Listas, Tuplas, Dicionários, Conjuntos, Collections
Utilizando, Condicionais e Laços de Repetição For e While

TUPLAS

1- **Número por Extenso:**
Crie um programa que tenha uma dupla totalmente preenchida com uma contagem por extenso, de zero até vinte. Seu programa deverá ler um número pelo teclado (entre 0 e 20) e mostrá-lo por extenso.

2- **Times de Futebol**
Crie uma tupla preenchida com os 20 primeiros colocados da Tabela do Campeonato Brasileiro de Futebol, na ordem de colocação. Depois mostre:

a) Os 5 primeiros times.
b) Os últimos 4 colocados.
c) Times em ordem alfabética.
d) Em que posição está o time da Chapecoense.

3- **Maior e menor valores em Tupla**
Crie um programa que vai gerar cinco números aleatórios e colocar em uma tupla. Depois disso, mostre a listagem de números gerados e também indique o menor e o maior valor que estão na tupla.

4- **Análise de dados em uma Tupla**

Desenvolva um programa que leia quatro valores pelo teclado e guarde-os em uma tupla. No final, mostre:
A) Quantas vezes apareceu o valor 9.
B) Em que posição foi digitado o primeiro valor 3.
C) Quais foram os números pares.

5- **Lista de Preços com Tupla**
Crie um programa que tenha uma tupla única com nomes de produtos e seus respectivos preços, na sequência. No final, mostre uma listagem de preços, organizando os dados em forma tabular.

6- **Contando vogais em Tupla**
Crie um programa que tenha uma tupla com várias palavras (não usar acentos). Depois disso, você deve mostrar, para cada palavra, quais são as suas vogais.

7- **Filtragem de Tuplas:**
Crie uma tupla de números. Utilize uma estrutura de repetição para criar uma nova tupla contendo apenas os números pares da tupla original.

8- **Múltiplos de Três:**
Gere uma tupla de números aleatórios.
Use uma estrutura de repetição para criar uma nova tupla contendo apenas os números múltiplos de 3.

9- **Soma Cumulativa:**
Crie uma tupla de números.
Implemente um loop **for** para criar uma nova tupla onde cada elemento seja a soma cumulativa dos elementos anteriores.

10- **Fatorial com While:**
Solicite ao usuário um número. Utilize um loop **while** para calcular o fatorial desse número.

11- **Ordenação de Tuplas:**
Crie uma tupla de palavras. Ordene a tupla em ordem alfabética usando uma estrutura condicional.

12- **Sequência de Fibonacci:**
Peça ao usuário para fornecer a quantidade de termos desejados.
Utilize um loop **while** para gerar a sequência de Fibonacci até o termo desejado.

13- **Validação de Senha:**
Peça ao usuário para inserir uma senha.
Utilize uma estrutura condicional para verificar se a senha atende a critérios específicos.

14- **Contagem de Elementos:**
Crie uma tupla mista de dados. Use um loop for para contar quantas vezes cada tipo de dado aparece na tupla.

15- **Média com Ponderação:**
Gere duas tuplas, uma com notas e outra com pesos. Calcule a média ponderada das notas.

16- **Busca em Tuplas:**
Crie uma tupla de strings. Implemente um loop while que permite ao usuário buscar por uma string na tupla.

17- **Subconjuntos:**
Gere uma tupla com elementos. Utilize um loop for para gerar todos os subconjuntos possíveis.

18- **Soma de Quadrados:**
Crie uma tupla de números.
Utilize um loop for para calcular a soma dos quadrados dos elementos.

19- **Validação de Email:**
Peça ao usuário para inserir um e-mail.
Utilize uma estrutura condicional para validar se o e-mail possui um formato correto.

20- **Números Primos:**
Implemente um loop while para gerar os primeiros 10 números primos.

21- **Maior Elemento:**
Crie uma tupla de números.
Utilize um loop for para encontrar o maior elemento na tupla.

22- **Troca de Valores:**
Crie duas tuplas. Utilize uma estrutura condicional para trocar os valores entre as duas tuplas.

23- **Contagem de Palavras:**
Peça ao usuário para inserir uma frase. Utilize um loop for para contar quantas palavras a frase possui.

24- **Média com Exclusão:**
Gere uma tupla de notas. Permita ao usuário escolher uma nota para ser excluída e, em seguida, calcule a média das notas restantes.

25- **Conversão de Temperaturas:**
Crie uma tupla com temperaturas em graus Celsius. Use um loop for para converter cada temperatura para Fahrenheit e armazenar em uma nova tupla.

LISTAS

1- **Faça uma lista de comprar com listas**. O usuário deve ter a possibilidade de inserir, apagar e listar valores da sua lista. Não permita que o programa quebre com erros de índices inexistentes na lista.

2- **Calculo do primeiro dígito do CPF**
CPF: 746.824.890-70
Colete a soma dos 9 primeiros dígitos do CPF multiplicando cada um dos valores por uma contagem regressiva começando de 10

Ex.: 746.824.890-70 (746824890)
10 9 8 7 6 5 4 3 2
* 7 4 6 8 2 4 8 9 0
70 36 48 56 12 20 32 27 0

Somar todos os resultados:
70+36+48+56+12+20+32+27+0 = 301
Multiplicar o resultado anterior por 10: 301 * 10 = 3010
Obter o resto da divisão da conta anterior por 11
3010 % 11 = 7
Se o resultado anterior for maior que 9: resultado é 0
contrário disso: resultado é o valor da conta
O primeiro dígito do CPF é 7

3- **Maior e Menor valores na Lista**
Faça um programa que leia 5 valores numéricos e guarde-os em uma lista.
No final, mostre qual foi o maior e o menor valor digitado e as suas respectivas posições na lista.

4- **Valores únicos em uma Lista**
Crie um programa onde o usuário possa digitar vários valores numéricos e cadastre-os em uma lista. Caso o número já exista lá dentro, ele não será adicionado. No final, serão exibidos todos os valores únicos digitados, em ordem crescente.

5- **Lista ordenada sem repetições**
Crie um programa onde o usuário possa digitar cinco valores numéricos e cadastre-os em uma lista, já na posição correta de inserção (sem usar o sort()). No final, mostre a lista ordenada na tela.

6- **Extraindo dados de uma Lista**
Crie um programa que vai ler vários números e colocar em uma lista. Depois disso, mostre:
A) Quantos números foram digitados.
B) A lista de valores, ordenada de forma decrescente.
C) Se o valor 5 foi digitado e está ou não na lista.

7- **Dividindo valores em várias listas**
Crie um programa que vai ler vários números e colocar em uma lista. Depois disso, crie duas listas extras que vão conter apenas os valores pares e os valores ímpares digitados, respectivamente. Ao final, mostre o conteúdo das três listas geradas.

8- **Validando expressões matemáticas**
Crie um programa onde o usuário digite uma expressão qualquer que use parênteses. Seu aplicativo deverá analisar se a expressão passada está com os parênteses abertos e fechados na ordem correta.

9- **Números Pares em uma Lista:**
Crie uma lista de números.
Use uma estrutura de repetição para percorrer a lista.
Dentro da estrutura de repetição, use uma condicional para verificar se o número é par.

10- **Verificar Nome na Lista:**
Crie uma lista de nomes.
Solicite um nome do usuário.
Use uma condicional para verificar se o nome fornecido está na lista.

11- **Média de uma Lista de Números:**
Crie uma lista de números.
Use uma estrutura de repetição para somar todos os números.
Divida a soma pelo número de elementos na lista para obter a média.

12- **Manipulação de Matrizes:** Crie uma função que receba duas matrizes como entrada e retorne a multiplicação dessas matrizes, utilizando listas para representar as matrizes.

13- **Filtragem de Números Primos**: Escreva um programa que gere uma lista de números primos no intervalo de 1 a 1000, utilizando estruturas de repetição e condicionais.

14- **Análise de Sequência Numérica:** Desenvolva um código que avalie uma lista de números e identifique se há uma sequência crescente ou decrescente, considerando elementos consecutivos.

15- **Busca e Substituição em Listas:** Crie um programa que busque um elemento específico em uma lista e substitua todas as suas ocorrências por um valor determinado.

16- **Ordenação Personalizada:** Implemente um algoritmo de ordenação que organize uma lista de strings com base no comprimento das palavras, do maior para o menor.

17- **Geração de Sequência de Fibonacci:** Escreva um código que gere os primeiros 100 termos da sequência de Fibonacci, armazenando-os em uma lista.

18- **Avaliação de Desempenho:** Crie uma função que avalie o desempenho de outra função, medindo o tempo de execução para diferentes tamanhos de entrada e plotando um gráfico.

19- **Análise de Palíndromos:** Implemente um programa que verifique se uma lista de palavras é um palíndromo ou não, considerando a ordem das letras.

20- **Manipulação de Strings Avançada:** Desenvolva um código que inverta a ordem das palavras em uma frase, preservando a ordem das letras dentro de cada palavra.

21- **Validação de Sudoku:** Crie uma função que valide se uma matriz 9x9 representa uma solução válida para um jogo de Sudoku.

22- **Compressão de Dados:** Implemente um algoritmo de compressão de dados que reduza o tamanho de uma lista de números repetidos.

23- **Contagem de Palavras em Arquivos:** Escreva um programa que leia um arquivo de texto e conte a ocorrência de cada palavra, apresentando os resultados em ordem alfabética.

24- **Filtragem de Dados:** Crie um código que filtre uma lista de dicionários, selecionando apenas os elementos que atendam a critérios específicos.

25- **Simulação de Jogo de Tabuleiro:** Desenvolva um programa que simule o funcionamento de um jogo de tabuleiro, utilizando listas para representar o estado do tabuleiro e suas peças.

26- **Conversão de Bases Numéricas:** Implemente uma função que converta um número decimal para binário, octal e hexadecimal, armazenando os resultados em uma lista.

27- **Avaliação de Expressões Matemáticas:** Crie um código que avalie expressões matemáticas representadas como strings, considerando a precedência de operadores.

28- **Filtro de Dados Temporais:** Desenvolva um programa que filtre uma lista de eventos por data, selecionando apenas aqueles que ocorrem em um intervalo específico.

29- **Análise de Grafos:** Implemente um algoritmo que realize a busca em largura em um grafo representado por uma lista de adjacência.

30- **Reconhecimento de Padrões em Texto**: Escreva um programa que identifique a ocorrência de padrões específicos em um texto, utilizando expressões regulares.

31- **Simulação de Redes Sociais:** Crie um código que simule a adição de amigos em uma rede social, mantendo uma lista de conexões entre usuários.

DICIONARIOS

1- **Contagem de Palavras Únicas:** Crie um programa que conta o número de ocorrências de cada palavra única em um texto utilizando um dicionário.

2- **Fusão de Dicionários:** Desenvolva um código que combine dois dicionários, somando os valores para chaves comuns e adicionando as chaves exclusivas de cada dicionário.

3- **Média Ponderada de Notas:** Solicite ao usuário que insira notas e pesos correspondentes. Calcule a média ponderada das notas utilizando dicionários para armazenar as informações.

4- **Verificação de Anagramas:** Implemente um programa que verifica se duas palavras são anagramas, ou seja, se possuem as mesmas letras, mas em ordens diferentes.

5- **Fatorial com Memória:** Crie uma função que calcula o fatorial de um número, armazenando os resultados já calculados em um dicionário para evitar recálculos.

6- **Controle de Estoque:** Desenvolva um sistema simples de controle de estoque utilizando um dicionário para armazenar os itens e suas quantidades disponíveis.

7- **Busca por Chave Parcial:** Implemente um mecanismo de busca que permite encontrar chaves em um dicionário a partir de uma sequência de caracteres parciais.

8- **Validação de Senha:** Crie uma função que valida senhas de acordo com critérios específicos e utiliza um dicionário para armazenar mensagens de erro correspondentes a cada critério.

9- **Simulação de Jogo de Cartas:** Desenvolva um jogo de cartas simples utilizando dicionários para representar as cartas, os jogadores e as regras.

10- **Conversor de Moedas:** Crie um programa que realiza a conversão de valores monetários entre diferentes moedas, utilizando um dicionário para armazenar as taxas de câmbio.

11- **Avaliação de Expressões Matemáticas:** Implemente um avaliador de expressões matemáticas simples, utilizando dicionários para representar as operações e as variáveis.

12- **Ordenação Personalizada:** Desenvolva um algoritmo de ordenação personalizada para ordenar um dicionário com base em valores específicos.

13- **Validação de CPF:** Crie uma função que valida se um número de CPF é válido, utilizando dicionários para armazenar informações sobre os dígitos verificadores.

14- **Jogo da Forca:** Implemente o jogo da forca em que o programa escolhe uma palavra aleatória e o jogador tenta adivinhar, utilizando dicionários para rastrear as letras já escolhidas e o estado atual da palavra.

15- **Calculadora de Matrizes**: Crie uma calculadora que realiza operações de soma, subtração e multiplicação de matrizes, utilizando dicionários para representar as matrizes.

16- **Sistema de Recomendação Simples:** Desenvolva um sistema de recomendação que sugere itens com base nas preferências do usuário, armazenadas em um dicionário

17- **Simulação de Redes Sociais:** Crie uma simulação simplificada de uma rede social, utilizando dicionários para representar os perfis dos usuários e suas conexões.

18- **Adivinhação de Números:** Implemente um jogo em que o programa escolhe um número aleatório e o jogador tenta adivinhar, utilizando dicionários para rastrear as tentativas e fornecer dicas.

19- **Agenda Telefônica Avançada**: Desenvolva uma agenda telefônica que permite adicionar, remover e buscar contatos, utilizando um dicionário para armazenar as informações.

20- **Simulação de Filas em um Banco:** Crie um programa que simula o atendimento em um banco, utilizando dicionários para representar os clientes na fila e suas transações.

21- **Criptografia Simples:** Desenvolva um programa que criptografa e descriptografa mensagens usando um dicionário de substituição de caracteres.

22- **Jogo de Estratégia:** Crie um jogo de estratégia em que os jogadores possuem exércitos representados por dicionários, e as batalhas são decididas com base nas forças dos exércitos.

23- **Avaliação de Expressões Lógicas:**
Implemente uma função que avalia expressões lógicas contendo operadores AND, OR e NOT utilizando dicionários para mapear variáveis booleanas.

24- **Análise de Texto:**
Desenvolva um programa que analisa a frequência de cada palavra em um texto, utilizando um dicionário para armazenar as contagens.

25- **Simulação de Vendas:**
Crie um sistema de simulação de vendas em que os produtos, clientes e transações são representados por dicionários.

26- **Validação de E-mails:**
Implemente uma função que valida endereços de e-mail com base em regras específicas, utilizando dicionários para armazenar mensagens de erro.

27- **Simulação de Tráfego:**
Desenvolva um programa que simula o tráfego em uma interseção, utilizando dicionários para representar os semáforos e os carros.

28- **Análise de Dados Meteorológicos:**
Crie um programa que analisa dados meteorológicos, armazenados em dicionários, e fornece informações como média, máximo e mínimo de temperatura.

29- **Sistema de Reservas:**
Implemente um sistema de reservas de recursos, como salas de reunião, utilizando dicionários para rastrear a disponibilidade.

30- **Simulação de População:**
Desenvolva um programa que simula o crescimento de uma população ao longo do tempo, utilizando um dicionário para representar as características de cada indivíduo.

31- **Jogo de Palavras Cruzadas:**
Crie um jogo de palavras cruzadas em que os jogadores preenchem as palavras em um tabuleiro representado por um dicionário.

32- **Sistema de Pontuação em Jogos:**
Implemente um sistema de pontuação em jogos, onde os jogadores acumulam pontos com base em suas ações, utilizando um dicionário para rastrear as pontuações.

33- **Simulação de Bolsa de Valores:**
Desenvolva um programa que simula a variação de preços de ações ao longo do tempo, utilizando um dicionário para representar os valores das ações.

34- **Validação de Cartões de Crédito:**
Crie uma função que valida números de cartões de crédito utilizando o algoritmo de Luhn, com um dicionário para mapear mensagens de erro.

35- **Sistema de Reservas de Passagens:**
Implemente um sistema de reservas de passagens aéreas, utilizando dicionários para representar os voos, assentos e disponibilidade.

36- **Simulação de Ecossistema:**
Desenvolva um programa que simula um ecossistema com diferentes espécies interagindo, utilizando dicionários para representar os organismos.

37- **Controle de Acesso a Recursos:**
Crie um sistema que controla o acesso a recursos com base em permissões, utilizando dicionários para mapear usuários e suas autorizações.

38- **Simulação de Corrida de Carros:**
Implemente uma simulação de corrida de carros em que cada carro é representado por um dicionário e avança com base em condições específicas.

39- **Análise de Redes Sociais:**
Desenvolva um programa que analisa redes sociais, identificando conexões e influenciadores, utilizando dicionários para representar os perfis dos usuários.

40- **Sistema de Recomendação de Produtos:**
Crie um sistema de recomendação de produtos com base nas preferências dos usuários, utilizando dicionários para representar as avaliações.

CONJUNTOS

1- **União de conjuntos**
Dado dois conjuntos A e B, crie um programa que realize a união desses conjuntos.

2- **Interseção de conjuntos**
Desenvolva um programa que determine a interseção entre dois conjuntos A e B.

3- **Diferença entre conjuntos**
Elabore um programa que calcule a diferença entre dois conjuntos A e B.

4- **Verificação de subconjunto**
Crie um programa que verifique se um conjunto A é subconjunto de outro conjunto B.

5- **Combinações de conjuntos**
Desenvolva um programa que gere todas as combinações possíveis de elementos de dois conjuntos A e B.

6- **Adição condicional de elementos em um conjunto**
Implemente um programa que adicione elementos de um conjunto A em um conjunto B apenas se esses elementos atenderem a uma determinada condição.

7- **Remoção condicional de elementos em um conjunto**
Elabore um programa que remova elementos de um conjunto A com base em uma condição específica.

8- **Loop com conjuntos**
Crie um programa que utilize um loop for ou while para percorrer todos os elementos de um conjunto e realizar uma operação específica.

9- **Verificação de igualdade entre conjuntos**
Desenvolva um programa que verifique se dois conjuntos A e B são iguais.

10- **Geração de conjunto de potência**
Implemente um programa que gere o conjunto de potência de um conjunto dado.

11- **Remoção de duplicatas em um conjunto**
Crie um programa que remova duplicatas de um conjunto, mantendo apenas uma ocorrência de cada elemento.

12- **Contagem de elementos únicos em um conjunto**
Elabore um programa que conte quantos elementos únicos existem em um conjunto.

13- **Subconjunto próprio**
Desenvolva um programa que verifique se um conjunto A é um subconjunto próprio de outro conjunto B.

14- **Operações de conjunto em loop**
Implemente um programa que realize operações de conjunto em um loop até que uma condição específica seja atendida.

15- **Combinação de conjuntos com condicional**
Crie um programa que gere combinações de elementos de dois conjuntos A e B, incluindo apenas aquelas que atendem a uma determinada condição.

16- **Inversão de conjuntos**
Elabore um programa que inverta a ordem dos elementos em um conjunto.

17- **Interseção múltipla**
Desenvolva um programa que determine a interseção de vários conjuntos simultaneamente.

18- **Adição de elementos em um conjunto até uma condição ser atendida**
Implemente um programa que adicione elementos em um conjunto A até que uma condição específica seja atendida.

19- **Subconjunto comum**
Crie um programa que encontre o subconjunto comum a dois conjuntos A e B.

20- **Contagem regressiva com conjuntos**
Elabore um programa que, utilizando um conjunto, realize uma contagem regressiva de um número específico até zero.

COLLECTION

01- **Manipulação de Listas:**
Crie uma função que recebe uma lista de números e retorna a média dos valores presentes na lista.

02- **Operações com Dicionários:**
Desenvolva um programa que receba um dicionário com nomes e idades, e retorne o nome da pessoa mais velha.

03- **Filtragem de Dados:**
Escreva uma função que recebe uma lista de palavras e retorna apenas aquelas que são palíndromas.

04- **Contagem com Counter:**
Utilize a classe Counter para contar a frequência de cada letra em uma string.

05- **Estruturas Condicionais e List Comprehension:**
Crie uma expressão em list comprehension que retorna os quadrados dos números pares em uma lista.

06- **Mapeamento de Dados:**
Implemente uma função que recebe uma lista de números e retorna uma lista contendo o quadrado de cada número.

07- **Utilizando Set:**
Escreva um programa que verifica se duas listas têm elementos comuns, utilizando conjuntos (sets).

08- **Desafio com While:**
Crie um programa que solicita ao usuário adivinhar um número aleatório gerado pelo computador, dando dicas de "maior" ou "menor" até acertar.

09- **Filtragem de Listas com Enumerate:**
Implemente uma função que retorna os índices dos elementos maiores que 10 em uma lista.

10- **Agrupamento com defaultdict:**
Utilize um defaultdict para contar a quantidade de vezes que cada letra aparece em uma string.

11- **Operações com Tuplas:**
Escreva uma função que recebe duas tuplas e retorna uma nova tupla contendo os elementos comuns a ambas.

12- **Condições Aninhadas:**
Desenvolva um programa que classifica um aluno com base em suas notas, considerando conceitos como A, B, C, etc.

13- **Compreensão de Conjuntos:**
Crie uma expressão de compreensão de conjuntos que retorna os caracteres únicos em uma string.

14- **Contagem com While:**
Implemente um programa que conta quantos números pares existem em uma sequência, utilizando um loop while.

15- **Manuseio de Pilhas com Listas:**
Desenvolva uma função que verifica se uma expressão matemática contendo parênteses está balanceada.

16- **Ordenação Personalizada:**
Utilize a função sorted para ordenar uma lista de palavras pelo número de caracteres.

17- **Desafio com Zip:**
Escreva um programa que combina duas listas em um dicionário, utilizando a função zip.

18- **Aprimorando Listas com Itertools:**
Utilize a função permutations do módulo itertools para gerar todas as permutações de uma lista.

19- **Utilizando Break e Continue:**
Crie um programa que encontra o primeiro número primo em uma lista de números, utilizando break e continue.

20- **Manipulação de Filas com Deque:**
Implemente uma fila utilizando a classe deque do módulo collections e realize operações de enfileirar e desenfileirar elementos.