Funções do Básico ao Avançado

1. **Calculadora Simples:**
   Crie uma calculadora simples que recebe dois números e realiza operações de soma, subtração, multiplicação e divisão.

2. **Um print especial**
   Faça um programa que tenha uma função chamada escreva(), que receba um texto qualquer como parâmetro e mostre uma mensagem com tamanho adaptável.
   Ex:
   escreva('Olá, Mundo!')
   Saída:
   ~~~~~~~~~~
   Olá, Mundo!
   ~~~~~~~~~~

3. **Função de Contador**
   Faça um programa que tenha uma função chamada contador(), que receba três parâmetros: início, fim e passo. Seu programa tem que realizar três contagens através da função criada:
   a) de 1 até 10, de 1 em 1
   b) de 10 até 0, de 2 em 2
   c) uma contagem personalizada

4. **Funções para sortear e somar**
   Faça um programa que tenha uma lista chamada números e duas funções chamadas sorteia() e somaPar(). A primeira função vai sortear 5 números e vai colocá-los dentro da lista e a segunda função vai mostrar a soma entre todos os valores pares sorteados pela função anterior.

5. **Função que descobre o maior**
   Faça um programa que tenha uma função chamada maior(), que receba vários parâmetros com valores inteiros. Seu programa tem que analisar todos os valores e dizer qual deles é o maior.

6. **Função que calcula área**
   Faça um programa que tenha uma função chamada área(), que receba as dimensões de um terreno retangular (largura e comprimento) e mostre a área do terreno.

7. **Verificador de Números Pares:**
   Escreva uma função que recebe um número e verifica se é par. A função deve retornar **True** se o número for par e **False** caso contrário.

8. **Contador de Palavras:**
   Crie uma função que recebe uma frase como entrada e conta o número de palavras na frase.

9. **Maior entre Três Números:**
   Escreva uma função que recebe três números e retorna o maior entre eles.

10. **Tabuada:**
    Crie um programa que gera a tabuada de um número fornecido pelo usuário.

11. **Fatorial:**
    Implemente uma função para calcular o fatorial de um número.

12. **Adivinhe o Número:**
    Desenvolva um jogo onde o computador escolhe um número aleatório e o jogador tem que adivinhar qual é. O programa deve fornecer dicas se o palpite do jogador está alto ou baixo.

13. **Verificador de Palíndromos:**
Escreva uma função que verifica se uma palavra é um palíndromo (lê-se igual de trás para frente).

14. **Conversor de Temperatura:**
Implemente uma função que converte uma temperatura de Celsius para Fahrenheit ou vice-versa, dependendo da escolha do usuário.

15. **Jogo da Forca:**
Desenvolva um jogo da forca onde o computador escolhe uma palavra aleatória e o jogador tenta adivinhar a palavra.

16. **Contador de Frequência de Palavras:**
Escreva um programa que conta a frequência de cada palavra em um texto fornecido pelo usuário.

17. **Validador de Senhas Fortes:**
Crie uma função que verifica se uma senha atende aos critérios de uma senha forte, incluindo comprimento mínimo, uso de letras maiúsculas, minúsculas, números e caracteres especiais.

18. **Ordenação por Seleção:**
Implemente um algoritmo de ordenação por seleção para ordenar uma lista de números.

19. **Jogo da Velha:**
Crie uma versão simples do jogo da velha para dois jogadores.

20. **Fibonacci com Memoização:**
Escreva uma função que calcula os termos da sequência de Fibonacci usando memoização para melhorar o desempenho.

21. **Analisador de Números Primos:**
Desenvolva um programa que verifica se um número fornecido pelo usuário é primo.

22. **Busca Binária:**
Implemente um algoritmo de busca binária para encontrar um elemento em uma lista ordenada.

23. **Simulador de Dados:**
Crie um programa que simula o lançamento de dados e calcula a frequência de cada resultado.

24. **Funções para votação**
Crie um programa que tenha uma função chamada voto() que vai receber como parâmetro o ano de nascimento de uma pessoa, retornando um valor literal indicando se uma pessoa tem voto NEGADO, OPCIONAL e OBRIGATÓRIO nas eleições.

25. **Função para Fatorial**
Crie um programa que tenha uma função fatorial() que receba dois parâmetros: o primeiro que indique o número a calcular e outro chamado show, que será um valor lógico (opcional) indicando se será mostrado ou não na tela o processo de cálculo do fatorial.

26. **Ficha do Jogador**
Faça um programa que tenha uma função chamada ficha(), que receba dois parâmetros opcionais: o nome de um jogador e quantos gols ele marcou. O programa deverá ser capaz de mostrar a ficha do jogador, mesmo que algum dado não tenha sido informado corretamente.

27. **Validando entrada de dados em Python**
Crie um programa que tenha a função leiaInt(), que vai funcionar de forma semelhante ‘a função input() do Python, só que fazendo a validação para aceitar apenas um valor numérico.
Ex: n = leiaInt(‘Digite um n: ‘)

28. **Analisando e gerando Dicionários**

Faça um programa que tenha uma função notas() que pode receber várias notas de alunos e vai retornar um dicionário com as seguintes informações:

– Quantidade de notas
– A maior nota
– A menor nota
– A média da turma
– A situação (opcional)

29. **Interactive helping system in Python**
Faça um mini-sistema que utilize o Interactive Help do Python. O usuário vai digitar o comando e o manual vai aparecer. Quando o usuário digitar a palavra 'FIM', o programa se encerrará. Importante: use cores.

30. **Validador de Expressões Matemáticas:**
Escreva uma função que valida se uma expressão matemática fornecida pelo usuário é válida, considerando parênteses, colchetes e chaves.

*25 exercícios em Python de nível intermediário, abordando funções com parâmetros, funções sem parâmetros, estruturas de repetição, estruturas condicionais, list comprehension, ***args**, ****kwargs** e outros conceitos avançados:*

31. **Soma de Elementos:**
Crie uma função que recebe uma lista de números como parâmetro e retorna a soma de todos os elementos dessa lista.

32. **Média de Números:**
Escreva uma função que calcula a média de uma lista de números fornecida como parâmetro.

33. **Inversão de String:**
Implemente uma função que recebe uma string como entrada e retorna a string invertida.

34. **Quadrados com List Comprehension:**
Desenvolva um programa que utiliza list comprehension para criar uma lista dos quadrados dos números de 1 a 10.

35. **Comprimento das Palavras:**
Crie uma função que recebe uma lista de palavras como parâmetro e retorna uma lista com o comprimento de cada palavra.

36. **Verificador de Palíndromos:**
Escreva uma função que verifica se uma palavra é um palíndromo, levando em conta maiúsculas e minúsculas.

37. **Média com args:**
Implemente uma função que utiliza *args para calcular a média de uma quantidade variável de números.

38. **Dicionário de Comprimentos com kwargs:**
Desenvolva uma função que utiliza **kwargs para criar um dicionário com pares chave-valor, onde as chaves são strings e os valores são seus comprimentos.

39. **Lista de Números Pares:**

Crie uma função que recebe uma lista de números como parâmetro e retorna uma lista apenas com os números pares.

40. **Fatorial Recursivo:**
Escreva uma função que calcula o fatorial de um número utilizando recursão.

41. **Simulação de Jogo de Dados:**
Implemente um programa que simula um jogo de dados, onde os jogadores podem apostar em números específicos.

42. **Ordenação de Strings por Comprimento:**
Desenvolva uma função que recebe uma lista de strings e retorna uma nova lista com as strings ordenadas por comprimento.

43. **Verificador de Números Primos:**
Crie uma função que recebe um número como parâmetro e retorna True se ele for primo e False caso contrário.

44. **List Comprehension para Números Primos:**
Escreva um programa que utiliza list comprehension para criar uma lista de todos os números primos até 100.

45. **Contagem de Vogais:**
Implemente uma função que recebe uma frase como parâmetro e conta o número de ocorrências de cada vogal.

46. **Imprimir *args e kwargs:**
Desenvolva um programa que utiliza *args e **kwargs para imprimir os argumentos passados, tanto posicionais quanto nomeados.

47. **Elementos Comuns entre Listas:**
Crie uma função que recebe duas listas como parâmetro e retorna uma lista com os elementos comuns entre as duas listas.

48. **Lista de Números Ímpares:**
Escreva uma função que recebe uma lista de números como parâmetro e retorna uma lista apenas com os números ímpares.

49. **Simulação de Caixa Eletrônico:**
Implemente um programa que simula um caixa eletrônico, permitindo saques, depósitos e consulta de saldo

50. **Palavras que Começam com uma Letra:**
Desenvolva uma função que recebe uma lista de palavras e uma letra como parâmetro e retorna uma lista apenas com as palavras que começam com essa letra.

51. **Soma de Números Maiores que 10:**
Crie uma função que recebe uma lista de números como parâmetro e retorna a soma dos números maiores que 10.

52. **Tuplas com List Comprehension:**
Escreva um programa que utiliza list comprehension para criar uma lista de tuplas, onde cada tupla contém um número e seu quadrado correspondente.

53. **Verificador de Letras Maiúsculas:**

Implemente uma função que recebe uma string como parâmetro e retorna True se ela contiver apenas letras maiúsculas e False caso contrário.

54. **Jogo da Velha:**
Desenvolva um programa que simula um jogo da velha para dois jogadores.

55. **União de Elementos Únicos entre Listas:**
Crie uma função que recebe duas listas como parâmetro e retorna uma lista com os elementos únicos de ambas as listas, sem repetições.

*25 exercícios em Python de nível avançado, abrangendo funções com parâmetros, funções sem parâmetros, estruturas de repetição, estruturas condicionais, list comprehension, *args, **kwargs e conceitos mais avançados:*

56. **Algoritmo de Ordenação Avançado:**
Implemente uma função que utiliza um algoritmo de ordenação eficiente, como QuickSort ou MergeSort, para ordenar uma lista de números.

57. **Threads para Tarefas Concorrentes:**
Escreva um programa que utiliza threads para realizar tarefas concorrentes, como a busca paralela em uma lista grande.

58. **Criptografia e Descriptografia com AES:**
Desenvolva uma função que implementa a criptografia e descriptografia de mensagens utilizando o algoritmo de criptografia simétrica AES.

59. **Operações Assíncronas com asyncio:**
Crie um programa que utiliza asyncio para realizar operações assíncronas, como chamadas de API simultâneas, melhorando a eficiência do código.

60. **Análise de Sentimentos com NLP:**
Implemente uma função que utiliza uma biblioteca de processamento de linguagem natural (NLP) para realizar a análise de sentimentos em um texto.

61. **Web Scraping com BeautifulSoup ou Scrapy:**
Escreva um script que faz o scraping de dados de uma página web utilizando BeautifulSoup ou Scrapy.

62. **Autenticação Segura com JWT:**
Implemente um sistema de autenticação seguro utilizando tokens JWT (JSON Web Tokens) para garantir a segurança da comunicação entre cliente e servidor.

63. **Classificação com Machine Learning:**
Desenvolva uma função que utiliza machine learning para realizar a classificação de dados, utilizando uma biblioteca como Scikit-Learn.

64. **API RESTful com Flask ou Django:**
Crie um programa que utiliza Flask ou Django para construir uma API RESTful, proporcionando interações eficientes entre diferentes partes de uma aplicação.

65. **Agrupamento com K-means:**
Elabore uma função que utiliza o algoritmo K-means para realizar agrupamento de dados, identificando padrões complexos.

66. **Processamento de Imagens com OpenCV:**
Escreva um script que utiliza a biblioteca OpenCV para processar imagens, aplicando filtros e reconhecendo objetos.

67. **Sistema de Recomendação com Filtragem Colaborativa:**
Implemente um sistema de recomendação de itens utilizando algoritmos de filtragem colaborativa, explorando preferências de usuários similares.

68. **Análise de Séries Temporais:**
Desenvolva uma função que realiza a análise de séries temporais, utilizando bibliotecas como Pandas e NumPy para identificar tendências e padrões temporais.

69. **Visualizações de Dados com Matplotlib:**
Crie um programa que utiliza a biblioteca Matplotlib para criar visualizações complexas de dados, facilitando a interpretação de informações.

70. **Classificação com Árvore de Decisão:**
Elabore uma função que utiliza o algoritmo de árvore de decisão para realizar a classificação de dados, explorando diferentes ramos de decisão.

71. **Automação com Selenium:**
Escreva um script que utiliza o Selenium para automatizar interações com páginas web, tornando o processo de teste e extração de dados mais eficiente.

72. **Treinamento de Modelo com TensorFlow ou PyTorch:**
Implemente uma função que realiza o treinamento de um modelo de aprendizado profundo utilizando frameworks como TensorFlow ou PyTorch.

73. **Processamento de Linguagem Natural Avançado:**
Desenvolva um sistema de processamento de linguagem natural que inclua análise de entidades, sentimentos e compreensão contextual.

74. **API Assíncrona com FastAPI:**
Crie uma aplicação que utiliza o framework FastAPI para construir uma API assíncrona, aproveitando a eficiência do código assíncrono.

75. **Análise de Redes Sociais:**
Elabore uma função que realiza a análise de redes sociais, identificando comunidades, influenciadores e padrões de conexão.

76. **Otimização Genética:**
Escreva um programa que utiliza o algoritmo de otimização genética para resolver um problema complexo, como otimização de parâmetros.

77. **Recomendação de Filmes com Aprendizado Profundo:**
Implemente um sistema de recomendação de filmes baseado em aprendizado profundo, explorando arquiteturas de redes neurais.

78. **Análise de Importância de Páginas com PageRank:**
Desenvolva uma função que utiliza o algoritmo de PageRank para analisar a importância de páginas da web em uma rede.

79. **Processamento Avançado de Linguagem Natural com NLTK:**
Crie um script que utiliza a biblioteca Natural Language Toolkit (NLTK) para realizar processamento avançado de linguagem natural.

80. **nálise Estatística Avançada com Statsmodels:**
Elabore uma função que utiliza a biblioteca Statsmodels para realizar análise estatística avançada, explorando modelos estatísticos complexos.

81. **Processamento de Áudio com Librosa:**
Escreva um programa que utiliza a biblioteca Librosa para realizar análise e processamento de áudio, como extração de características musicais.

82. **Reconhecimento de Objetos com TensorFlow e OpenCV:**
Desenvolva um script que utiliza TensorFlow e OpenCV para realizar o reconhecimento de objetos em imagens ou vídeos.

83. **Implementação de um Chatbot:**
Crie um sistema de chatbot utilizando uma biblioteca como Rasa ou ChatterBot, capaz de responder a perguntas e manter diálogos contextuais.

84. **Análise de Grafos com NetworkX:**
Elabore uma função que utiliza a biblioteca NetworkX para realizar análise de grafos, identificando centralidade, caminhos mínimos, entre outros.

85. **Sistema de Recomendação Baseado em Conteúdo:**
Implemente um sistema de recomendação baseado em conteúdo que sugira itens similares com base nas características dos itens e nas preferências do usuário.

86. **Integração de API de Machine Learning:**
Crie um programa que faz a integração com uma API de machine learning, enviando dados para previsões e processando os resultados.

87. **Processamento de Linguagem Natural com SpaCy:**
Desenvolva um script que utiliza a biblioteca SpaCy para realizar tarefas avançadas de processamento de linguagem natural, como análise sintática e named entity recognition (NER).

88. **Implementação de um Framework Web:**
Crie um pequeno framework web, inspirado em frameworks populares como Flask ou Django, para entender os conceitos fundamentais por trás dessas ferramentas.

89. **Animação Gráfica com Pygame:**
Escreva um programa que utiliza a biblioteca Pygame para criar animações gráficas interativas.

90. **Aprendizado Não Supervisionado com K-means:**
Desenvolva uma função que utiliza o algoritmo K-means para realizar aprendizado não supervisionado em dados complexos.

91. **Reconhecimento Facial com OpenCV e Dlib:**
Implemente um sistema de reconhecimento facial utilizando as bibliotecas OpenCV e Dlib.

92. **Sistema de Recomendação de Música com Spotipy:**
Crie um programa que utiliza a biblioteca Spotipy para construir um sistema de recomendação de músicas baseado no histórico de reprodução do usuário.

93. **Otimização de Código com Numba:**
Refatore uma função existente para aproveitar a otimização JIT (Just-In-Time) da biblioteca Numba.

94. **Análise de Sentimento Multilíngue com Transformers:**
Desenvolva uma função que utiliza modelos de transformers, como BERT ou GPT, para realizar análise de sentimento em textos multilíngues.

95. **Integração de Banco de Dados NoSQL:**
Crie um programa que faz a integração com um banco de dados NoSQL, como MongoDB ou Cassandra, para armazenar e recuperar dados.