Todas as funções abaixo devem ser criadas em um **módulo** e utilizadas em um **programa que testará cada uma delas**.

1. Faça uma função que, dada uma determinada lista de palavras passada como argumento, **retorne** uma nova lista contendo o seu conteúdo em ordem crescente (da menor palavra para a maior palavra – considerando a quantidade de caracteres)

2. Faça uma função que, dada uma determinada lista de palavras passada como argumento, **retorne** o seu conteúdo em ordem alfabética.

3. Faça uma função que receba duas listas como parâmetros e **retorne** uma terceira lista que tenha os conteúdos das duas de forma alternada.

   Ex.:

   Lista1 = [“a”, “b”, ”c”]

   Lista2 = [1, 2, 3, 4, 5, 6]

   Lista3 = [“a”, 1, “b”, 2, “c”, 3, 4, 5, 6]

4. Faça uma função que receba uma lista como parâmetro e **retorne** somente os valores das posições pares desta lista

5. Faça uma função que receba uma lista como parâmetro e **retorne** os seus valores em ordem alfabética. A lista não poderá ser modificada durante a execução da função.

6. Faça uma função que receba como parâmetro uma lista com 10 valores reais informados pelo usuário. A função deverá colocar todos estes valores em duas listas: uma somente de números pares e uma somente com os números ímpares. Estas listas deverão ser impressas pela função.

7. Considere uma lista composta por listas de números inteiros. Faça uma função que receba uma lista deste tipo e que imprima o maior número existente em cada uma das listas internas e o maior número no geral.

8. Considere uma lista composta por listas de números reais. Faça uma função que receba uma lista deste tipo e calcule a média dos valores de cada lista interna, inserindo-os ao fim de cada uma delas. No fim da lista externa deve ser inserida a média geral dos valores de todas as listas. A função deverá **retornar** esta lista.

9. Faça uma função que receba uma lista de números inteiros como parâmetro e retorne uma nova lista com todos os elementos originais, exceto os primeiros e os últimos.
   Ex.:
   lista = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
   novaLista = [5, 6]

10. Faça uma função que recebe uma lista como parâmetro e remova o primeiro e o último elemento da lista.
    Ex.:
    lista = [1, 2, 3, 4, 5]
    Lista depois da função: [2, 3, 4]

11. Faça uma função que receba uma lista como parâmetro e retorne **True** caso a lista esteja ordenada de forma crescente e **False** caso contrário.

12. Faça uma função que receba uma lista como parâmetro e retorne **True** caso haja algum elemento que apareça mais de uma vez e **False** caso contrário. Ela não deve modificar a lista original.

13. Crie uma função que receba como parâmetro um valor inteiro e retorne uma lista de inteiros preenchida de forma aleatória de acordo com a quantidade especificada no parâmetro.
    Dica: para gerar valores aleatórios pesquise pelo módulo random do Python no Google.
    Ex.:
    Parâmetro: 5
    Lista retornada = [3, 2, 5, 4, 1]

14. Para verificar se um valor está contido em uma lista de inteiros, você pode usar o operador **in**, mas isso seria lento porque pesquisaria ao elementos em ordem.

    Caso a lista esteja ordenada de forma crescente, podemos acelerar as coisas com uma busca por bisseção (também conhecida como **busca binária**), que é semelhante ao que você faz quando procura uma palavra no dicionário. Você começa no meio e verifica se a palavra que está procurando vem antes da palavra no meio da lista. Se for o caso, procura na primeira metade da lista. Se não, procura na segunda metade.

    De qualquer forma, você corta o espaço de busca restante pela metade a cada execução.

    Escreva uma função que receba como parâmetro uma lista ordenada de inteiros, um valor-alvo e devolva o índice do valor na lista se ele estiver lá, ou **None** se não estiver.

    Escreve uma segunda função que terá o mesmo objetivo que a primeira (encontrar o índice do valor na lista), porém esta função deverá implementar a **busca binária** para encontrar o resultado desejado.

    Para testar as funções utilize a função criada no exercício 13 e gere listas com grandes quantidades de valores (1 milhão, 2 milhões, etc). Ao final, você deverá medir o tempo de execução de ambas as funções e apresentar o resultado na tela.

Dica: para o medir o tempo de execução de uma função pesquise pelo módulo time do Python no Google.
Vídeo com a explicação sobre buscas binárias:
https://www.youtube.com/watch?v=5T1SDEZzCLo