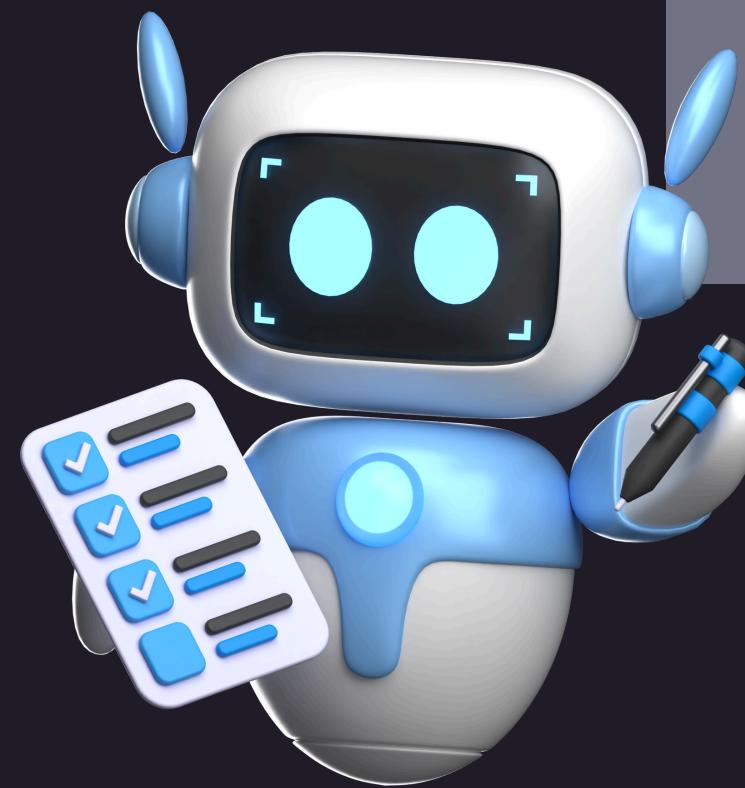




# PYTHON



# OPERADORES ARITMÉTICOS:

Operador	Operação
+	adição
-	subtração
*	multiplicação
/	Divisão
//	divisão de inteiros
**	potenciação
%	módulo (resto da divisão inteira)



# OPERADORES RELACIONAIS:

Operador	Operação
<code>==</code>	igualdade
<code>&gt;</code>	maior que
<code>&lt;</code>	menor que
<code>!=</code>	diferente
<code>&gt;=</code>	maior ou igual
<code>&lt;=</code>	menor ou igual

# OPERADORES LÓGICOS:

Operador	Operação
not	não
and	e
or	ou



- Algumas Funções Matemáticas.

```
# -*- coding: utf-8 -*-
```

```
import math
```

```
# Exponenciação
```

```
a = 3 ** 2
```

```
print(a)
```

```
# Radiciação
```

```
b = math.sqrt(a)
```

```
print(b)
```

```
print(math.pi)
```

```
n = 0
```

```
print(math.cos(n))
```

```
print(math.sin(n))
```



- Função print()
  - Exibe uma mensagem na tela do computador.

```
>>> print("Olá, mundo!")
Olá, mundo!
```

## Função input()

- Espera o usuário digitar um texto no teclado e pressionar <ENTER>.

```
>>> nome = input("Informe o seu nome: ")
Informe o seu nome: Maria
>>> print(nome)
Maria
```

The screenshot shows a Windows-style window titled "Exercício 1- soma de dois valores.py - C:\Users\2anoA\Documents\SENAI\LOPAL\Python\A". The menu bar includes "Arquivo", "Redimensionar", "Edição", and "File Edit Format Run Options Window Help". The code editor contains the following Python script:

```
#--*-coding: UTF-8 -*-  
  
print("Olá usuário. Você irá calcular dois números inteiros!")  
num1 = int(input("Digite o primeiro valor: "))  
num2 = int(input("Digite o segundo valor: "))  
soma = num1 + num2  
print("O resultado da soma desses dois números é: ", soma)
```



The screenshot shows a terminal window titled "IDLE Shell 3.13.1". The menu bar includes "Edit Shell Debug Options Window Help". The terminal output is as follows:

```
Python 3.13.1 (tags/v3.13.1:0671451, Dec 3 2024, 19:06:28) [MSC v.1942 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
= RESTART: C:\Users\2anoA\Documents\SENAI\LOPAL\Python\Aula 1\Exercício 1- soma  
de dois valores.py  
Olá usuário. Você irá calcular dois números inteiros!  
Digite o primeiro valor:
```

# IF, ELSE E ELIF

**If**- É utilizado para condicionar a execução de um código, ou seja, para executar um comando apenas se uma condição for verdadeira.

**else** - Especifica o que fazer, caso o resultado da avaliação da condição seja falso, ele permite executar alguns comandos se uma condição for verdadeira e outros se ela for falsa.

**elif** - Substitui um par else if, mas sem criar outro nível de estrutura. Permite verificar múltiplas condições sequencialmente.



# Exemplos:

## Elif

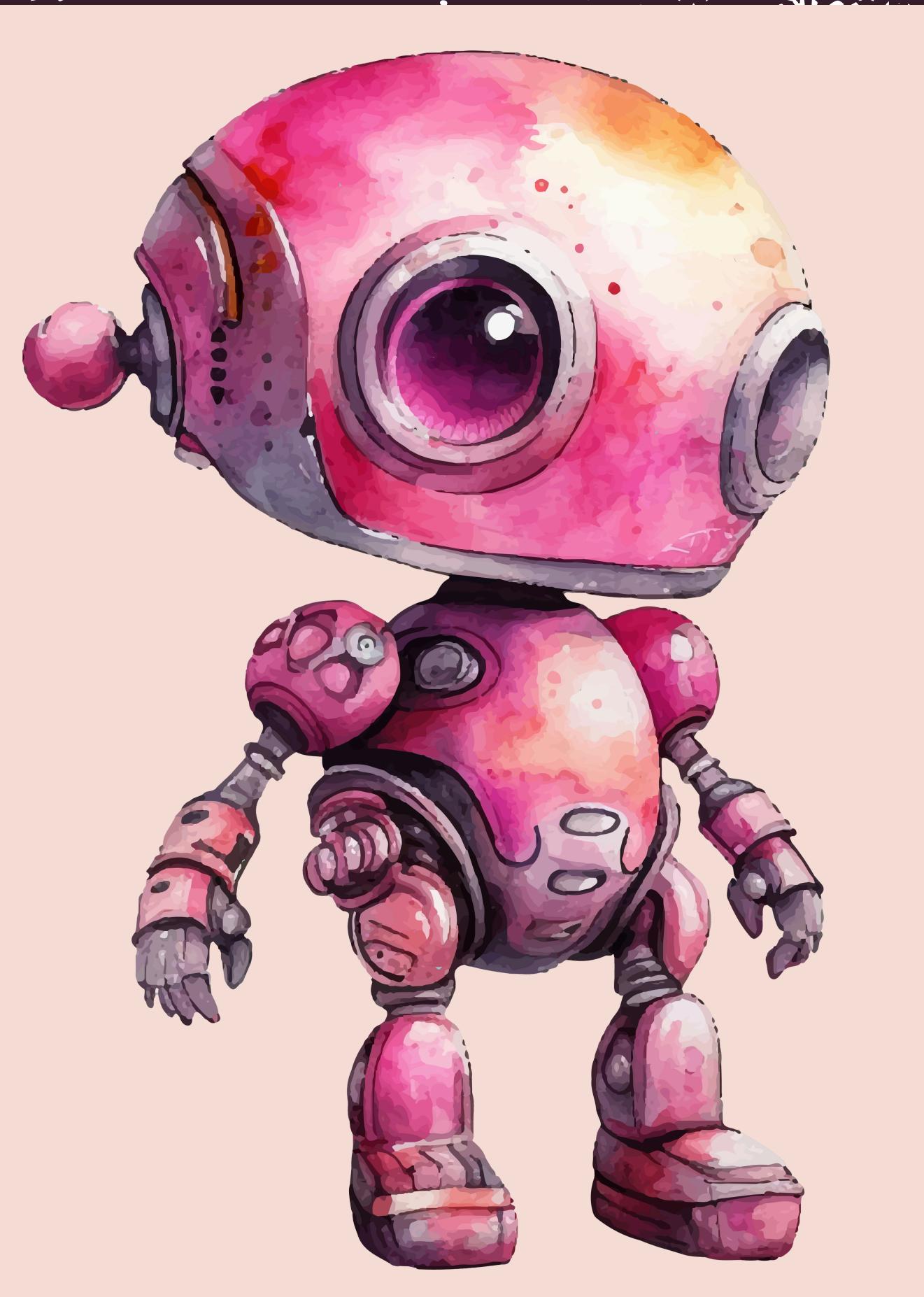
```
# -*- coding: UTF-8 -*-
categoria = int(input("Digite a categoria do produto: "))
if categoria == 1:
    preco = 10
elif categoria == 2:
    preco = 18
elif categoria == 3:
    preco = 23
elif categoria == 4:
    preco = 26
elif categoria == 5:
    preco = 31
else:
    print ("Categoria inválida, digite um valor entre 1 e 5!")
    preco = 0
print ("O preço do produto é: R$%6.2f" % preco)
```

## Condições - If

```
# -*- coding: UTF-8 -*-
a = int(input("Primeiro valor: "))
b = int(input("Segundo valor: "))
if a > b:
    print ("O primeiro número é o maior!")
if b > a:
    print ("O segundo número é o maior!")
```

## Else:

```
# -*- coding: UTF-8 -*-
idade = int(input("Digite a idade do seu carro: "))
if idade <= 3:
    print ("Seu carro é novo")
else:
    print ("Seu carro é velho")
```



# WHILE E WHILE TRUE

**While-** Define para quantas vezes vc quer repetir. O while repete um bloco de código enquanto uma condição for verdadeira.

**While True-** É infinito e para somente com o break. O While True repete indefinidamente até que uma instrução de interrupção seja acionada, ou seja, “break”.

# EXEMPLOS:

```
# -*- coding: UTF-8 -*-
```

```
n = int(input("Tabuada de: "))

x = 0
while x <= 10:
    print (n, "x", x, "=", (n * x))
    x = x + 1
```

```
# -*- coding: UTF-8 -*-
```

```
n = int(input("Tabuada de: "))

x = 0
while x <= 10:
    print (n, "x", x, "=", (n * x))
    x = x + 1
```

Usando global:

```
# -*- coding: UTF-8 -*-
```

```
a = 5 # variável global
```

```
def muda_e_imprime():

    global a # variável global
```

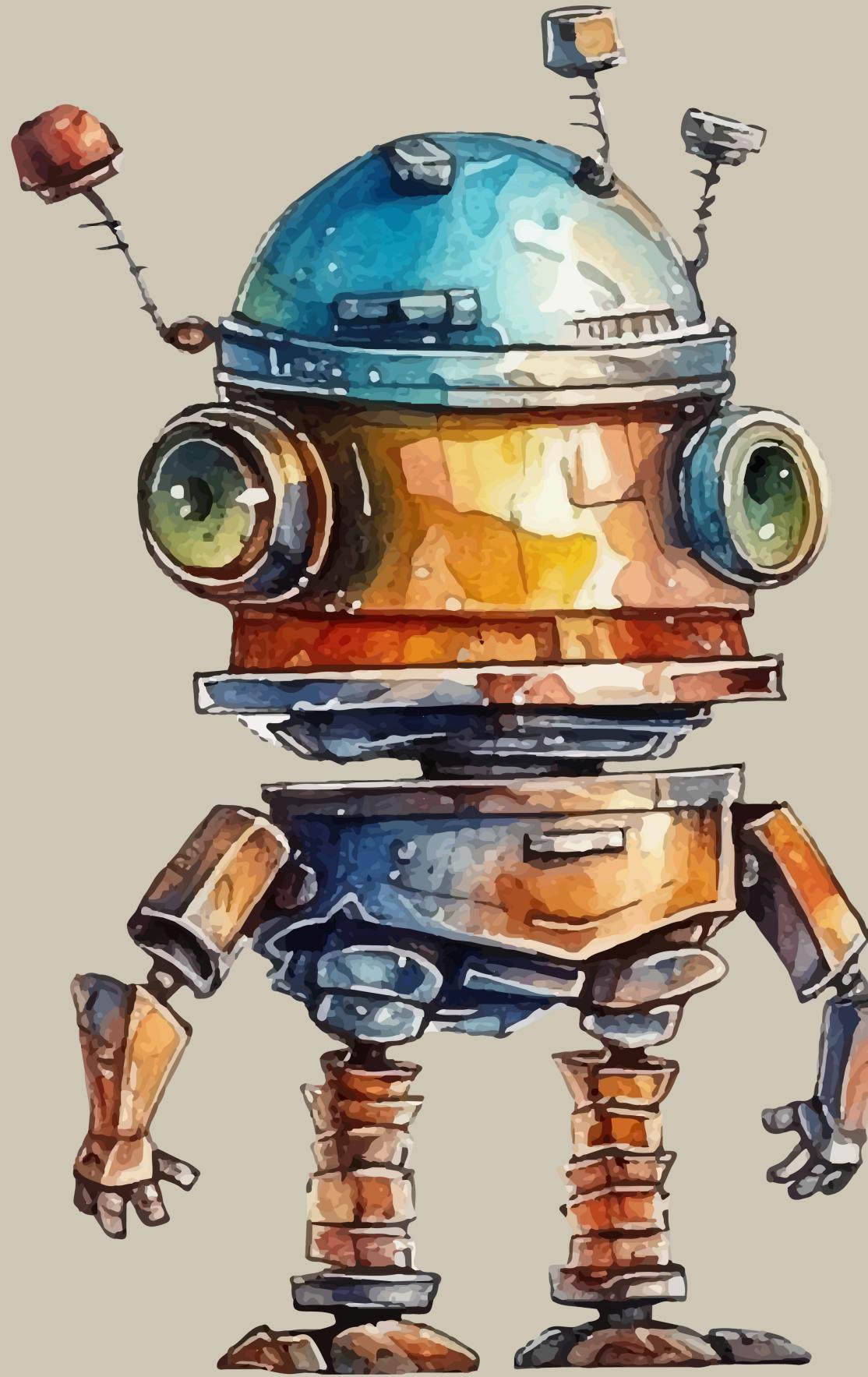
```
a = 7
```

```
print("a dentro da função: %d" % a)
```

```
print("a antes de mudar: %d" % a)
```

```
muda_e_imprime()
```

```
print("a depois de mudar: %d" % a)
```



# FOR V IN RANGE

- Ele permite repetir uma ação um determinado número de vezes.
- Retorna uma sequência de números e é imutável, o que significa que seu valor é fixo

# EXEMPLOS

## Exemplo 1 – Contagem de 1 a 5:

```
python

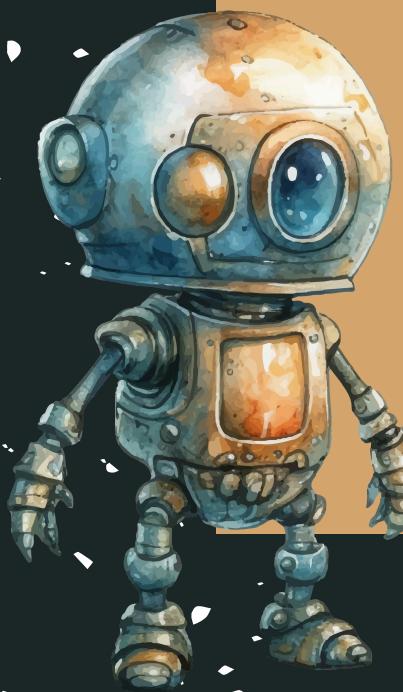
for i in range(1, 6):
    print(i)
.
```

## Exemplo 2 – Percorrendo uma lista:

```
python

frutas = ["Maçã", "Banana", "Uva"]

for fruta in frutas:
    print(fruta)
```



# DEF

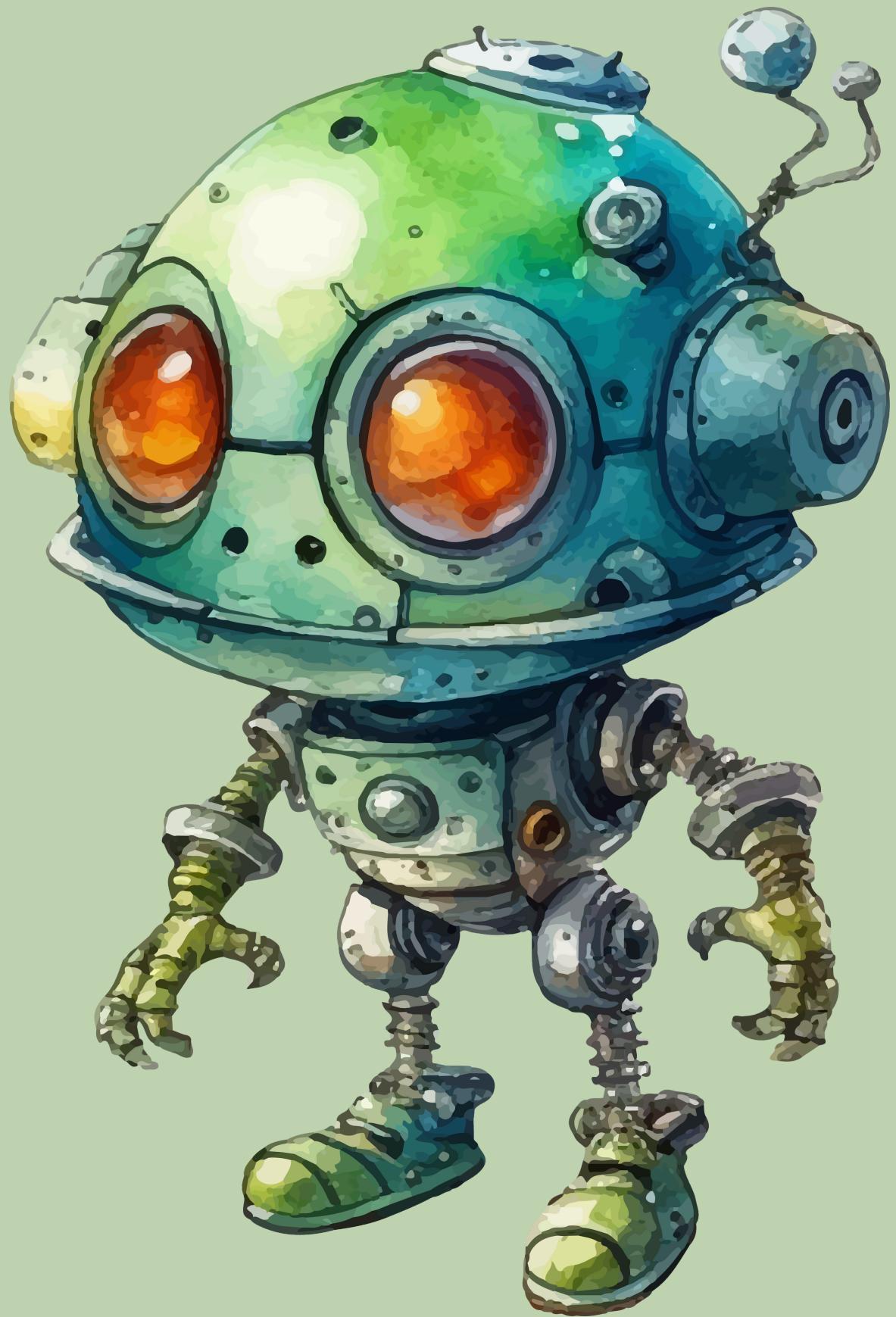
def deve ser utilizada para definir uma função e return para devolver algum valor.

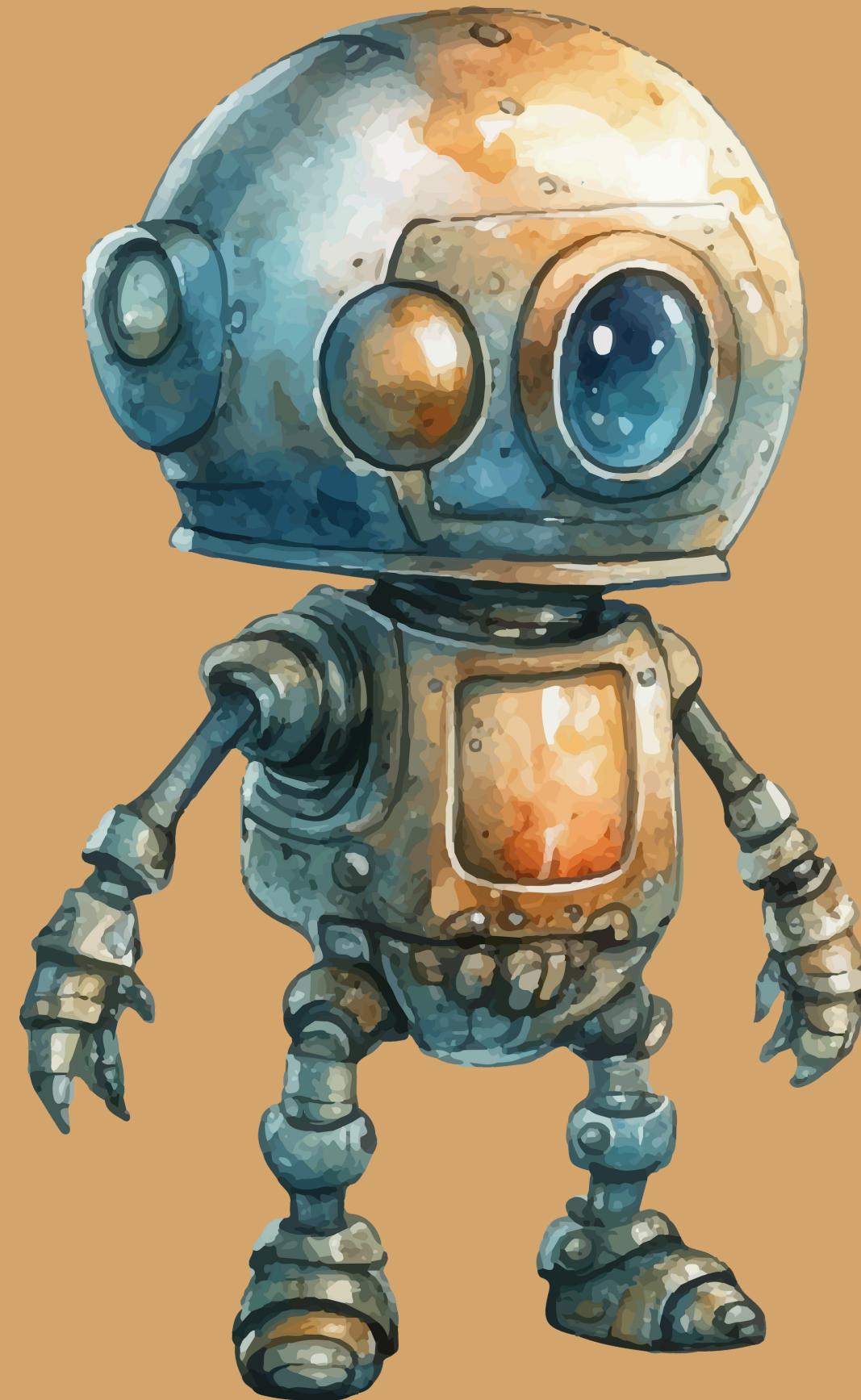
## Exemplo 2 – Função com parâmetro:

```
python

def soma(a, b):
    return a + b

resultado = soma(5, 3)
print("A soma é:", resultado)
```





# RETURN

Indica o valor que a função deve retornar quando chamada

# GLOBAL

- Definida fora de uma função.
- Pode ser vista e alterada por qualquer função do programa.
- Deve ser utilizada com cuidado, como por exemplo em constantes.

```
print("Olá,Mundo!")
```

" - para mensagem

()- toda função tem que ter

print- para escrever

números são usados para fazer cálculo

mensagem são usada para serem exibidas na tela.

exemplo correto:

```
print(7+4) = 11
```

exemplo errado:

Ao usar " o programa entende que é pra juntar os números, não para soma-los

```
print("7"+"4") = 74
```

toda mensagem tem que estar entre "



## INFORMAÇÕES EXTRAS:

- Use while para repetições infinitas que dependem de uma condição.
- Use for quando souber o número de repetições.

