Lista de Exercícios 21 – Tuplas e Listas

Atenção: todos os problemas desta lista devem ser resolvidos com a criação de funções. Para cada um deles, crie uma função "main" que deve chamar a função que resolve o exercício.

1. Faça uma função Python chamada min_max(t), que recebe uma tupla de números "t", e retorna outra tupla contendo respectivamente o menor e o maior valor de "t".

2. Faça uma função Python chamada inverte(t) que recebe uma tupla "t" (contendo objetos de qualquer tipo) e retorna uma outra tupla com os mesmos objetos da tupla original, porém em ordem invertida. Exemplo: para uma tupla t = ("cacau", "abacaxi", "morango"), a função deve retornar ("morango", "abacaxi", "cacau")

3. Faça uma função Python chamada inverte_duplo(t) que recebe uma tupla "t" (contendo objetos do tipo string ou números inteiro) e retorna uma outra tupla com os objetos da tupla original invertidos e em ordem invertida.
   Exemplos:
   - para uma tupla t = ("cacau", "abacaxi", "morango"), a função deve retornar ("ognarom", "ixacaba", "uacac")
   - para uma tupla t =(1, 51, 27, 2), a função deve retornar (2, 72, 15, 1)

   Observação: você pode fazer sua solução implementando mais de uma função, por exemplo: uma função para inverter cada elemento da tupla, e outra para inverter a tupla. Lembre-se que a tupla é imutável, por isso provavelmente você vai ter que usar listas enquanto monta a solução do problema.

4. Um conjunto de números inteiros em Python pode se representado por uma tupla ou uma lista de números. Construa uma função chamada intersecção(c1, c2), que recebe dois conjuntos (tuplas) e retorna uma tupla representando a intersecção dos dois conjuntos.
   Observações/dicas:
   - considere que não existem valores repetidos dentro dos conjuntos
   - os valores não estão necessariamente em ordem
   - internamente a função precisará usar uma lista (porque tuplas são imutáveis)
   - talvez seja necessário fazer um laço duplo de repetição (um laço dentro de outro laço)

   Exemplo: se c1 = (1,3,2,9) e c2 = (9, 5, 1, 22) o resultado seria (1, 9) ou (9, 1) (a ordem não importa)

5. Faça uma função Python chamada par_impar(x) que recebe uma tupla **x** de 30 elementos inteiros, e retorna duas tuplas **a** e **b**. A tupla **a** deve conter os elementos pares de **x** e tupla **b**, os elementos ímpares.