FASE 1

# Símbolos

## Operadores Aritméticos

|  |  |  |
| --- | --- | --- |
| Operador | Nome | Função |
| + | **Adição** | Realiza a soma de ambos operandos. |
| - | **Subtração** | Realiza a subtração de ambos operandos. |
| \* | **Multiplicação** | Realiza a multiplicação de ambos operandos. |
| / | **Divisão** | Realiza a Divisão de ambos operandos. |
| // | **Divisão inteira** | Realiza a divisão entre operandos e a parte decimal de ambos operandos. |
| % | **Módulo** | Retorna o resto da divisão de ambos operandos. |
| \*\* | **Exponenciação** | Retorna o resultado da elevação da potência pelo outro. |

## Operadores Relacionais

|  |  |  |
| --- | --- | --- |
| Operador | Nome | Função |
| == | Igual a | Verifica se um valor é igual ao outro |
| != | Diferente de | Verifica se um valor é diferente ao outro |
| > | Maior que | Verifica se um valor é maior que outro |
| >= | Maior ou igual | Verifica se um valor é maior ou igual ao outro |
| < | Menor que | Verifica se um valor é menor que outro |
| <= | Menor ou igual | Verifica se um valor é menor ou igual ao outro |

## Operadores booleanos

|  |  |
| --- | --- |
| Operador | Definição |
| and | Retorna True se ambas as afirmações forem verdadeiras |
| or | Retorna True se uma das afirmações for verdadeira |
| not | Retorna Falso se o resultado for verdadeiro |

## Símbolos de Atribuição

|  |  |
| --- | --- |
| Operador | Equivalente a |
| = | x = 1 |
| += | x = x + 1 |
| -= | x = x - 1 |
| \*= | x = x \* 1 |
| /= | x = x / 1 |
| %= | x = x % 1 |

## Símbolos identificadores de tipos de dados

**Inteiros (int):** Representados por números inteiros.

**Ponto flutuante (float):** Representados por números com ponto decimal.

**String (str):** Sequência de caracteres.

**Lista (list):** Estrutura de dados ordenada e mutável que pode conter elementos de tipos variados.

**Tupla (tuple):** Estrutura de dados ordenada, mas imutável.

**Dicionário (dict):** Estrutura de dados não ordenada, que armazena pares de chave e valor.

**Conjunto (set):** Estrutura de dados não ordenada, sem elementos duplicados.

**Booleano (bool):** Tipo que pode ter dois valores: True ou False.

**NoneType (None):** Representa a ausência de valor.

## Símbolo identificador de início de bloco

Em Python, o início de um bloco de código não é indicado por um símbolo específico como em outras linguagens (por exemplo, { em C ou Java). Em vez disso, Python utiliza a indentação para marcar o início de um bloco de código.

# Palavras

## Identificadora de blocos

**def - Para definir uma função**

Usada para iniciar um bloco de código que define uma função.

**if / elif / else - Para estruturas condicionais**

Usadas para iniciar um bloco de código condicional. A indentação indica os blocos de código que serão executados dependendo da condição.

**for - Para laços de repetição (loop)**

Usada para iniciar um bloco de código em um laço for.

**while - Para laços de repetição baseados em condição**

Usada para iniciar um bloco de código em um laço while.

**try / except / finally - Para tratamento de exceções**

Usadas para iniciar um bloco de código que lida com exceções.

**class - Para definir uma classe**

Usada para iniciar um bloco de código que define uma classe.

**with - Para trabalhar com gerenciadores de contexto (geralmente usado em manipulação de arquivos)**

Usada para iniciar um bloco de código onde um gerenciador de contexto é usado (como abrir arquivos).

**async / await - Para programação assíncrona**

Usadas para definir blocos assíncronos, como funções assíncronas e chamadas assíncronas.

## Tipos de dados

**None -** Representa um tipo especial chamado NoneType, que é usado para indicar a ausência de valor ou um valor nulo.

**True / False -** São palavras-chave que representam os valores booleanos True e False, que pertencem ao tipo bool.

**Is -** A palavra-chave is é usada para comparar a identidade de dois objetos (ou seja, se eles são o mesmo objeto em memória), e não os seus valores.

**Del -** A palavra-chave del é usada para excluir objetos, o que pode afetar tipos de dados como listas, dicionários, ou até variáveis simples.

**and / or -** São operadores lógicos que atuam sobre valores booleanos (bool), mas não são tipos de dados em si.

**In -** A palavra-chave in é usada para verificar a presença de um valor em uma sequência (como uma lista, tupla, ou string). Essa operação envolve tipos de dados como list, tuple, str, etc.

 **if, elif, else** — Condições que podem atuar sobre valores de tipos como bool.

 **for, while** — Laços que iteram sobre sequências de tipos como list, tuple, str, etc.

 **try, except, finally** — Usados para tratamento de exceções, o que pode envolver diferentes tipos de dados ao lidar com erros.

## Funções built-in

 **print()**  
Exibe uma saída no console.

 **len()**  
Retorna o comprimento (número de itens) de um objeto (como uma lista, string, etc.).

 **type()**  
Retorna o tipo de um objeto.

 **int()**  
Converte um valor para um inteiro.

 **float()**  
Converte um valor para um número de ponto flutuante (decimal).

 **str()**  
Converte um valor para uma string.

 **abs()**  
Retorna o valor absoluto de um número.

 **round()**  
Arredonda um número para o número de casas decimais especificado.

 **sum()**  
Retorna a soma de um iterável (como uma lista ou tupla).

 **max() / min()**  
Retorna o maior ou menor valor de um iterável, respectivamente.

 **sorted()**  
Retorna uma nova lista com os elementos de um iterável ordenados.

 **range()**  
Gera uma sequência de números, geralmente usada em loops.

 **input()**  
Lê uma linha de entrada do usuário (sempre retorna uma string).

 **isinstance()**  
Verifica se um objeto é uma instância de uma classe ou tipo específico.

 **help()**  
Exibe a documentação de um objeto ou função.

 **id()**  
Retorna o "ID" de um objeto (uma referência única na memória).

 **zip()**  
Combina dois ou mais iteráveis, retornando um iterador de tuplas.

 **all()**  
Retorna True se todos os elementos de um iterável forem verdadeiros (ou o iterável estiver vazio).

 **any()**  
Retorna True se algum elemento de um iterável for verdadeiro.

## Operadores booleanos

* **and:** Retorna True se ambas as expressões forem verdadeiras.
* **or:** Retorna True se pelo menos uma expressão for verdadeira.
* **not:** Inverte o valor lógico da expressão.

## Demais (import, from, in, as, etc)

* **Controle de fluxo:** if, else, elif, for, while, break, continue, pass
* **Definição de funções e classes:** def, return, class
* **Módulos e importação:** import, from, as
* **Tratamento de exceções:** try, except, finally
* **Manipulação de escopos:** global, nonlocal
* **Gerenciamento de contexto:** with
* **Funções anônimas:** lambda