Lista de Revisão - Tipos de Dados, Coleções, Estruturas Condicionais, Estruturas de Repetição, Funções e Escopo de Funções

Instruções:

Para realizar a atividade, você deve seguir as seguintes instruções:

1. Crie um novo repositório no GitHub para a atividade.
2. Clone o repositório em sua máquina local usando o Git.
3. Crie um arquivo para cada seção (Tipos de Dados, Coleções, Funções, Escopo de Funções, Estruturas Condicionais e Estruturas de Repetição) e responda, pelo menos 2 questões de cada seção, ou 4 questões da seção “Combo”.
4. Para cada questão, inclua o código em Python e uma breve explicação do que o programa faz.
5. Após finalizar a atividade comite as mudanças usando o comando git commit -m "Respostas da atividade sobre programação em Python" ou utilizando a ferramenta GitHub Desktop.
6. Envie as mudanças para o repositório remoto no GitHub usando o comando git push origin main.
7. Verifique se as respostas foram enviadas corretamente no repositório remoto no GitHub.

Lembre-se de utilizar boas práticas de programação, como comentar o código e usar nomes de variáveis descritivos. Além disso, é importante responder, quando possível, pelo menos 3 questões de cada seção para demonstrar seu conhecimento em diferentes áreas da programação em Python. Boa sorte!

Checklist de Conhecimentos:

Utilize a lista de conhecimentos abaixo para identificar aspectos das aulas anteriores que precisam ser revisados.

Tipos de Dados:

- Entender os tipos básicos de dados em Python (int, float, bool, str)
- Conhecer os tipos compostos de dados em Python (list, tuple, set, dict)
- Compreender as diferenças entre tipos mutáveis e imutáveis em Python
- Saber converter entre tipos de dados em Python

Coleções:

- Saber criar e manipular listas, tuplas, sets e dicionários
- Entender como funcionam as operações de acesso, inserção e remoção de elementos em cada tipo de coleção
- Compreender as diferenças entre os tipos de coleções em Python
- Saber como usar a compreensão de listas e dicionários em Python

Estruturas Condicionais:

- Saber criar e usar as estruturas condicionais if/else e elif em Python
- Entender como usar operadores lógicos (and, or, not) em Python
- Saber como usar operadores de comparação (==, !=, <, >, <=, >=) em Python

Estruturas de Repetição:

- Saber criar e usar as estruturas de repetição for e while em Python
- Entender como usar a instrução "break" para interromper um loop em Python
- Saber como usar a instrução "continue" para pular para a próxima iteração de um loop em Python

Funções:

- Saber criar funções em Python
- Compreender os conceitos de parâmetros e argumentos em Python
- Entender o escopo de variáveis em Python
- Saber retornar valores a partir de funções em Python

Escopo de Funções:

- Entender a diferença entre variáveis globais e locais em Python
- Saber como acessar e modificar variáveis globais dentro de uma função em Python
- Compreender o uso da palavra-chave "global" em Python

Lembre-se que esta é apenas uma lista básica de conhecimentos e que há muitos outros conceitos importantes em Python que podem ser explorados!

Lista de Questões:

Tipos de Dados

1. Escreva um programa que solicite ao usuário um número inteiro e imprima o dobro desse número.

2. Escreva um programa que solicite ao usuário o raio de um círculo e calcule a área desse círculo.

3. Escreva um programa que solicite ao usuário seu nome e idade e imprima uma mensagem personalizada com essas informações.

Coleções

1. Escreva um programa que leia uma lista de números inteiros e imprima o maior e o menor número da lista.

2. Escreva um programa que leia uma lista de nomes e crie um dicionário onde a chave é o nome e o valor é o número de vezes que o nome aparece na lista.

3. Escreva um programa que leia uma lista de números inteiros e remova todos os valores duplicados. Em seguida, imprima a lista sem os valores duplicados.

Estruturas Condicionais

1. Escreva um programa que solicite ao usuário um número e imprima se ele é positivo, negativo ou zero.

2. Escreva um programa que leia o nome e a idade de uma pessoa e imprima uma mensagem personalizada com base na idade. Se a idade for menor que 18 anos, imprima "Você é menor de idade". Se a idade estiver entre 18 e 65 anos, imprima "Você é adulto". Caso contrário, imprima "Você é idoso".

3. Escreva um programa que solicite ao usuário a nota de um aluno em uma prova e imprima a mensagem "Aprovado" se a nota for maior ou igual a 7, "Reprovado" se a nota for menor que 5 e "Recuperação" se a nota estiver entre 5 e 7.

4. Escreva um programa que solicite ao usuário a idade e o sexo de uma pessoa e imprima uma mensagem personalizada com base nas seguintes condições:

    - Se a pessoa for do sexo feminino e tiver menos de 25 anos, imprima "Você é uma jovem mulher".

    - Se a pessoa for do sexo feminino e tiver 25 anos ou mais, imprima "Você é uma mulher adulta".

    - Se a pessoa for do sexo masculino e tiver menos de 25 anos, imprima "Você é um jovem homem".

- Se a pessoa for do sexo masculino e tiver 25 anos ou mais, imprima "Você é um homem adulto".

Estruturas de Repetição

1. Escreva um programa que imprima todos os números ímpares entre 1 e 50.
2. Escreva um programa que leia uma lista de números inteiros e imprima a média desses números.
3. Escreva um programa que solicite ao usuário um número e imprima a tabuada desse número até o valor 10.
4. Escreva um programa que solicite ao usuário um número e imprima todos os números primos menores que esse número.

Funções

1. Escreva uma função que receba uma lista de números inteiros e retorne o maior número da lista.
2. Escreva uma função que receba uma lista de palavras e retorne uma lista contendo apenas as palavras que começam com a letra "a".
3. Escreva uma função que receba uma lista de números inteiros e retorne a soma dos números pares da lista.
4. Escreva uma função que receba uma lista de dicionários contendo informações sobre pessoas (nome, idade, cidade) e retorne uma lista contendo apenas os nomes das pessoas que moram em uma cidade específica.

Escopo de Funções

1. Escreva um programa que solicite ao usuário dois números e imprima a soma, a subtração, a multiplicação e a divisão desses números. Crie funções separadas para cada operação matemática.
2. Escreva um programa que solicite ao usuário um número e imprima se esse número é par ou ímpar. Crie uma função para determinar se um número é par e outra função para determinar se um número é ímpar.
3. Escreva uma função que receba uma lista de números inteiros como parâmetro e retorne a média dos números. A função deve verificar se a lista está vazia e retornar None caso esteja. Em seguida, utilize uma variável global para contar quantas vezes a função foi chamada e imprimir o valor da contagem ao final de cada chamada.
4. Escreva uma função que calcule o fatorial de um número inteiro n. A função deve utilizar uma variável local para armazenar o resultado e uma estrutura de repetição para calcular o fatorial. Em seguida, utilize uma variável global para contar quantas

vezes a função foi chamada e imprimir o valor da contagem ao final de cada chamada.

5. Escreva uma função que receba uma lista de strings como parâmetro e retorne a string com o maior número de caracteres. A função deve utilizar uma variável local para armazenar a string com o maior número de caracteres e uma estrutura de repetição para percorrer a lista. Em seguida, utilize uma variável global para contar quantas vezes a função foi chamada e imprimir o valor da contagem ao final de cada chamada.
6. Escreva uma função que receba um número inteiro como parâmetro e retorne True se o número for primo e False caso contrário. A função deve utilizar uma variável local para armazenar o resultado e uma estrutura de repetição para verificar se o número é divisível por outro número. Em seguida, utilize uma variável global para contar quantas vezes a função foi chamada e imprimir o valor da contagem ao final de cada chamada.

Combo de conteúdos

A realização de 4 questões, de forma correta, dessa seção isentam o aluno de ter que realizar as outras seções.

1. Escreva uma função que receba como parâmetro uma lista de números inteiros e retorne a soma dos números pares na lista. Em seguida, utilize um laço de repetição para solicitar ao usuário uma lista de números inteiros, chame a função e imprima o resultado.
2. Escreva uma função que receba como parâmetro uma string e verifique se a string é um palíndromo (ou seja, pode ser lida da mesma forma de trás para frente). Em seguida, utilize uma estrutura de repetição para solicitar ao usuário uma string, chame a função e imprima se a string é um palíndromo ou não.
3. Escreva uma função que receba como parâmetro uma lista de strings e retorne a quantidade de strings que possuem mais de 5 caracteres. Em seguida, utilize uma estrutura condicional para perguntar ao usuário se ele deseja adicionar mais strings à lista, e utilize um laço de repetição para solicitar ao usuário as novas strings, chame a função e imprima o resultado.
4. Escreva uma função que receba como parâmetro uma lista de números inteiros e retorne o número máximo na lista. Em seguida, utilize uma estrutura de repetição para solicitar ao usuário uma lista de números inteiros, chame a função e imprima o resultado. Se a lista estiver vazia, a função deve retornar None e o programa deve imprimir uma mensagem informando que a lista está vazia.
5. Escreva uma função que receba como parâmetro um número inteiro n e retorne uma lista com os n primeiros números da sequência de Fibonacci. Em seguida, utilize uma estrutura condicional para perguntar ao usuário se ele deseja gerar a sequência

de Fibonacci para outro número, e utilize um laço de repetição para solicitar ao usuário os novos valores de n, chame a função e imprima o resultado.

Dicas

1. Para a questão que pede para encontrar o menor valor em uma lista, utilize a função min(). Para a questão que pede para encontrar a média, utilize a função sum() e a função len().
2. Para a questão que pede para contar a quantidade de vezes que uma letra aparece em uma string, utilize a função count(). Para a questão que pede para inverter uma string, utilize o slicing com passo negativo [::-1].
3. Para a questão que pede para encontrar o maior número em uma lista, utilize a função max(). Para a questão que pede para encontrar o segundo menor número em uma lista, ordene a lista com a função sorted() e retorne o segundo elemento.
4. Para a questão que pede para somar os dígitos de um número, converta o número para string e utilize a função sum() em uma list comprehension para converter os caracteres de volta para inteiros. Para a questão que pede para verificar se um número é primo, utilize uma estrutura de repetição para verificar se o número é divisível por algum número entre 2 e a sua raiz quadrada.
5. Para a questão que pede para criar uma lista com os números pares entre 1 e 100, utilize uma estrutura de repetição com um passo de 2. Para a questão que pede para criar uma lista com os números primos entre 1 e 100, utilize uma estrutura de repetição e a função is_prime() criada anteriormente.
6. Para as questões que envolvem escopo de funções, lembre-se que variáveis globais podem ser acessadas e modificadas dentro de uma função, mas é boa prática evitar o uso excessivo de variáveis globais. Para a questão que pede para contar a quantidade de vezes que a função foi chamada, utilize uma variável global e incremente seu valor no final de cada chamada.