Computação I

Lista de exercícios 6 – Tuplas, strings e interação com o usuário.

Atenção! Leia as instruções antes de fazer a lista! A padronização do nome do arquivo e dos nomes das funções é muito importante e está explicada no arquivo de instruções!

Data de entrega: 08/06/2021

A menos que esteja explicitamente pedido na questão, não utilize nenhum método ou função já existente do Python exceto pelas funções print, len, format, upper, lower e range. **A função input deve ser usada somente na questão 5.** De forma simplificada, função é tudo o que precisa ser chamado com parênteses, e método é tudo o que precisa ser chamado com parênteses, mas que está associado a uma variável, isto é, variavel.metodo(args). Funções de transformação de tipo (int, float, str) são permitidas **apenas na questão 5**. Não importe nenhum módulo. Não é necessário testar se os dados passados por argumento e se os dados passados através do input (questão 5) são válidos. Não crie funções dentro de funções. **Atenção para a diferença entre o return e o print.** O return permite que o valor seja utilizado fora da função, enquanto o print só imprime na tela. Quando eu peço para retornar um valor (independentemente do tipo), vocês precisam utilizar o return; quando eu peço para mostrar, ou imprimir, na tela, vocês precisam usar o print. Outro detalhe é que a utilização de return print('string') está errada, pois nesse caso o valor retornado não é a string e sim None (o return está retornando o que o print retorna, que é None). Isso faz diferença na correção e, principalmente, é um conceito importante.

1. Escreva uma função em Python que recebe, por argumento, uma tupla com nomes e idades (de forma intercalada e começando com um nome) e retorna duas tuplas, uma somente com os nomes e outra somente com as idades (nessa ordem).

2. Escreva uma função em Python que recebe, por argumento, uma tupla de tuplas e uma string. A tupla de tuplas indica os resultados de jogos entre times de futebol. Cada subtupla possui o nome de dois times e um número, que pode ser 0, 1 ou 2. Se o número for igual a 0, o time da posição 0 foi o vencedor, se o número for igual a 1, o time da posição 1 foi o vencedor, e se o número for igual a 2 os times empataram. A string recebida por argumento possui o nome de um time, que pode apresentar tanto caracteres em minúsculo, quanto caracteres em maiúsculo (veja o arquivo de testes para exemplos). Seu código deve ser capaz de reconhecer o time independente da forma como os caracteres foram digitados. A função deve retornar a quantidade de pontos do time passado por argumento, considerando que a quantidade de pontos é calculada por: 3*vitórias+empates. Se o time não possuir nenhuma vitória ou empate, retorne 0. Assuma que o nome do time passado como segundo argumento aparecerá em pelo menos uma subtupla presente na tupla passada como primeiro argumento.

3. Escreva uma função em Python que receba uma string como entrada e retorne uma tupla de dois elementos que representam, nesta ordem: a quantidade de caracteres diferentes de espaços em branco presentes na string de entrada, e a quantidade de palavras presentes na string. Observe que o caractere de espaço em branco é representado por " " ou ' ', e não por "" ou '' (os quais correspondem a uma string vazia). Considere que o primeiro caractere que é diferente de espaço em branco presente na string sempre corresponde a uma palavra, e que novas palavras correspondem a caracteres diferentes de espaço em branco que estão precedidos por um espaço em branco. Por exemplo: para a string de entrada ('Uma string.teste'), o programa deve retornar (15, 2), pois existem 15 caracteres diferentes de espaço em branco, e 2 palavras (que são 'Uma' e 'string.teste'). Observe que, no exemplo anterior, o trecho "string.teste" é considerado uma única palavra, porque apenas o caractere 's' do começo é precedido por um espaço em branco na string de entrada. Todos os demais caracteres subsequentes, incluindo o '.', sempre são antecedidos por um caractere que não é um espaço em branco (portanto, assume-se que eles compõem a mesma palavra). Veja o arquivo de

testes para outros exemplos de entrada. Assuma que as strings de entrada nunca terão caracteres de escape ("\n", "\r", "\t", entre outros).

4. Faça uma função em Python que receba duas strings, e retorne a quantidade de vezes que a segunda string ocorre na primeira (desconsiderando sobreposições). Por exemplo: para a entrada ('temos uma string de teste', 'te'), a saída deve ser 3, pois "te" ocorre três vezes na primeira string (1 vez em "temos" e 2 vezes em "teste"). No caso da entrada ("aaaa", "aa"), o seu código deve retornar 2, pois, desconsiderando a sobreposição (onde uma sobreposição de ocorrência é definida quando um caractere de uma dada posição é usado para compor mais de uma ocorrência), a string "aa" aparece na string "aaaa" considerando os caracteres das posições 0 e 1 e considerando os caracteres das posições 2 e 3. Veja que a ocorrência que considera os caracteres das posições 1 e 2 não deve ser contada, pois essa ocorrência existe com sobreposição. Sobreposições de ocorrências da segunda string na primeira **não** devem ser contabilizadas pela função a ser implementada. Ou seja, uma ocorrência válida deve conter apenas caracteres de posições que não foram usadas para formar outra ocorrência. Dica: utilize o operador de fatiamento para determinar se a segunda string ocorre na primeira.

5. Escreva uma função em Python que interage com um usuário, dono de uma loja, para pedir o nome de um produto (“Digite o nome do produto: ”), em seguida, a quantidade do produto em estoque (“Digite a quantidade do produto: ”) e, em seguida e por último, o preço desse produto (“Digite o preco do produto: ”). Repita a interação para cinco produtos, utilizando **obrigatoriamente** uma estrutura de repetição. A função deve imprimir, em uma única linha, a quantidade e os nomes dos produtos que possuem quantidade menor que 50, considerando o seguinte formato (assumindo que existem 3 produtos com menos de 50 unidades): Ha 3 produtos com menos de 50 unidades: nomeProduto1; nomeProduto2; nomeProduto3. Em seguida, abaixo dessa linha, a função deve imprimir o nome do produto mais caro dentre todos os produtos (ou seja, não somente aqueles que respeitam as condições acima) e seu valor com **duas casas decimais**. O formato desta mensagem deve ser o seguinte: O produto mais caro eh o arroz, e ele custa 5.00 reais (assumindo que arroz seja o produto mais caro fornecido pelo usuário, e que ele custa 5 reais). Note que essa função não deve possuir nenhum argumento de entrada. Os valores são passados por input. A função deve retornar uma tupla de tuplas com todos os produtos, onde cada subtupla contém, nessa ordem, o nome (string), a quantidade (int) e o preço (float) do produto. Atenção, os tipos indicados precisam ser respeitados: eu desejo uma tupla de tuplas, onde cada subtupla contém uma string, um inteiro e um float. Veja que, nessa função, eu peço tanto para imprimir (print) quanto para retornar (return). **Por favor, dentro da função, não imprima nada na tela além do que foi pedido e imprima as mensagens usando exatamente as mesmas palavras e caracteres dos exemplos acima!** Veja dois exemplos de execução no arquivo com os testes.