



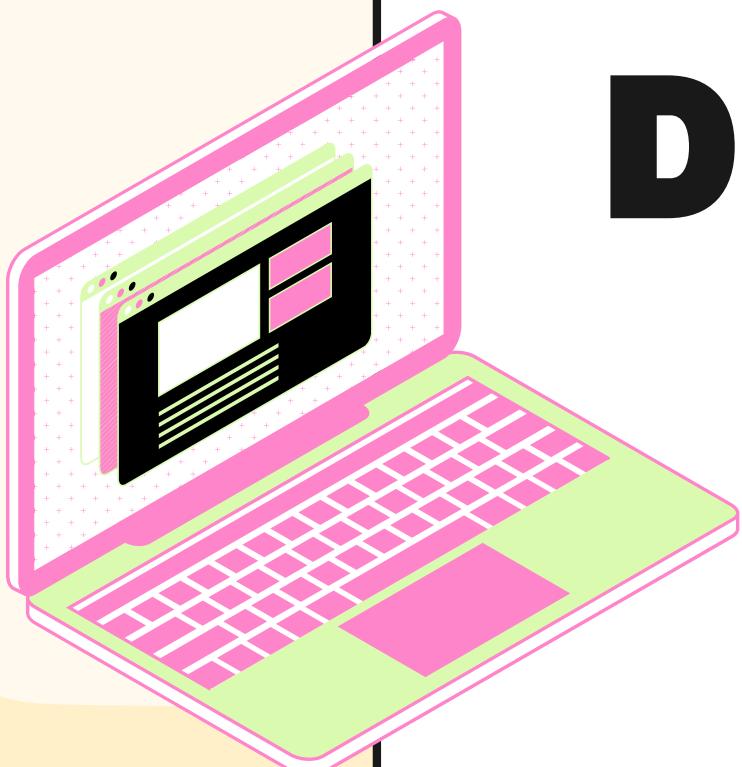
#INCLUDE <MENINAS.UFF>



Include
Meninas

OFICINA DE PROGRAMAÇÃO COM PYTHON

Aula 3



Fundação Carlos Chagas Filho de Amparo
à Pesquisa do Estado do Rio de Janeiro



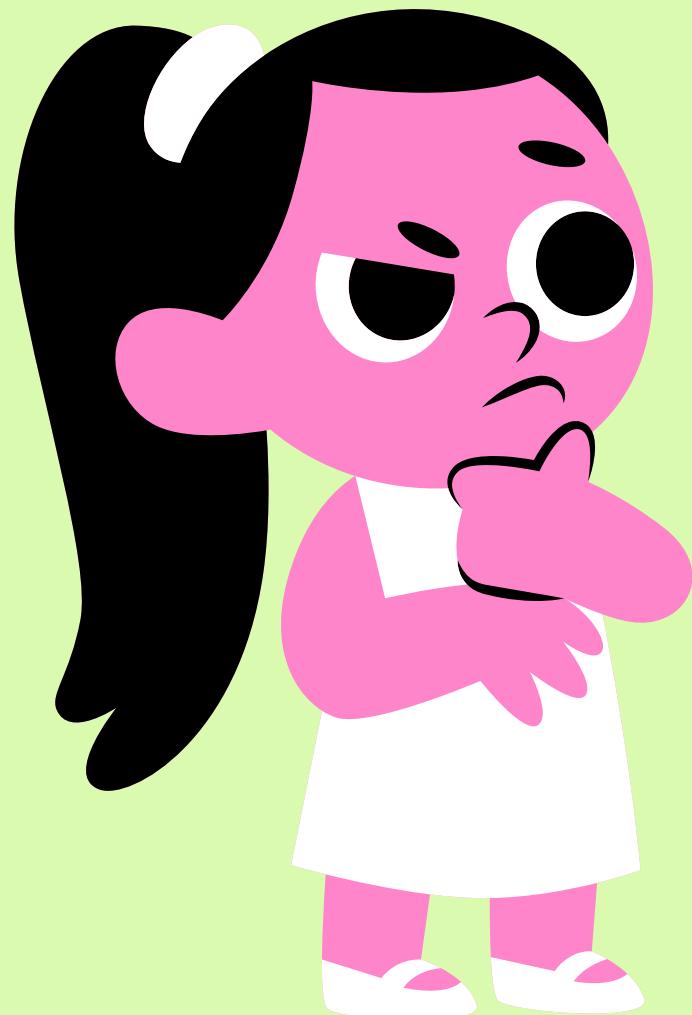
PRÓ-REITORIA DE EXTENSÃO



Universidade
Federal
Fluminense



O QUE NÓS VIMOS ANTERIORMENTE?



THONNY:

The screenshot shows the Thonny Python IDE interface. In the top-left corner, there is a code editor window titled "Contador.py". Inside the code editor, there is handwritten red text: "le 2" above line 2, and "1" next to line 8. The code itself is:

```
1
2
3
4
5
6 Contador = 0
7
8 for i in range(5):
9     Contador+=1
10    print(Contador)
11
12
```

In the bottom-left corner, there is a shell window titled "Shell". It contains the command ">>> %Run Contador.py" and the output "1\n2\n3\n4\n5". Handwritten red text "3" is written next to the number 3 in the output.

1. Para **escrever** o código.
2. Para **executar** o código. (▶)
3. Para ver o **resultado**.

Comando Input

O comando **input** permite que qualquer um que execute o programa atribua um valor à variável. Ou seja, o **valor** da variável **não fica restrito** a programadora.

Comando Input

O comando input cria uma variável que espera receber do usuário um determinado tipo de dado, que pode ser:

FLOAT: número decimal

INT: número inteiro

STRING: “texto”

Comando Input - Formato

Caso você queira receber um número **INTEIRO** do usuário, seu comando input terá essa forma:

```
A = int(input("Digite um número INTEIRO: "))
```

Comando Input - Formato

Caso você queira receber um número **DECIMAL** do usuário, seu comando input terá essa forma:

```
B = float(input("Digite um número DECIMAL: "))
```

Comando Input - Formato

Caso você queira receber um **TEXTO** do usuário,
seu comando input terá essa forma:

```
C = input("Digite o nome de sua cidade: ")
```

Comando Input - EXEMPLO

Imagine que você seja a programadora de uma papelaria.

Seu programa deve perguntar ao cliente **o que** ele deseja comprar da loja e a **quantidade** desse item.

Como o seu algoritmo ficaria?



Comando Input - EXEMPLO

```
print("Bem vindo(a) à papelaria!")

compra = input("O que você deseja comprar? ")

quantidade = int(input("Qual a quantidade desejada? "))

print("Você deseja comprar ", quantidade, compra)
```

Operadores Numéricos

- Podemos realizar cálculos com as variáveis desde que os seus valores **sejam apenas numéricos**.

Operação	Símbolo	Exemplo
Soma	+	<code>print(A+B)</code> -> 7
Subtração	-	<code>print(A-B)</code> -> 3

Operação	Símbolo	Exemplo
Multiplicação	*	<code>print(A*B)</code> -> 10
Divisão	/	<code>print(A/B)</code> -> 2.5

Estruturas Condicionais

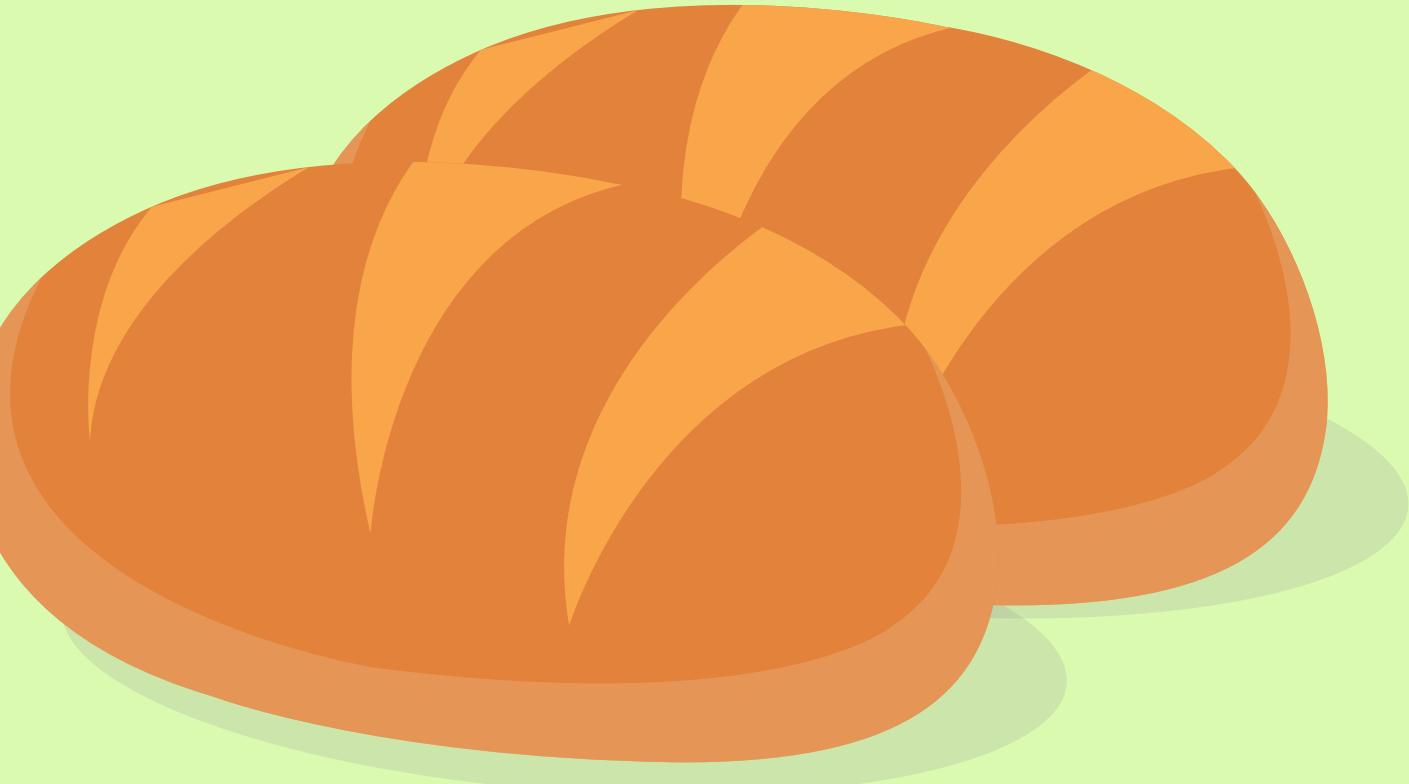
- As estruturas condicionais (também chamadas de estruturas de decisão) permitem que um programa apresente mais de um caminho de execução, chegando a resultados, muitas vezes, diferentes.
- **Elas são:**
 1. **IF** (do inglês, “**se**”);
 2. **ELSE** (do inglês, “**senão**”).

Estrutura Condicional: EXEMPLO

Código:

```
if (pão estiver fresco):  
    compre três reais de pães
```

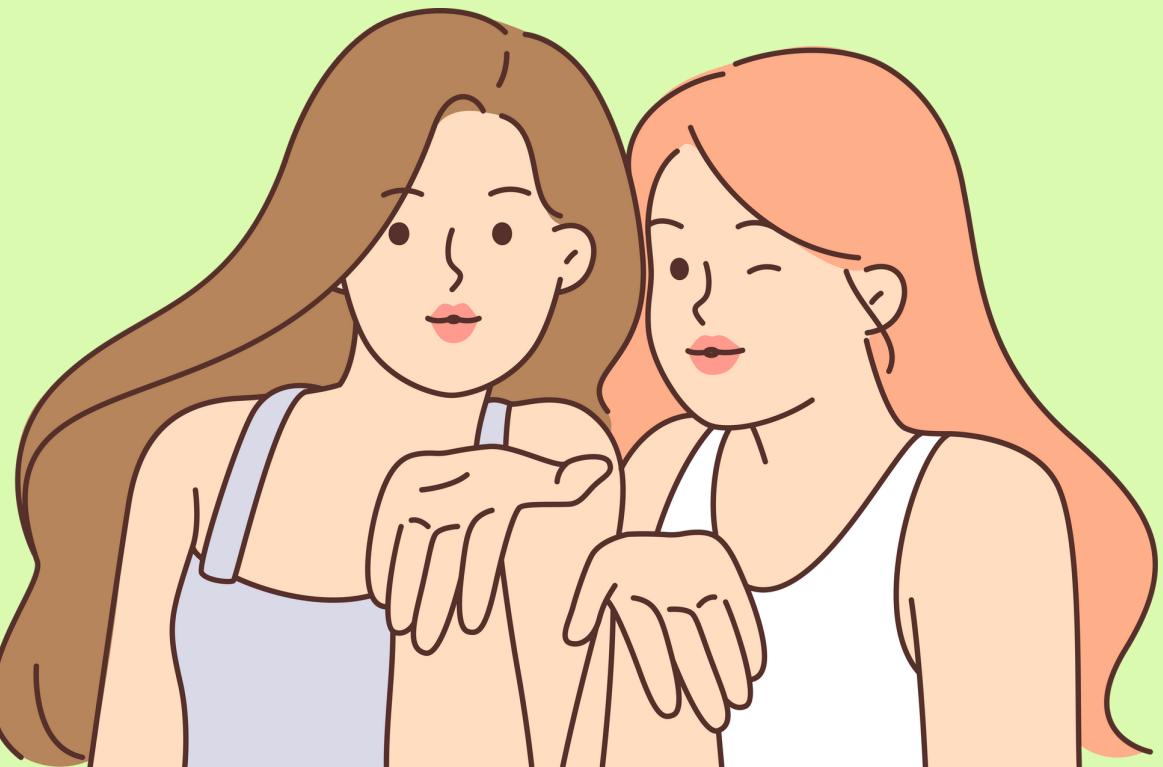
```
else:  
volte para casa sem comprar nada
```



Estrutura Condicional: EXEMPLO 2

Imagine a situação:

Juliana e Fernanda são irmãs. Nossso programa deve nos dizer qual das duas é mais velha. Como poderíamos fazer isso no Python?



Estrutura Condicional: EXEMPLO 2

```
idade_Juliana = int(input("Digite a idade de Juliana: "))
```

```
idade_Fernanda = int(input("Digite a idade de Fernanda: "))
```

```
if (idade_Juliana > idade_Fernanda):
```

```
    print("Juliana é mais velha do que Fernanda!")
```

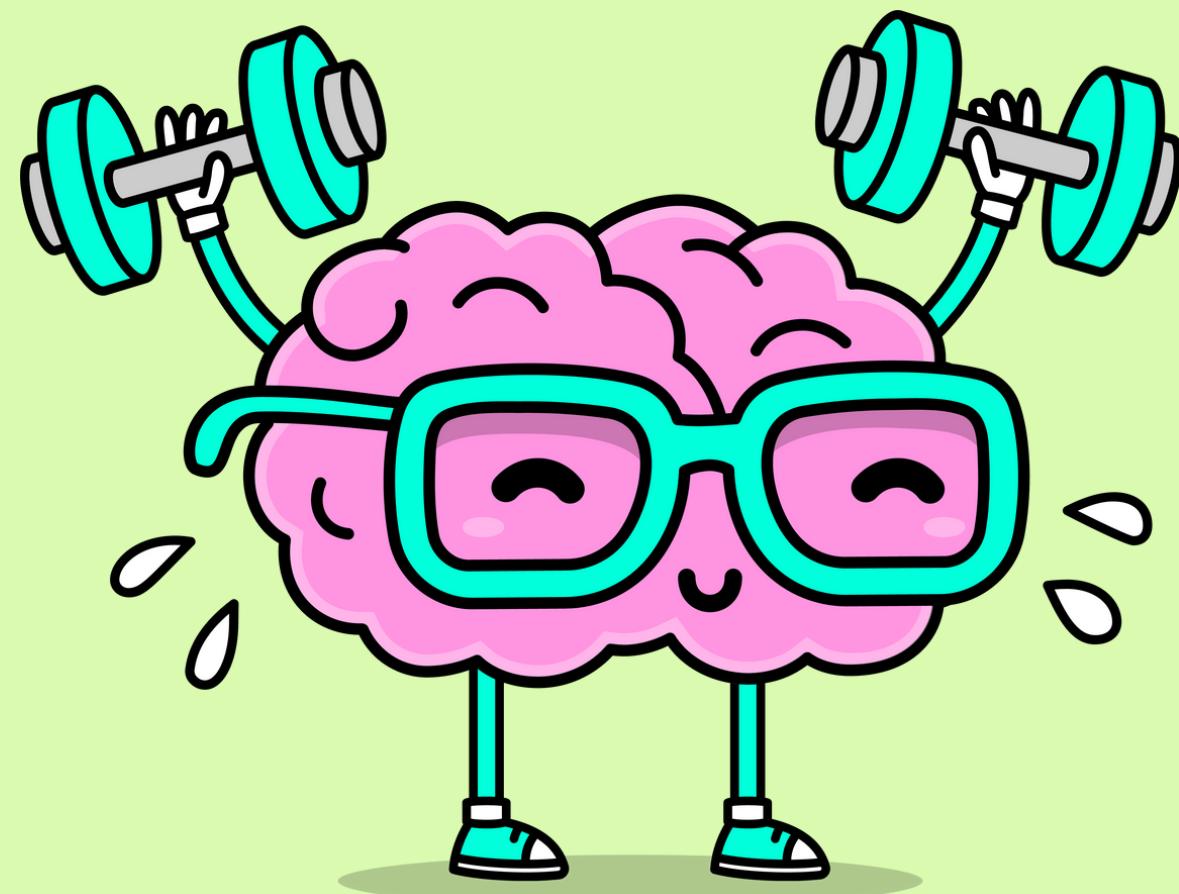
```
else:
```

```
    print("Fernanda é mais velha do que Juliana!")
```

DÚVIDAS?



VAMOS EXERCITAR!



Faça um programa que:

Encontre o **dobro** de um número **caso** ele seja **positivo** e o seu **triplo caso** seja **negativo**, imprima o resultado em ambos os casos.

Possível resolução para esse exercício:

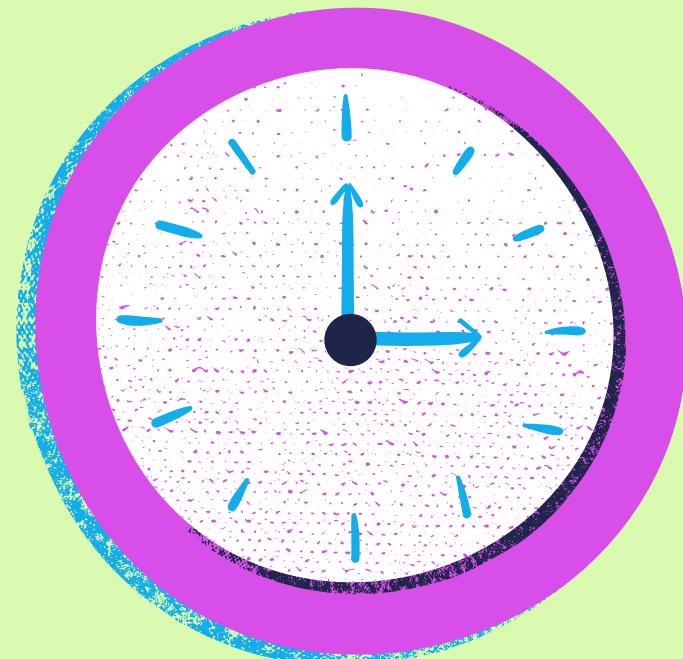
Encontre o **dobro** de um número **caso** ele seja **positivo** e o seu **triplo caso** seja **negativo**, imprima o resultado em ambos os casos.

```
numero = float(input("Digite um número qualquer: "))

if (numero > 0):
    print(2 * numero)
else:
    print(3 * numero)
```

Estruturas de Repetição

- As estruturas de repetição permitem que **um mesmo bloco de comando** seja executado **diversas vezes**.

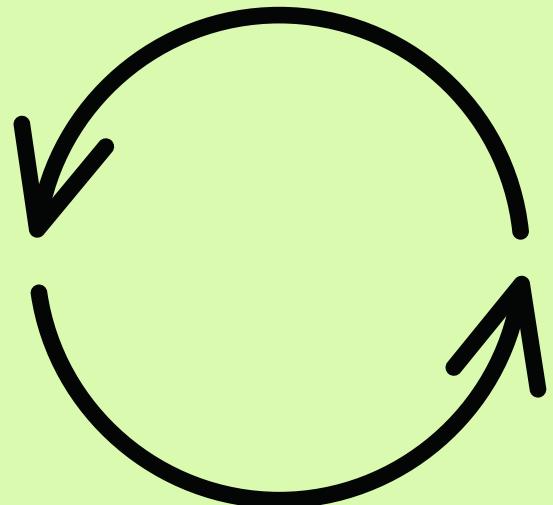


Estruturas de Repetição

- Existem **dois tipos** de estrutura de repetição:
 1. **WHILE** (do inglês, “enquanto”: **repetição condicional**);
 2. **FOR** (do inglês, “por”: **repetição contável**).

Repetição Condicional -

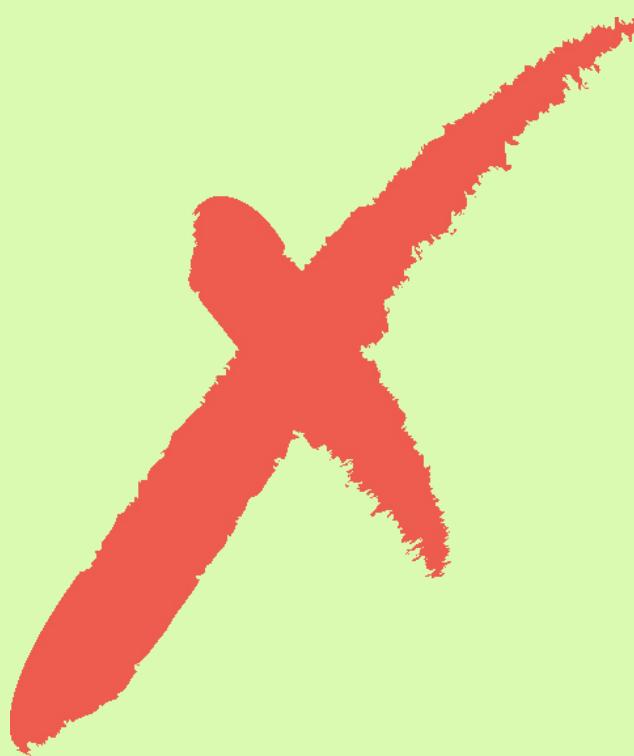
WHILE



- Ela executa um **bloco de comandos** enquanto uma condição for **verdadeira**.
- Quando a condição passar a ser **falsa**, a instrução que vier após o bloco do while será executada.
- Se a condição for sempre verdadeira, o loop irá acontecer **infinitamente**.

Repetição Condicional - WHILE

- Se a condição for **falsa desde o início**, o bloco de instruções **nunca** será executado.



Repetição Condicional - WHILE

VAMOS VER UM EXEMPLO?

Imagine que:

Você esteja preparando uma jarra de suco para o almoço e deseja adoçá-lo.

Você adiciona uma colherada de açúcar e prova o suco, mas ele continua pouco adoçado.

Então, você vai adicionando mais colheradas e provando **até que ele esteja da forma que deseja**.



Se isso fosse um código, poderíamos escrevê-lo da seguinte forma:

while suco não estiver adoçado:

adicone mais uma colher de açúcar.



Observe que:

Você esteja preparando uma jarra de suco para o almoço e deseja adoçá-lo. Você adiciona uma colherada de açúcar e prova o suco, mas ele continua pouco adoçado. Então, você vai adicionando mais colheradas e provando até que ele esteja da forma que deseja.

Observe que:

Nesse caso, o **critério de parada do loop** foi: o **suco estar na forma que você deseja!**

Sendo assim, **quando o suco estiver adoçado o suficiente, a condição para que o loop ocorra**, que é “**o suco não está adoçado**”, será **falsa**.

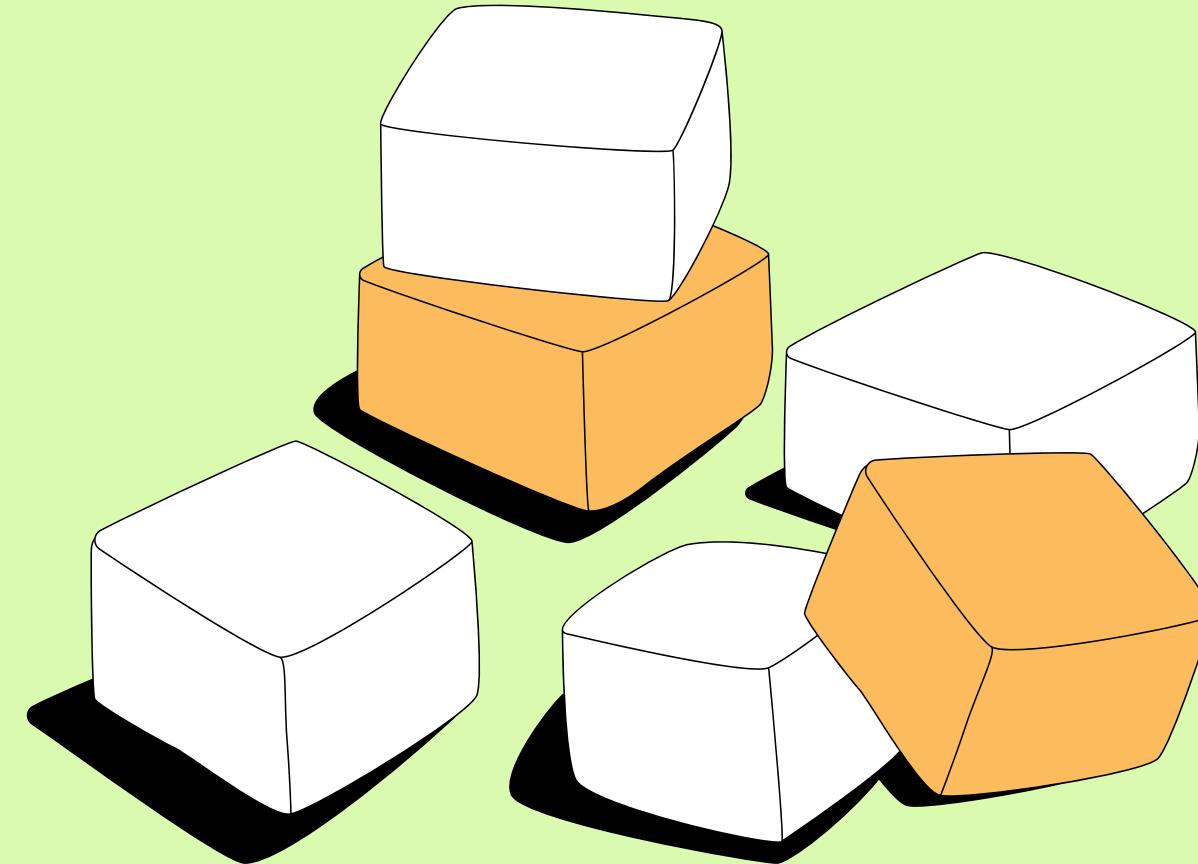


while suco não estiver adoçado:
adicone mais uma colher de açúcar.



Observe que:

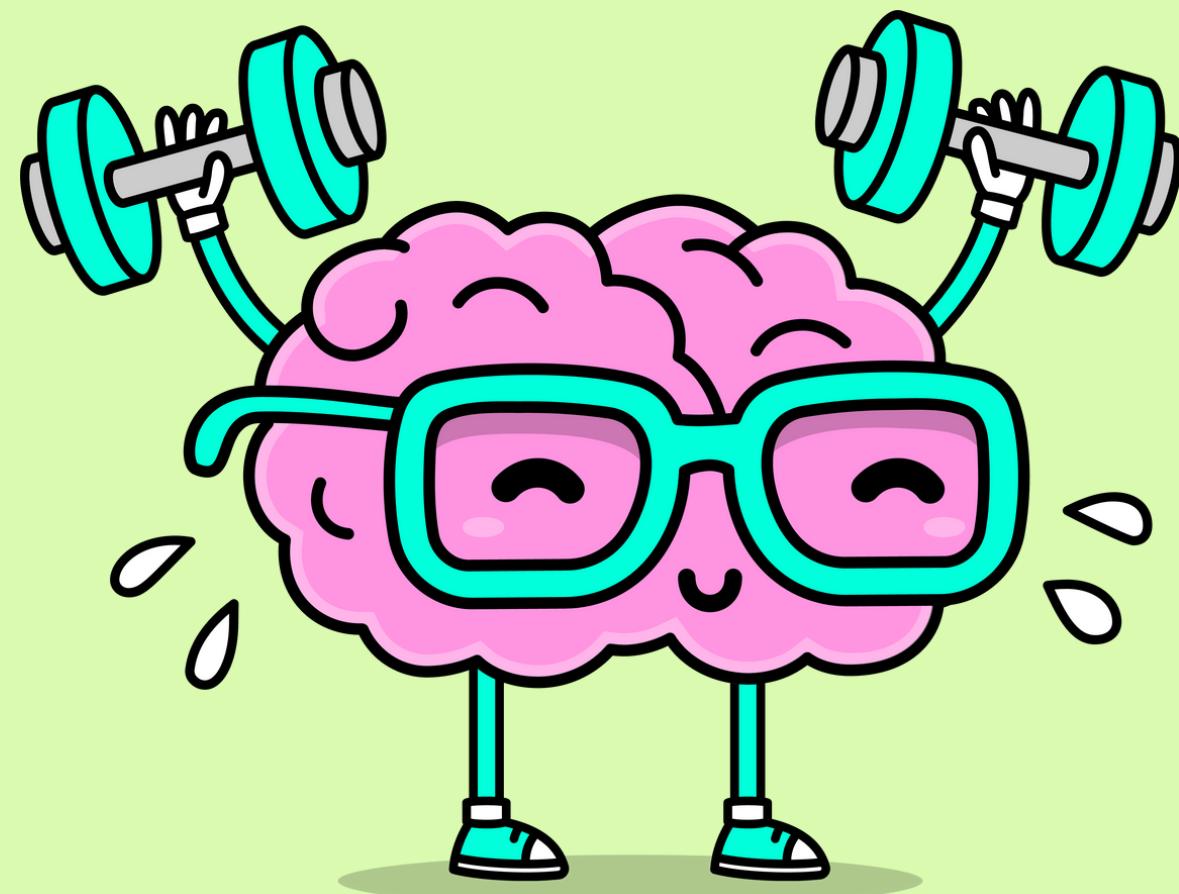
Dessa forma, o bloco de comando dentro do while vai parar de executar quando o suco estiver adoçado.



DÚVIDAS?



VAMOS EXERCITAR!



Então, vamos lá!

Vamos fazer um programa que:

- **Conte de 0 até 10.**

8

9

10

**Ou seja, o nosso programa fará com que o computador printe
(escreva) as seguintes mensagens:**

- 0
- 1
- 2
- 3
- 4
- ...
- 10

Possível resolução pro exercício abaixo

Vamos fazer um programa que:
Conte de 0 até 10.

A = 0

while (A < 11):

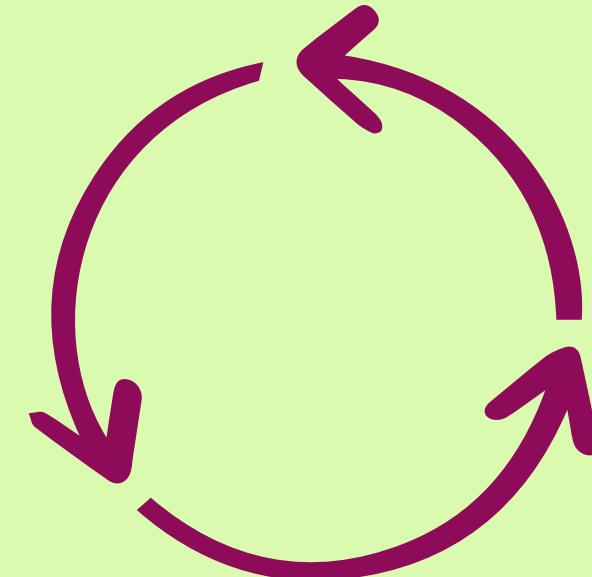
print(A)

A = A + 1

Loops Infinitos

Ao programar, nós devemos tomar cuidado para não acabar gerando loops infinitos, pois esses loops resultam em um travamento no computador.

Por isso, o ideal é que todo while tenha uma condição de parada.



Loops Infinitos - EXEMPLO

idade = 1

while idade > 0:

print(idade)

idade = idade + 1

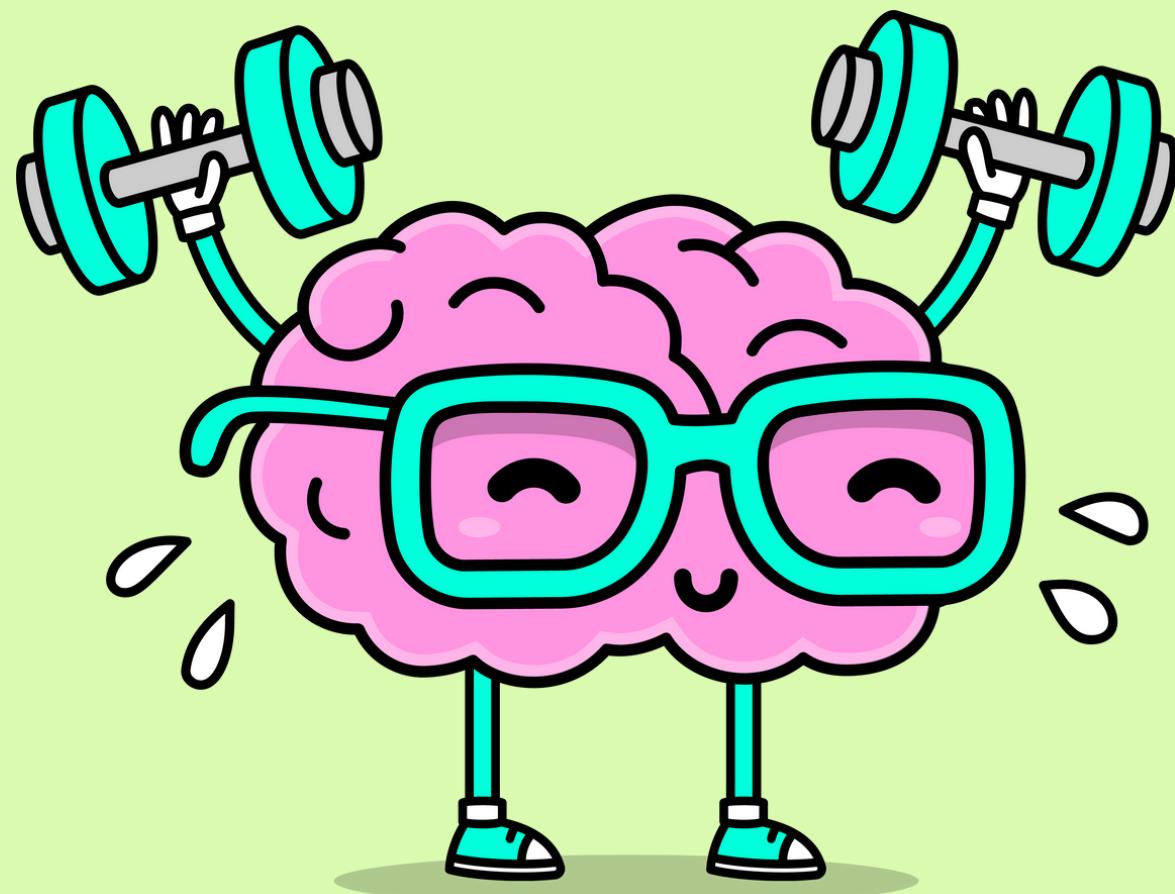
print("A idade da criança é ", idade)

O que irá acontecer?

DÚVIDAS?



VAMOS EXERCITAR!



Então, vamos lá!

Faça um programa que:

- 1. Receba** dois valores;
- 2. Escreva** todos os números que existem entre o **primeiro valor** e o **segundo**.

Vamos ver um exemplo?

- Após o usuário definir duas variáveis inteiros, como: **A = 1** e **B = 5**;
- O computador deverá imprimir: **2**
 3
 4



Então, vamos lá!

Faça um programa que:

- 1. Receba** dois valores;
- 2. Escreva** todos os números que existem entre o **primeiro valor** e o **segundo**.

Observação:

- Teste o que acontece quando:
 1. Os números são **iguais**;
 2. O **primeiro** número é **maior** que o **segundo**.
- Tente pensar em uma forma de solucionar isso com o que você já aprendeu!

8

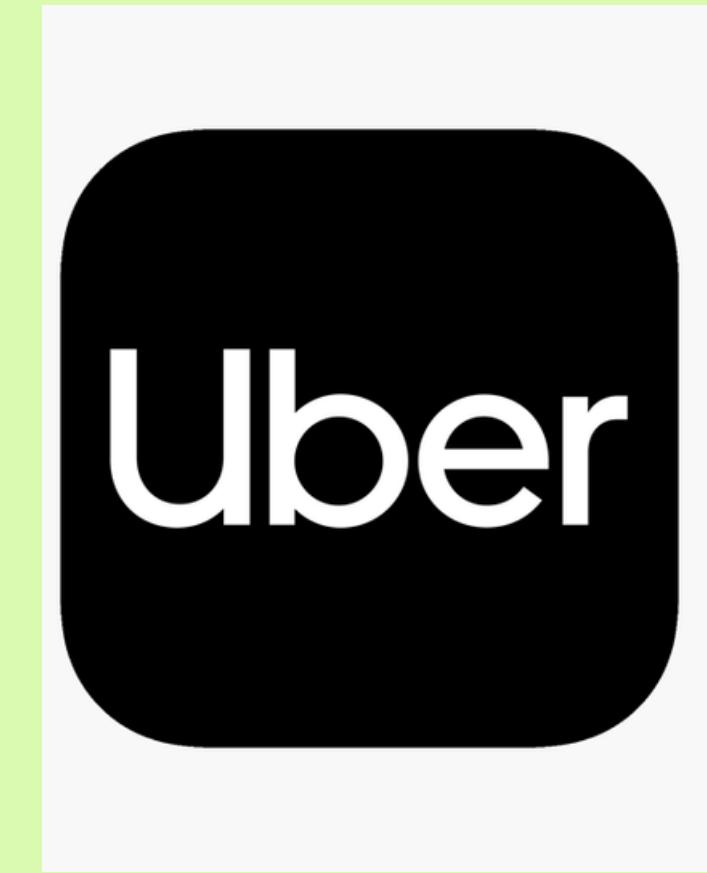
9

10

DÚVIDAS?

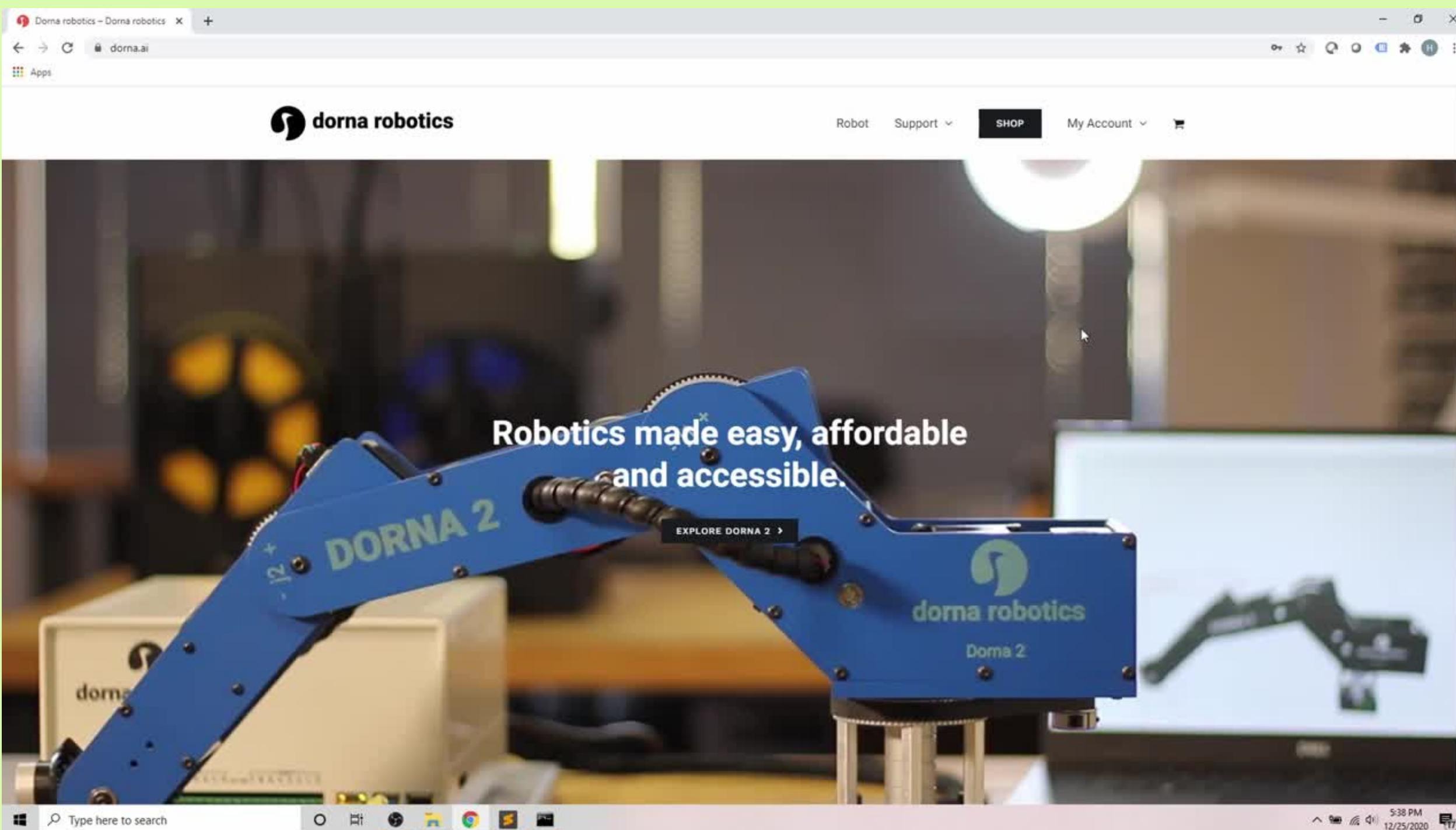


ONDE O PYTHON É USADO?



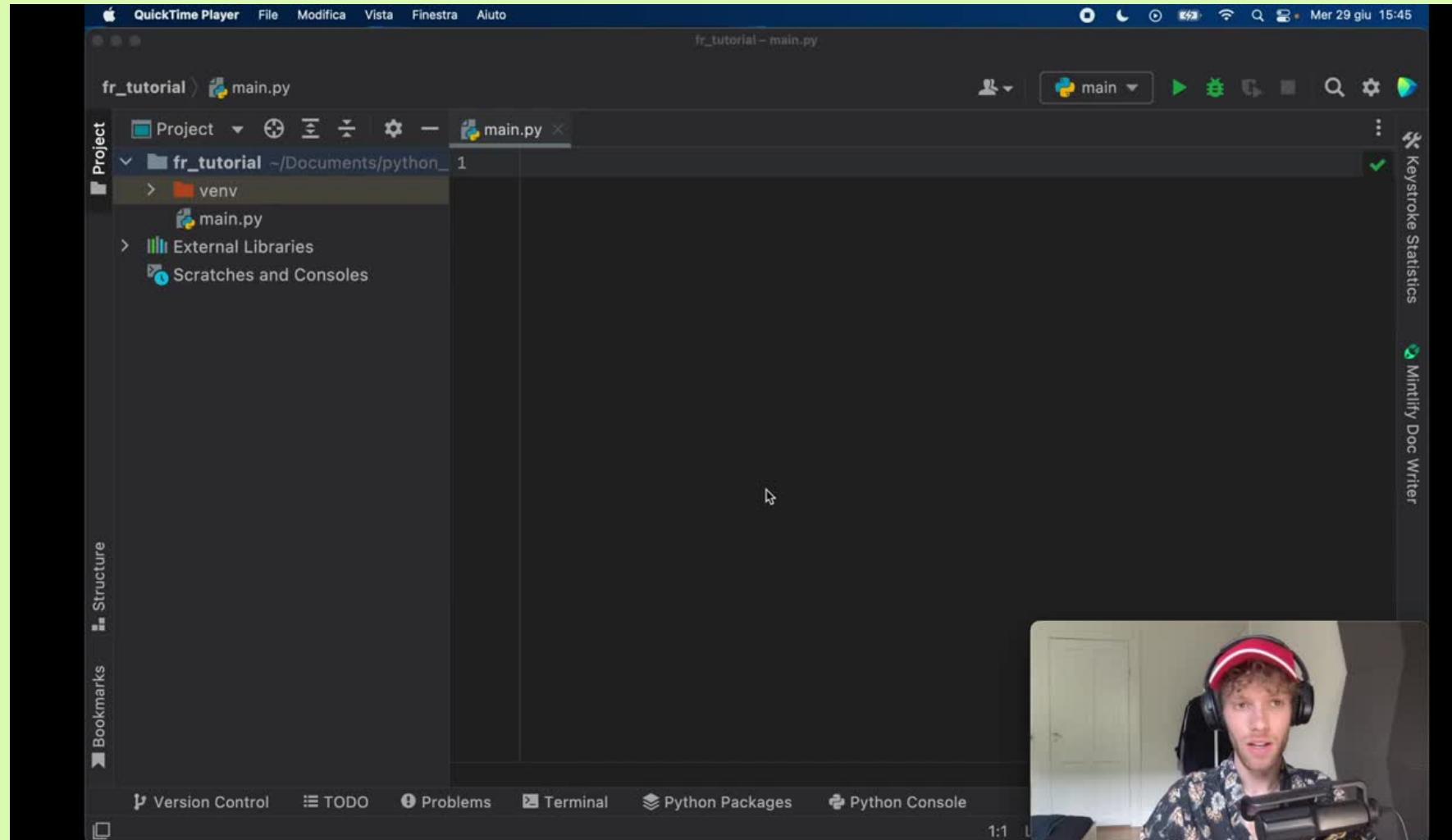
ONDE O PYTHON PODE SER USADO?

Na robótica...



ONDE O PYTHON PODE SER USADO?

Reconhecimento facial...

A screenshot of the PyCharm IDE showing the 'main.py' file. The code implements a function 'face_confidence' that calculates a confidence score based on face distance and a threshold. It then checks if the face distance is greater than the threshold to determine if a match is found. Below this, another part of the code initializes a video capture object and enters a loop to process frames. A video preview window in the bottom right corner shows the developer's face.A screenshot of the PyCharm IDE showing the continuation of the 'main.py' file. The code defines a 'run_recognition' method that initializes a video capture object and processes frames in a loop. It prints known face names and handles frame processing. A video preview window in the bottom right corner shows the developer's face.

CONSIDERAÇÕES FINAIS



Parabenizamos e agradecemos muito pela sua participação e esperamos que tenha gostado das nossas oficinas! Não esqueça de entrar no Classroom da turma para acessar os materiais e de responder os formulários.

Nos siga nas redes sociais!

- Instagram: **@includemeninas**
- Facebook: **@include.meninas.uff**
- Youtube: **Include Meninas UFF**