# Objetivos da Aula

Definição dos elementos comuns em scripts Python.

```
def main():
if __name__ == "__main__":
main()
```

```
def main():
if __name__ == "__main__":
main()
```

Definição de dois elementos comuns em scripts Python:

**def main():** é a definição de uma função chamada main.

**def** - É a palavra-chave em Python usada para iniciar a definição de uma função.

**main** - É o nome dado a esta função. É um nome convencional, não uma palavra-chave ou um requisito obrigatório da linguagem Python em si (diferente de linguagens como C ou Java, onde main tem um significado especial para o compilador/runtime).

No entanto, é uma prática muito comum e recomendada usar main como o nome da função principal que contém a lógica central de execução do seu script.

**()** - Parênteses indicam que é uma função. Eles podem conter parâmetros se a função precisar receber dados, mas frequentemente a função main é definida sem parâmetros.

**:** - O dois-pontos marca o início do bloco de código (indentado) que pertence a essa função.

Propósito: A principal razão para usar def main(): é organização e modularidade. Agrupa o código principal do seu script em uma unidade lógica, separando-o de outras definições de funções, classes ou importações. Isso torna o script mais legível e fácil de entender, pois fica claro onde a execução principal começa.

```
def main():
if __name__ == "__main__":
main()
```

if __name__ == "__main__":

main()

Definição: Este é um idioma padrão em Python que verifica se o script está sendo executado diretamente ou se ele foi importado como um módulo em outro script.

__name__:

Esta é uma variável especial interna do Python.

O interpretador Python atribui automaticamente um valor a __name__ para cada script.

Se você executa o script diretamente (por exemplo, python meu_script.py no terminal),

o Python define __name__ com o valor da string "__main__".

Se você importa o script em outro módulo (por exemplo, **import meu_script** dentro de **outro_script.py**), o Python define __name__ com o nome do próprio módulo (neste caso, a string "meu_script").

```
def main():
if __name__ == "__main__":
main()
```

if __name__ == "__main__":

Esta linha está, portanto, verificando: "Este script está sendo executado como o programa principal?".

A condição será True apenas quando o script for executado diretamente.

Bloco Indentado (main()): O código dentro deste bloco if (neste caso, a chamada main()) só será executado se a condição for True, ou seja, se o script for o ponto de entrada principal da execução.

main()

Esta é a chamada para a função main que foi definida anteriormente com def main():

```
def main():
if __name__ == "__main__":
main()
```

Propósito:

O principal objetivo deste bloco é permitir que um script Python seja reutilizável. Você pode definir funções e classes no seu script que podem ser úteis para outros programas.

Ao usar **if __name__ == "__main__":**, você garante que a lógica principal do script (que pode incluir coisas como pedir input ao usuário, imprimir resultados, iniciar um processo) não seja executada automaticamente quando o script for simplesmente importado por outro módulo.

Apenas as definições (funções, classes) estarão disponíveis para o módulo importador, mas o código dentro do if não rodará indesejadamente.

Em Resumo:

**def main():** Define um bloco de código organizado como a lógica principal do script (por convenção).

**if __name__ == "__main__":** Verifica se o script está sendo executado diretamente.

A chamada **main()** dentro do if executa a lógica principal definida em def main(): apenas quando o script é executado diretamente.

Essa combinação é a maneira padrão e recomendada de estruturar scripts Python que podem ser executados sozinhos ou ter partes reutilizadas por importação.