





# FIAP

## Aula 5

---

COMPUTATIONAL THINKING USING PYTHON



# AGENDA



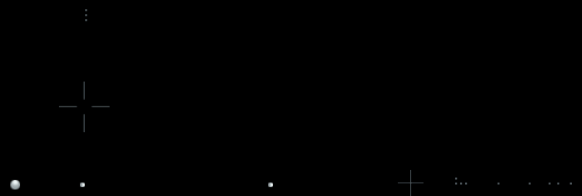
1. Estruturas de Repetição

2. Laço while

3. Variáveis contadoras, acumuladoras e de sinalização booleano

4. Conceituar estrutura de repetição "infinita"

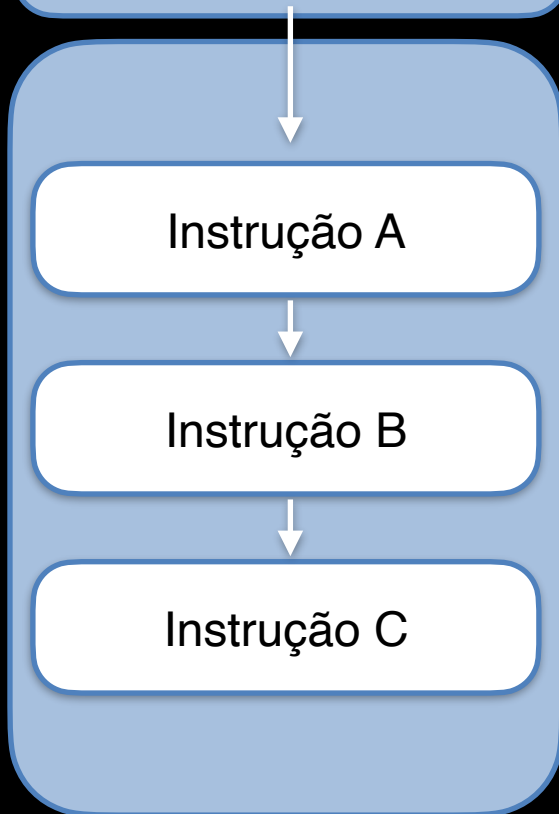
5. Combinar estruturas de repetições com estruturas de seleção



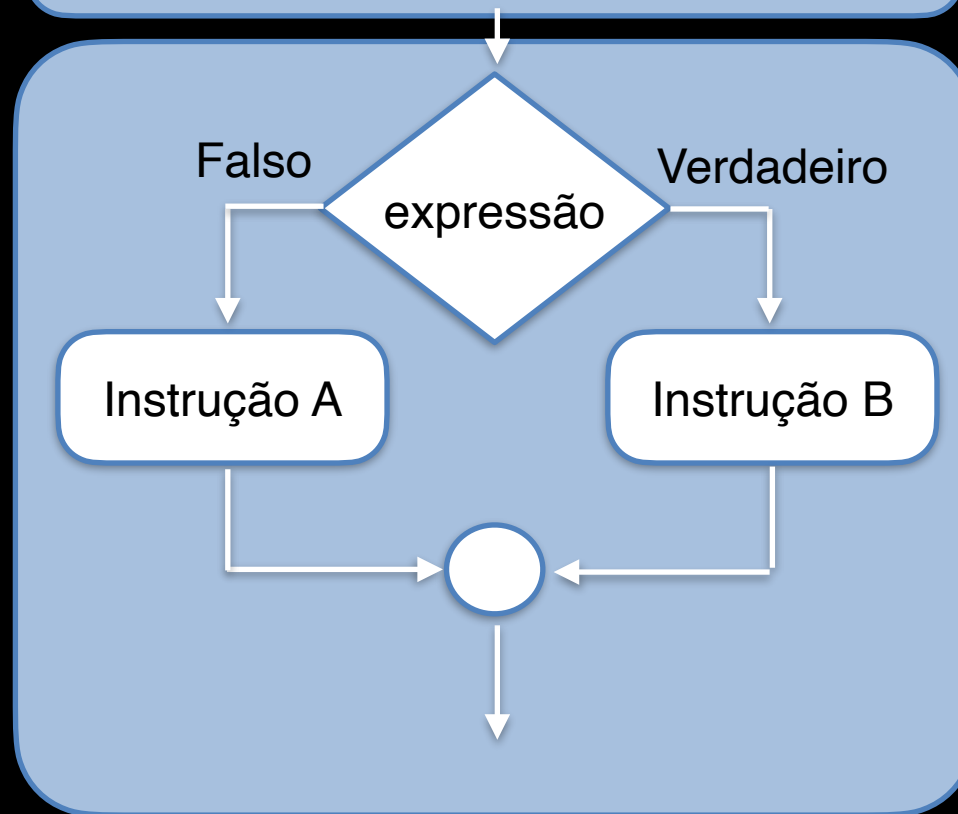
Também conhecidas como estruturas de iteração, malhas de repetição, laços e *loops*

# Representações

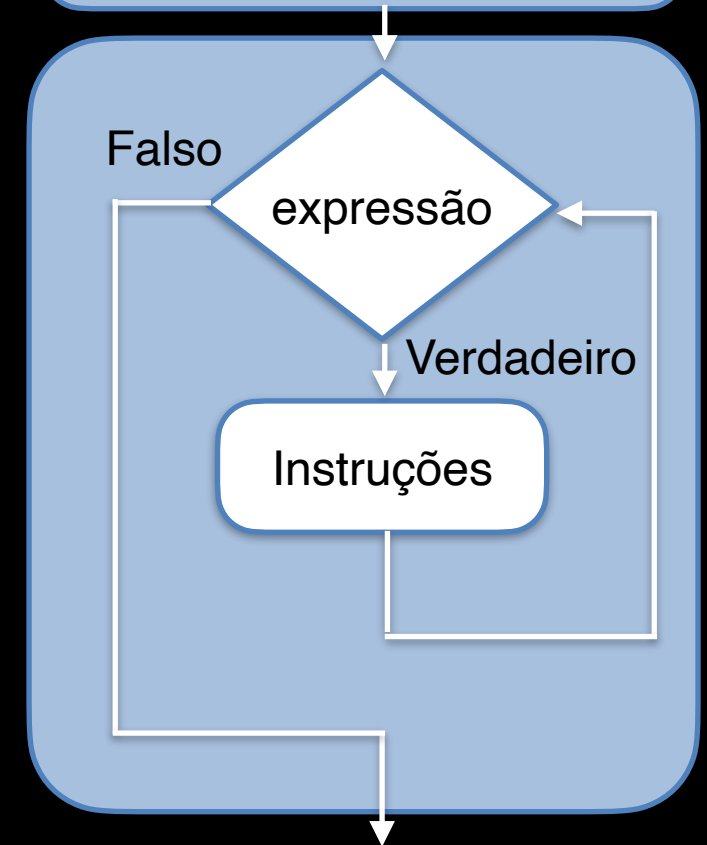
Estrutura Sequencial



Estrutura de Seleção



Estrutura de Repetição







# Estruturas de Repetição em Linguagem Python



Estruturas de repetição são aplicadas quando precisamos criar programas que demandam a **repetição** de uma instrução ou uma sequência de instruções

```
print( 'Computational Thinking using Python' )
```

[illegible]



# Exibição dos cinco primeiros números naturais

```
print (1)  
print (2)  
print (3)  
print (4)  
print (5)
```

## Alternativa para exibição dos cinco primeiros números naturais

```
n = int(input('Número: '))  
print(n + 1)  
print(n + 2)  
print(n + 3)  
print(n + 4)  
print(n + 5)
```

## Alternativa para exibição dos cinco primeiros números naturais

```
n = int(input('Número: '))
```

```
n += 1  
print(n)  
n += 1  
print(n)  
n += 1  
print(n)  
n += 1  
print(n)
```



while

for

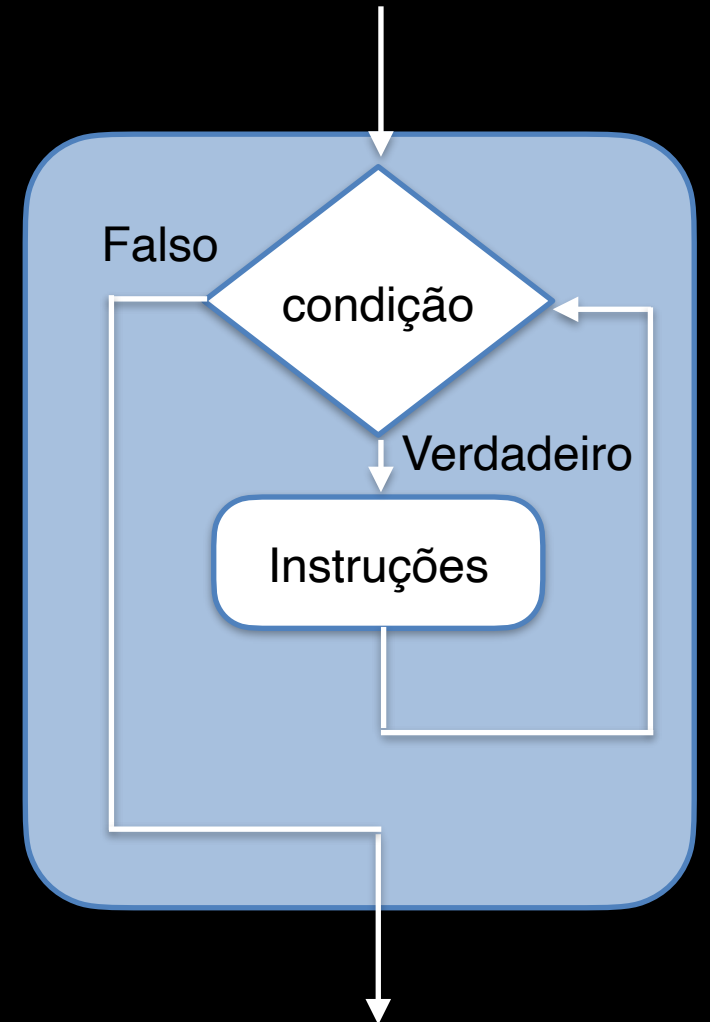
# Tipos de estruturas de repetição

- Quantidade de repetições **indefinidas (while)**
  - não é possível determinar quantas vezes as instruções irão se repetir antes do início do *loop*
- Quantidade de repetições **definidas (for)**
  - é possível determinar o número de vezes em que as instruções irão se repetir antes do início do *loop*

# Estrutura de repetição *while*

# Estrutura de repetição while

```
while <condição>:  
    <bloco de código>
```





# Estrutura de repetição while

```
x = 1
while x <= 5:
    print(x)
    x += 1

print('Valor de x: ', x)
```

# Estrutura de repetição while

```
x = int(input('Valor: '))  
cont = 0  
while cont < 5:  
    x += 1  
    print('Cont: ', cont)  
    cont += 1
```

# Estrutura de repetição while

```
inicio = int(input('Inicio: '))  
fim = int(input('Fim: '))  
x = inicio  
while x <= fim:  
    print(x)  
    x += 1
```



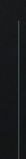
Teste no Python Tutor os  
exemplos anteriores





# Exercícios

+





# Exercícios

1. Crie um programa que exiba todos os números inteiros de 100 à 200 em ordem decrescente.
2. Crie um programa que leia um número natural  $n$  dado pelo usuário e exiba só os  $n$  primeiros pares a partir de 0. Por exemplo, se  $n=6$  ser exibido 0 2 4 6 8 10
3. Escreva um programa que leia um número, calcule e imprima a tabuada deste número (de 1 à 10)

# Exercícios

4. Crie um programa que solicite ao usuário dois números naturais  $x$  e  $y$ , o programa deverá exibir o quociente da divisão inteira de  $x$  por  $y$  sem usar os operadores de divisão e multiplicação. Por exemplo, se  $x=7$  e  $y=2$  a resposta será 3, pois podemos raciocinar que o quociente da divisão inteira de  $x$  por  $y$  é dado pela quantidade de vezes que  $y$  pode ser subtraído de  $x$  sem que  $x$  se torne negativo.





# Variável contadora

**Variável contadora** é utilizada para contar quantas vezes o bloco de instruções do laço foi executado e/ou **controlar** quantas vezes ele será executado

# Variável contadora

```
executa = input('Executar o bloco: ')
contador = 0
while executa == 'sim':
    contador += 1
    executa = input('Executar o bloco: ')

print(f'O bloco foi executado {contador} vezes')
```

# Variável contadora

```
contador = 10
while contador > 0:
    print(contador)
    contador -= 1
print('Fogo!')
```

# Exercício

Crie um programa que peça letras como entrada, uma por vez, até que seja lida a letra 'x', ao final, o programa deve exibir a quantidade de letras lidas sem contabilizar 'x'.

**Lembre-se:** Python diferencia letras maiúsculas de minúsculas.





# Variável acumuladora

**Variável acumuladora** é utilizada para **acumular** valores que, em geral, não são constantes, ou seja, o incremento ou decremento pode ser variável.



# Exercício

Crie um programa que receba como entrada os preços de itens comprados em um supermercado por um cliente, ao final, o programa deverá exibir o total da compra. Para informar que não há mais itens a serem comprados, o cliente deve digitar o valor -1.

# Solução válida

```
total = 0
preco = float(input('Preço do item: '))
while preco != -1:
    total += preco
    preco = float(input('Preço do item: '))

print(f'Total da compra: R$ {total:.2f}')
```

# Exercício

Crie um programa que receba como entrada o crédito e depois o preço de itens comprados por esse cliente. O programa deverá parar de solicitar novos preços quando o crédito disponível for insuficiente para pagar por um deles. Ao final exiba o total da compra e o crédito restante.

# Solução válida

```
credito = float(input('Seu crédito: '))
total = 0 #variável acumuladora
preco = float(input('Preço do item: '))
while credito >= preco:
    total += preco
    credito -= preco
    preco = float(input('Preço do item: '))
print(f'Total da compra: R$ {total:.2f}')
print(f'Crédito restante: R$ {credito:.2f}')
```



# Variável *flag* booleana

***Variável *flag**** sinaliza com um valor booleano se o laço deve ou não ser encerrado

# Exemplo

```
total = 0
quero_comprar = True
while quero_comprar:
    preco = float(input('Preço: '))
    total += preco
    opcao = input('Continuar comprando (s/n)? ')
    if opcao != 's':
        quero_comprar = False

print(f'Total da compra: R$ {total:.2f}')
```





# Laço infinito



**Laço infinito** é uma estrutura de repetição em que a condição associada ao laço é sempre **verdadeira**

# Exemplos de laço infinito

```
n = 0
```

```
while n <= 10:  
    print(n)
```

```
print('Fora do while')
```

```
n = 0
```

```
while n >= 0:  
    print(n)  
    n += 2
```

```
print('Fora do while')
```



# Combinação de estruturas de controle de fluxo



# Exemplo

```
n = 10
cont = 1
while cont <= 10:
    if (cont % 2 == 0):
        print(f'{cont} é par')
    else:
        print(f'{cont} é ímpar')
    cont+=1
```



Obrigado e até a próxima aula!