de testes que executaram corretamente, desde que o código esteja coerente com o enunciado. **A submissão de um código que não implementa o algoritmo requisitado, mas que exibe as saídas esperadas dos testes abertos a partir da comparação de trechos da entrada será considerada fraude e acarretará a atribuição de nota zero à média final da disciplina.**