

SSC0800 - Introdução à Ciência de Computação I

# Operadores de Repetição

Prof.: Leonardo Tórtoro Pereira

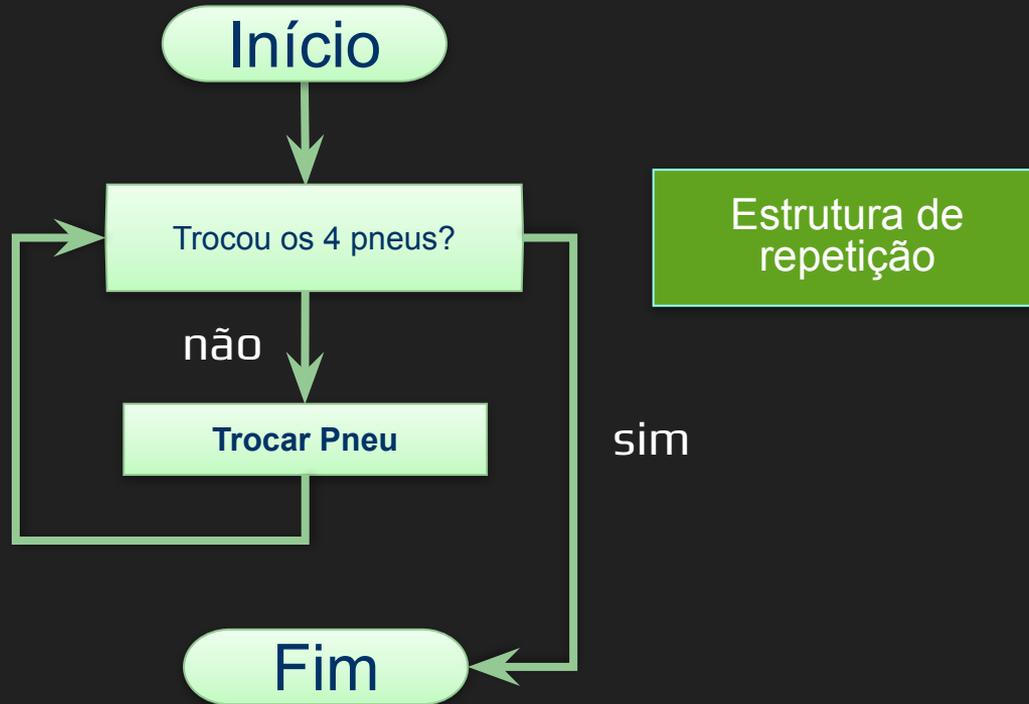
[leonardop@usp.br](mailto:leonardop@usp.br)

Baseado no material dos profs Fernando S. Osório e Claudio F.M. Toledo

Na aula passada...



# Algoritmo para trocar um pneu



# Loops

## Entry Controlled

for

```
for( initialization ; condition; updation)
{
}
```

while

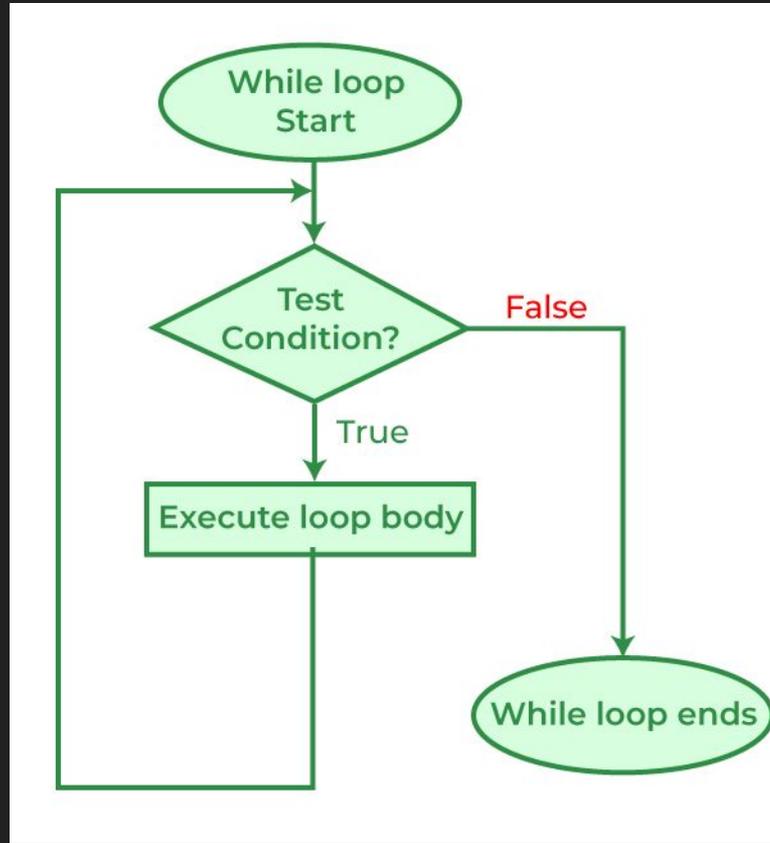
```
while( condition )
{
}
```

## Exit Controlled

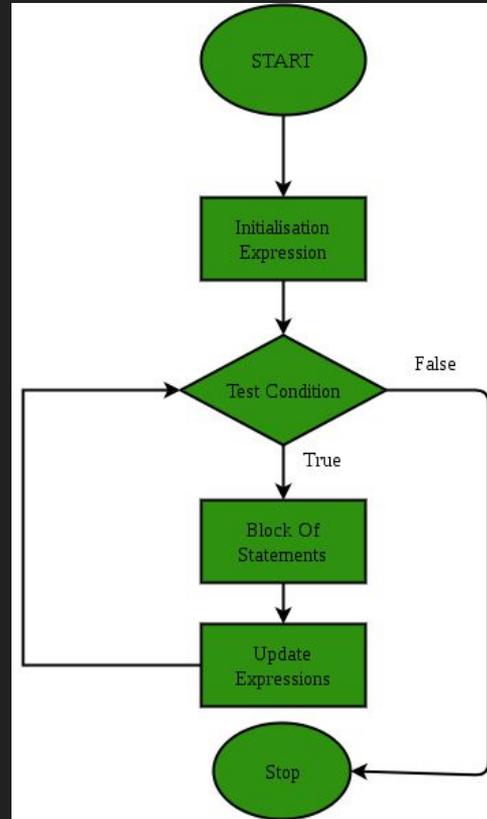
do-while

```
do
{
}while( condition )
```





Fonte: <https://www.geeksforgeeks.org/cpp-loops/>



Fonte: <https://www.geeksforgeeks.org/loops-in-c-and-cpp/>

# Para Refrescar

<https://www.hackerrank.com/challenges/python-loops/problem>

O que vamos aprender hoje?



# Objetivos

- Aprender a aninhar fluxos de repetição em Python
  - ◆ Fluxos de repetição com listas e strings
- Entender como loop funciona por baixo dos panos em Python
- Introduzir o operador Else em loops
- Conhecer o Do While em outras linguagens (C)

# Tópicos da Aula

- Aninhamento de repetições
- For com listas
- Implementação do for em listas no Python
- Else
- Do While (em C)

# Aninhamento

# Aninhamento

- Assim como nos condicionais, os operadores de repetição podem ser aninhados
- Loop interno vai ocorrer completamente a cada iteração do loop externo
- Cuidado com a complexidade!

## *for aninhado*

```
for i in range(3):  
    for j in range(3):  
        print(i, j)  
        print("-----")
```

# While aninhado

```
i = 0
while ( i < 3):
    j = i+1
    while(j > 0):
        print(i, j)
        j-=1
    i+=1
```

# Exemplos

# Loops em Strings e Listas

## *For em lista*

- Segunda opção principal para iterar sobre elementos
  - ◆ "x in list"
    - Itera sobre os elementos de uma lista
    - Não tem um contador associado a ele
  - ◆ Lista pode ser uma string

## *For em lista*

```
numbers = [1, 5, 8, 20, 30, 132]
for x in numbers:
    print(x)
```

```
numbers = [1, 5, 8, 20, 30, 132]
for x in numbers:
    if(not x%2):
        print(x)
```

## *For em lista*

```
numbers = [1, 5, 8, 20, 30, 132]
for i in range(len(numbers)):
    print(i, numbers[i])
```

```
for letter in "python forever":
    if ord(letter) < 110:
        print(letter)
```

## Cuidado com coleções

- Listas (e outras coleções) podem dar problemas ao serem modificadas enquanto a iteração é realizada
- Normalmente, opera-se sobre uma cópia da coleção

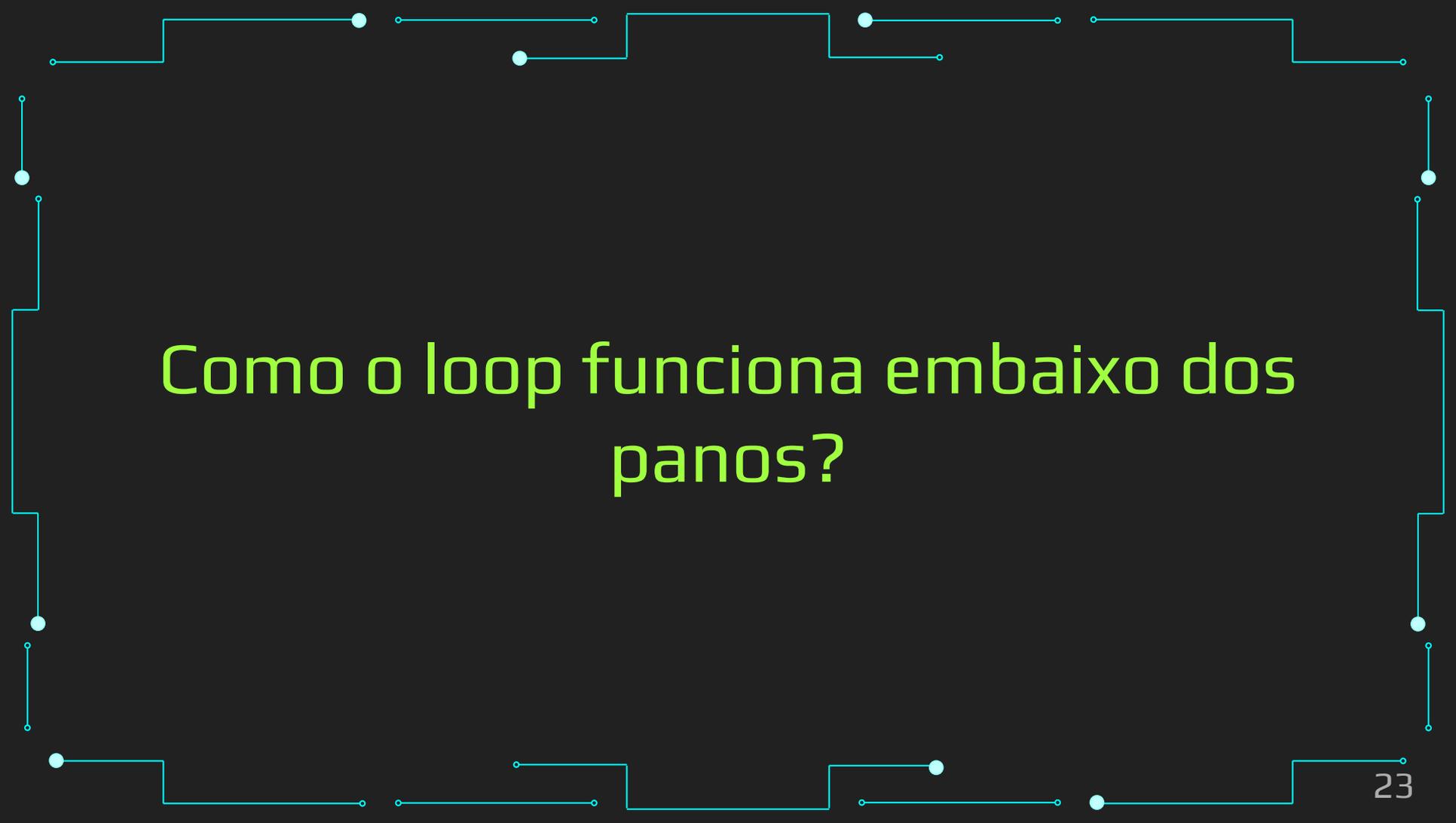
## *For em lista*

```
users = {'Hans': 'active', 'Éléonore': 'inactive', '景太郎':  
        'active'}
```

```
for user, status in users.copy().items():  
    if status == 'inactive':  
        del users[user]
```

```
active_users = {}  
for user, status in users.items():  
    if status == 'active':  
        active_users[user] = status
```

# Exemplos



Como o loop funciona embaixo dos  
panos?

## O loop por trás do loop

- Transforma a lista num “objeto” iterable
- Roda um loop infinito buscando uma exceção
- Enquanto rodar
  - ◆ Pega o “next”
  - ◆ Realiza a operação
- Para quando next levanta uma exceção

# O loop por trás do loop

```
fruits = ["apple", "orange", "kiwi"]
```

```
iter_obj = iter(fruits)
```

```
while True:
```

```
    try:
```

```
        fruit = next(iter_obj)
```

```
        print(fruit)
```

```
    except StopIteration:
```

```
        break
```

# Exemplos



Else nos loops

## Else

- Executado ao “final” dos loops
  - ◆ For: após a última iteração
  - ◆ While: quando condição vira falsa
- Não é executada se o loop acaba com *break*
- Usada para uma ação quando o *break* não ocorrer
  - ◆ “Fluxo normal” do loop

## *Else*

```
for n in range(2, 10):  
    for x in range(2, n):  
        if n % x == 0:  
            print(n, 'equals', x, '*', n//x)  
            break  
    else:  
        # loop fell through without finding a factor  
        print(n, 'is a prime number')
```

# Exemplo

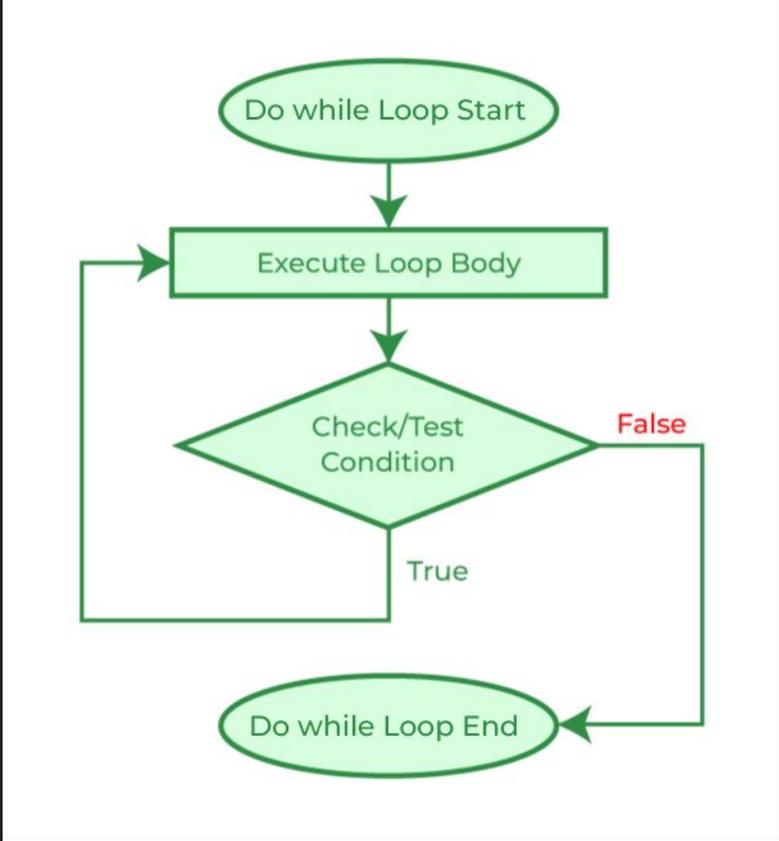
# Do While (não em Python)

## *Do While*

- Igual ao *while*, mas a condição é testada depois do corpo do laço
- Ou seja, o corpo é executado pelo menos uma vez

# *Do While*

```
int main () {
    int i, j;
    i = 0;
    j = 10;
    do {
        //Fazer algo
        printf ("%d %d\n", i, j);
        ++i;
        j-=i;
    } while ( i < 3 && j > 0);
    return 0;
}
```



Fonte: <https://www.geeksforgeeks.org/loops-in-c-and-cpp/>

# Referências

# Referências

1. <https://www.learnpython.org/>
2. <https://www.w3schools.com/python/>
3. <https://panda.ime.usp.br/cc110/static/cc110/index.html>
4. [https://www.youtube.com/playlist?list=PLcoJJSvnDgcKpOi\\_UeneTNTIVOigRQwcn](https://www.youtube.com/playlist?list=PLcoJJSvnDgcKpOi_UeneTNTIVOigRQwcn)
5. <https://docs.python.org/3/tutorial/introduction.html>
6. <https://docs.python.org/3/tutorial/controlflow.html>
- 7.