Atividade 2: Sintaxe da Linguagem Python

O que são palavras-chave em Python? Inclua alguns exemplos.

Palavras-chave em Python são termos reservados que possuem um significado pré-definido e uma função específica na linguagem. Elas não podem ser usadas como nomes de variáveis, funções ou quaisquer outros identificadores definidos pelo usuário, pois isso causaria um erro de sintaxe. São elementos fundamentais para a estrutura e o funcionamento dos programas em Python.

Alguns exemplos de palavras-chave incluem:

- if: Usada para iniciar uma instrução condicional.
- else: Usada em conjunto com if para definir um bloco de código a ser executado se a condição do if for falsa.
- for: Usada para iniciar um loop de iteração.
- while: Usada para iniciar um loop que continua enquanto uma condição for verdadeira.
- def: Usada para definir uma função.
- class: Usada para definir uma classe.
- import: Usada para importar módulos.
- True: Representa o valor booleano verdadeiro.
- False: Representa o valor booleano falso.
- None: Representa um valor nulo.

O que são identificadores e explique as regras para nomeação em Python.

Identificadores em Python são nomes dados a variáveis, funções, classes, módulos e outros objetos. Eles servem para identificar e referenciar esses elementos no código.

As regras para nomeação de identificadores em Python são:

- **Caracteres Permitidos:** Os identificadores podem consistir em letras (maiúsculas e minúsculas), dígitos (0-9) e o caractere underscore (_).
- **Primeiro Caractere:** O primeiro caractere de um identificador deve ser uma letra (maiúscula ou minúscula) ou um underscore (_). Não pode começar com um dígito.
- **Case-sensitive:** Python é *case-sensitive*, o que significa que minhavariavel é diferente de MinhaVariavel.
- **Palavras-chave:** Identificadores não podem ser palavras-chave da linguagem (conforme explicado acima).
- **Espaços:** Identificadores não podem conter espaços em branco.
- **Caracteres Especiais:** Não é permitido o uso de outros caracteres especiais como !, @, #, $, %, etc.

Exemplos Válidos:

- nome_usuario
- temperatura
- _contador
- valorTotal
- x

Exemplos Inválidos:

- 1nome (começa com dígito)
- minha variavel (contém espaço)
- class (palavra-chave)
- total$ (contém caractere especial)

A importância da indentação e como ela é usada em Python. Use um exemplo.

A indentação é de suma importância em Python, pois ela define a estrutura do código e os blocos de execução. Ao contrário de outras linguagens que utilizam chaves {} ou palavras-chave como begin e end para delimitar blocos, Python utiliza o número de espaços em branco ou tabulações no início de cada linha para indicar a hierarquia do código. Uma indentação consistente e correta é fundamental para que o interpretador Python entenda a lógica do programa.

Como a indentação é usada:

- **Blocos de Código:** A indentação agrupa declarações que fazem parte de um mesmo bloco de código, como os corpos de loops (for, while), condicionais (if, else, elif), funções (def) e classes (class).
- **Níveis de Aninhamento:** O aumento do nível de indentação (geralmente 4 espaços ou uma tabulação) indica que as linhas de código pertencem a um bloco aninhado. A diminuição do nível de indentação indica o fim de um bloco.
- **Erros de Sintaxe:** Indentação incorreta resultará em IndentationError ou TabError, que são erros de sintaxe.

Exemplo:

```
idade = 18
if idade >= 18:
    print("Você é maior de idade.") # Este print está indentado, pertence ao bloco do if
    print("Pode tirar a carteira de motorista.") # Este print também está indentado, pertence ao bloco do if
else:
    print("Você é menor de idade.") # Este print está indentado, pertence ao bloco do else
print("Fim do programa.") # Este print não está indentado, não pertence a nenhum dos blocos condicionais
```

Neste exemplo:

- As linhas print("Você é maior de idade.") e print("Pode tirar a carteira de motorista.") estão indentadas com o mesmo nível após o if, indicando que fazem parte do bloco de código a ser executado se idade >= 18 for True.
- A linha print("Você é menor de idade.") está indentada após o else, indicando que faz parte do bloco de código a ser executado se a condição do if for False.
- A linha print("Fim do programa.") não está indentada, o que significa que ela será executada independentemente da condição do if/else.

Cite exemplos de problemas de sintaxe que você teve ao tentar programar em Python.

Ao programar em Python, alguns problemas de sintaxe comuns que tive incluem:

- **IndentationError:** Este é um dos mais frequentes. Ocorre quando a indentação está inconsistente (misturar espaços e tabulações) ou incorreta (um nível de indentação inesperado). Por exemplo, esquecer de indentar uma linha após um if ou for.# Exemplo de IndentationError

```
if True:
print("Isso causará um erro de indentação.")
```

- **SyntaxError: invalid syntax:** Este erro é genérico e pode ocorrer por diversas razões, como:
    - **Parênteses/Colchetes/Chaves Incompletos:** Esquecer de fechar um parêntese em uma função, um colchete em uma lista ou uma chave em um dicionário.# Exemplo de SyntaxError por parêntese faltando

      ```
      minha_lista = [1, 2, 3
      print(minha_lista)
      ```

    - **Uso de Palavras-chave como Identificadores:** Tentar usar uma palavra-chave reservada como nome de variável.# Exemplo de SyntaxError por palavra-chave usada como identificador

      ```
      def = 10
      ```

    - **Operadores Incorretos:** Usar operadores de forma inadequada.# Exemplo de SyntaxError por operador incorreto

      ```
      x ==+ 5
      ```

    - **Strings Não Fechadas:** Esquecer de fechar as aspas em uma *string*.# Exemplo de SyntaxError por string não fechada

      ```
      mensagem = "Olá mundo
      ```

- **NameError: name '...' is not defined** (Embora tecnicamente não seja um SyntaxError, é um erro de tempo de execução muito comum causado por erros de digitação ou referências a variáveis não definidas, que podem ser confundidas com problemas de sintaxe por iniciantes).# Exemplo de NameError

print(minha_variavel) # se minha_variavel não foi definida

- **TypeError** (similar ao NameError, é um erro de tempo de execução, mas frequentemente ocorre devido a operações inválidas entre tipos de dados, o que pode ser confundido com um problema de sintaxe se a causa não for clara).# Exemplo de TypeError
  resultado = "5" + 2 # tentará concatenar uma string com um inteiro