SSC0304 - Introdução à Programação para Engenharias

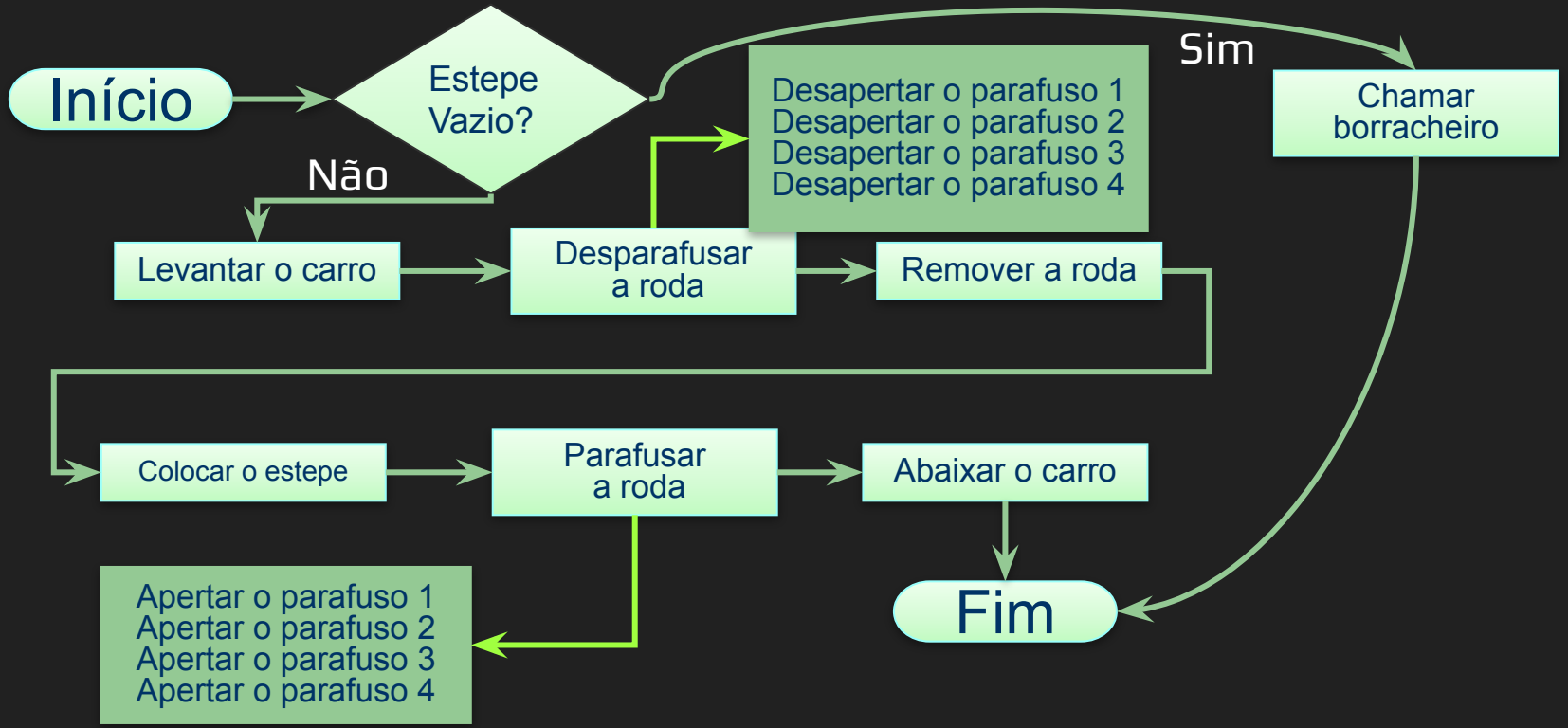
Operadores de Repetição

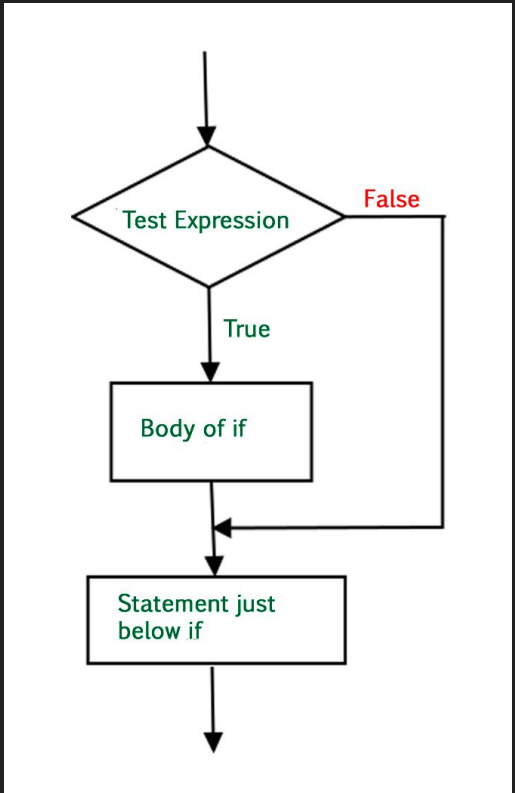
Prof.: Leonardo Tórtoro Pereira
leonardop@usp.br

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

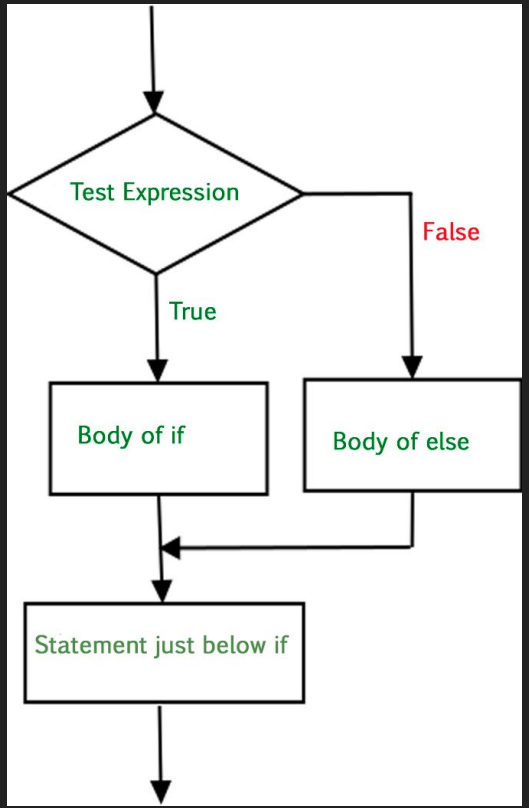
Na aula passada...

Algoritmo para trocar um pneu - Relembrando

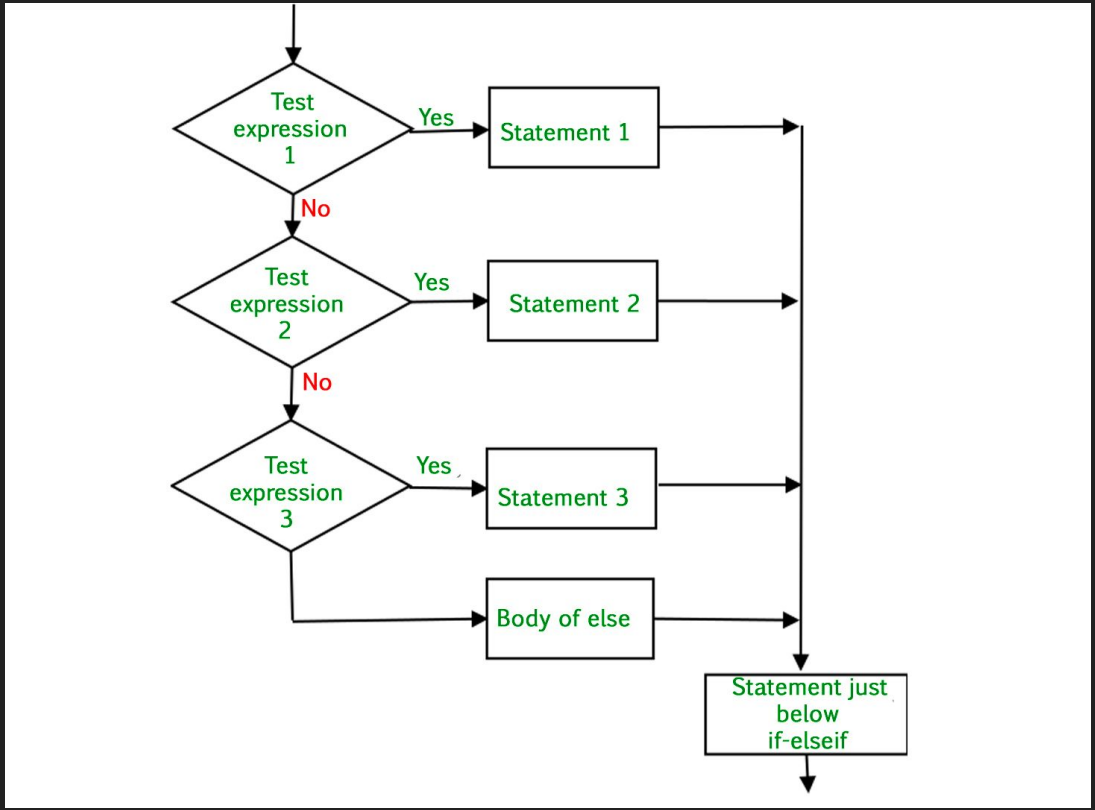




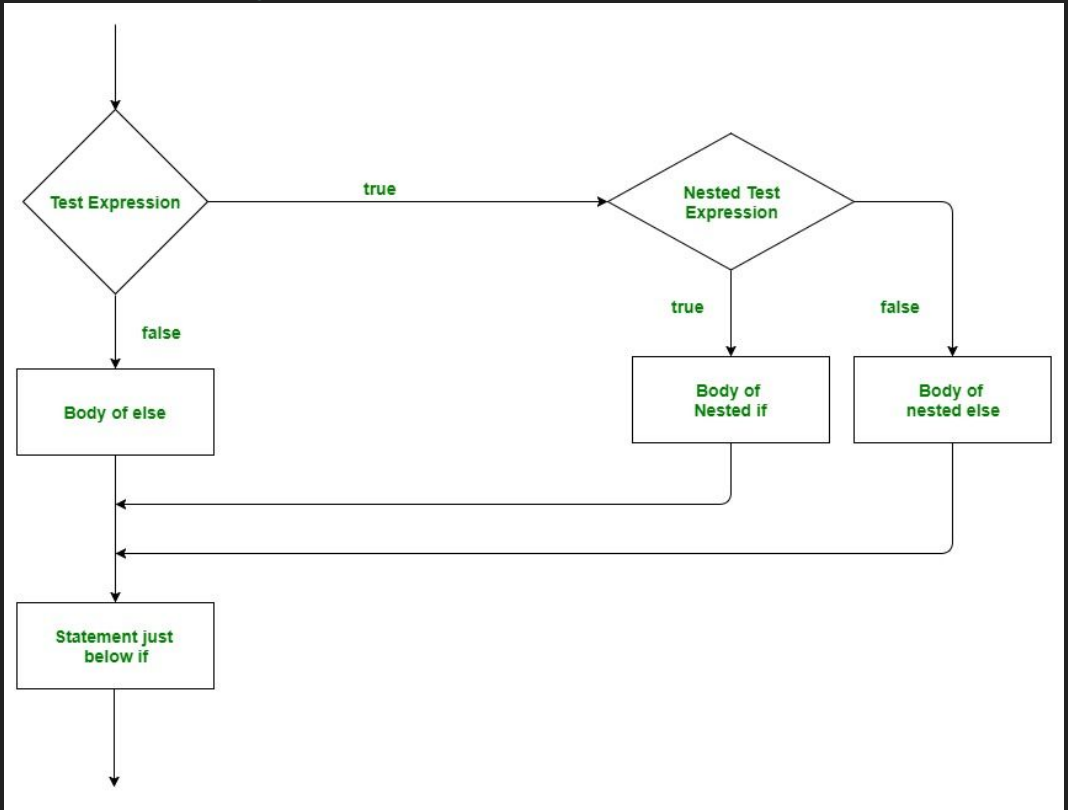
Fluxograma de uma estrutura condicional. Fonte [5]



Fluxograma do If-Else. Fonte [5]



Fluxograma do If-Elif-Else. Fonte [5]



Fluxograma de aninhamentos. Fonte [5]

```php
function register()
{
    if (!empty($_POST)) {
        $msg = '';
        if ($_POST['user_name']) {
            if ($_POST['user_password_new']) {
                if ($_POST['user_password_new'] === $_POST['user_password_repeat']) {
                    if (strlen($_POST['user_password_new']) > 5) {
                        if (strlen($_POST['user_name']) < 65 && strlen($_POST['user_name']) > 1) {
                            if (preg_match('/^[a-z\d]{2,64}$/i', $_POST['user_name'])) {
                                $user = read_user($_POST['user_name']);
                                if (!isset($user['user_name'])) {
                                    if ($_POST['user_email']) {
                                        if (strlen($_POST['user_email']) < 65) {
                                            if (filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)) {
                                                create_user();
                                                $_SESSION['msg'] = 'You are now registered so please login';
                                                header('Location: ' . $_SERVER['PHP_SELF']);
                                                exit();
                                            } else $msg = 'You must provide a valid email address';
                                        } else $msg = 'Email must be less than 64 characters';
                                    } else $msg = 'Email cannot be empty';
                                } else $msg = 'Username already exists';
                            } else $msg = 'Username must be only a-z, A-Z, 0-9';
                        } else $msg = 'Username must be between 2 and 64 characters';
                    } else $msg = 'Password must be at least 6 characters';
                } else $msg = 'Passwords do not match';
            } else $msg = 'Empty Password';
        } else $msg = 'Empty Username';
        $_SESSION['msg'] = $msg;
    }
    return register_form();
}
```

Evite o if Hadouken!

Para Refrescar

https://www.hackerrank.com/challenges/py-if-else/problem

O que vamos aprender hoje?

Objetivos

- Aprender a realizar fluxos de repetição em Python
  - While
  - For
- Aprender os operadores break e continue

Tópicos da Aula

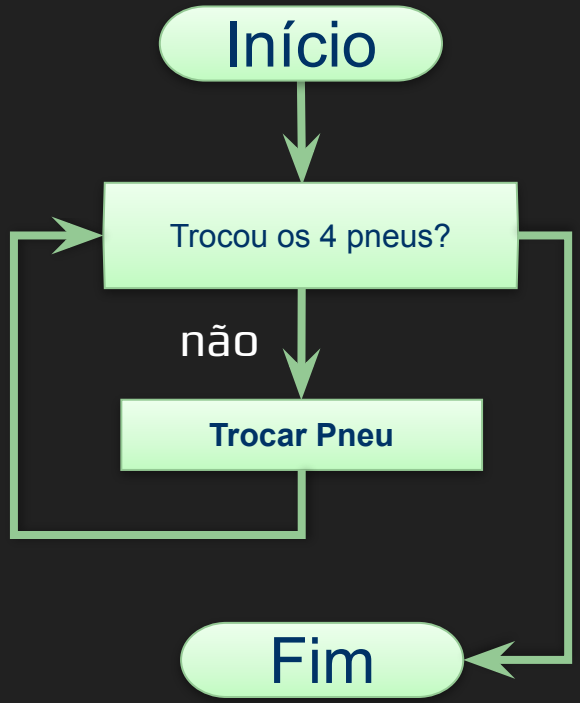
- Estrutura de Repetição
- Operador While
- Operador For
- Operadores break e continue

Estruturas de Repetição

Estruturas de Repetição

- São usadas para *repetir* um bloco de comandos
- Sequência de instruções que são repetidas até que certa(s) condição(ões) sejam atendidas
- Podem ser controlados na entrada ou saída do corpo do laço

Algoritmo para trocar um pneu



Estrutura de repetição

Estruturas de Repetição

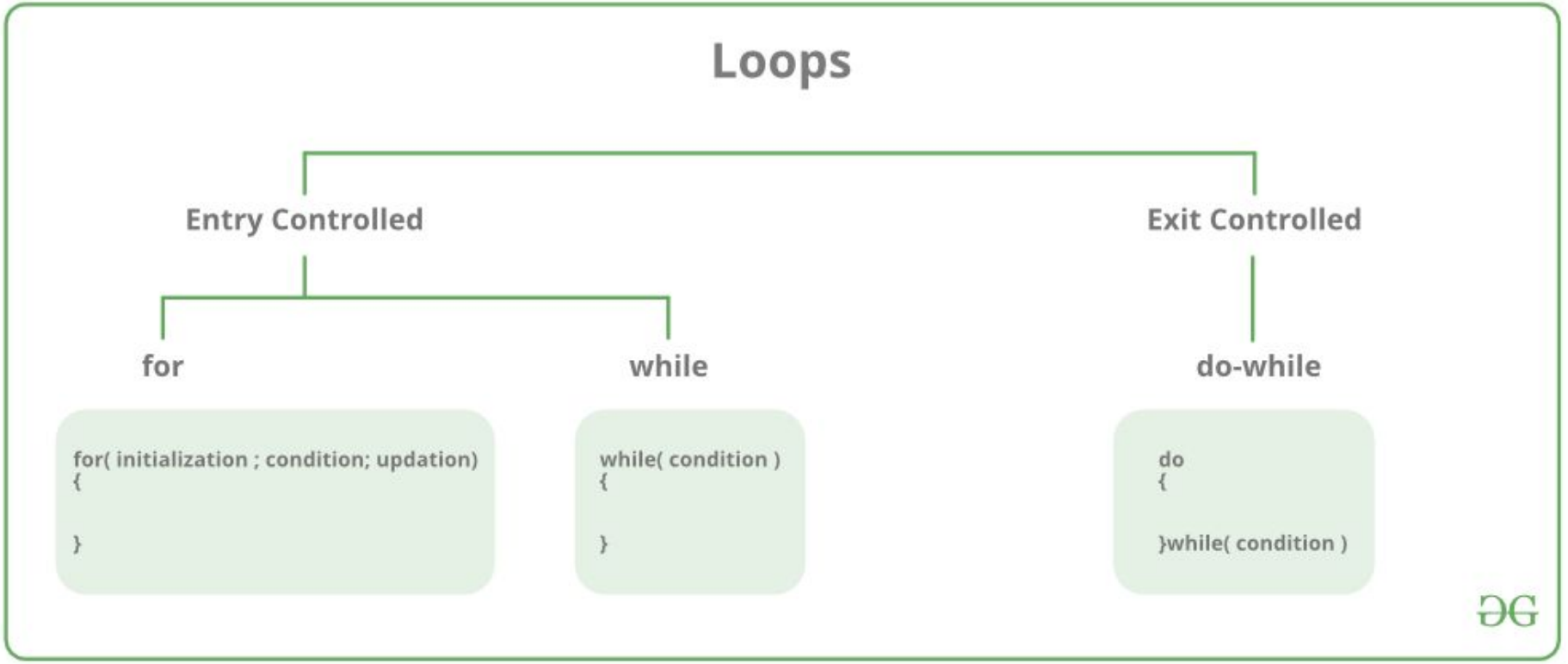
- Podem ser divididas em dois tipos:
  - As que a condição de parada é testada na entrada
  - E as que a condição de parada é testada na saída

Estruturas de Repetição

- Entrada
  - Condição é testada antes de entrar no corpo do laço
  - *For* e *While*
- *for* (inicialização; teste; atualização) { corpo }
- Inicialização; *while* (teste) {corpo; atualização}

Estruturas de Repetição

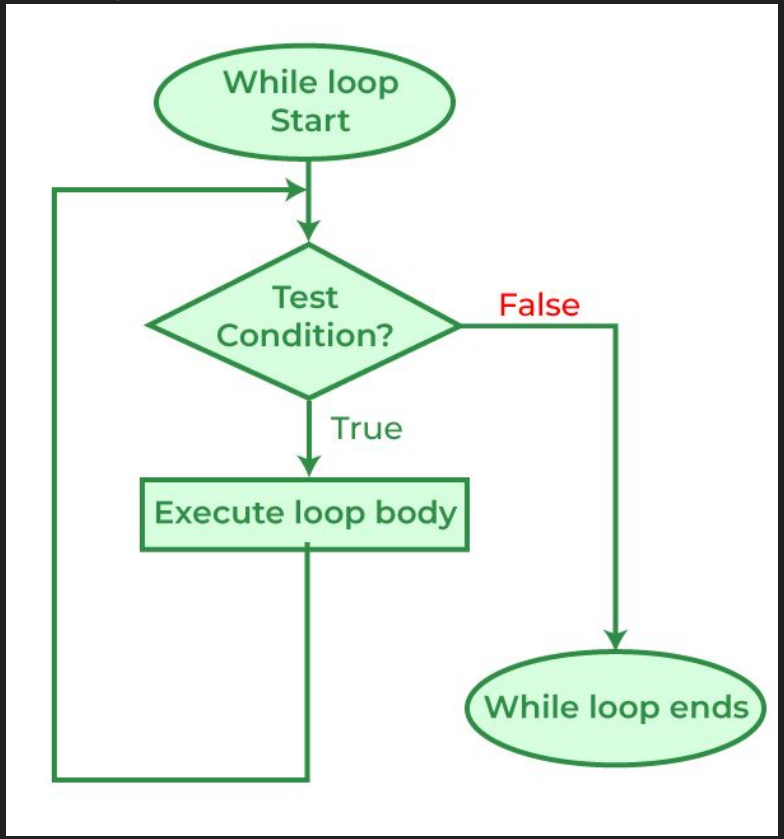
- Saída
  - Condição é testada ao final do corpo do laço
  - O corpo do laço é sempre executado **pelo menos** uma vez
  - *do* {corpo; atualização} *while* (teste)
  - Não existe nativamente em Python!



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

While

- Geralmente, os laços de *for* são feitos quando sabemos quantas iterações queremos. Quando não sabemos, é mais comum usar um *while*
- Normalmente é possível usar qualquer um dos dois para laços
  - Mas a legibilidade pode ficar difícil ou não intuitiva



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

While

- While -> Enquanto
- Enquanto a condição for verdadeira, executa o laço
- Condição é qualquer expressão
  - Booleano, inteiro, lista, string...
  - 0 é falso
  - != 0 é verdadeiro

While

```
a, b = 0, 1
while a < 10:
    print(a)
    a, b = b, a+b
```

While

- O corpo do While pode ter quantas expressões forem necessárias
  - Todas identadas
- Obs:
  - Em python, podemos retornar várias coisas de uma mesma função!

While

- Assim como If, While pode ter várias condições
- Operadores lógicos são necessários
  - and, or, not...
- Cuidado!
  - As variáveis verificadas nas condições precisam ser atualizadas para que o while pare!
    - Senão...

Loop infinito. Fonte: https://media.tenor.com/HoeUWr262WwAAAAC/mario-running.gif

While

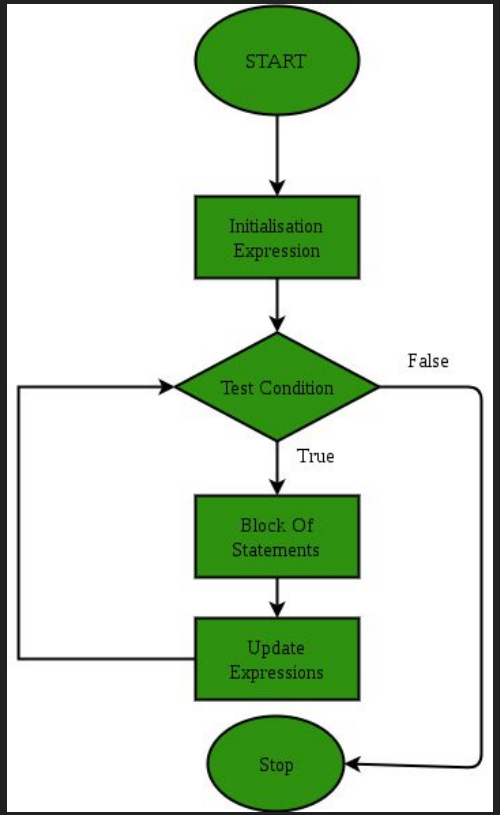
```
i = 0
j = 10
while i < 3 and j > 0:
   print("%d %d" % (i, j))
   i+=1
   j-=i
```

Exemplos

For

For

- Uma variável de laço controla o laço
- Inicializamos esta variável com um valor
- Checamos se a variável é menor ou maior que um valor
  - Se for verdadeiro
    - Continua para o bloco
    - Atualiza a variável do laço
  - Se for falso
    - Sai do laço



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

For

- Duas opções principais para iterar sobre elementos
  - “i in range(x, y, z)”
    - Itera como um for clássico
    - x é o começo, y é a parada
    - z é o “passo”
  - A segunda, sobre listas, veremos no final

For

```
for i in range(3):
  print(i)


for i in range(1, 5):
  print(i)


for i in range(1, 10, 2):
  print(i)
```

Exemplos

Break e Continue

Break e Continue

- É possível sair de um loop sem sair da condição do loop
  - Comando *break*
  - Útil para loops com condições de exceção
  - Evitar
    - Difícil de encontrar no código
- Sai apenas do loop mais interno*

Break e Continue

```
x = 3
for n in range(1, 10):
    if n % x == 0:
        print(n, 'é divisível por', x)
        break
    else:
        print(n, 'não é divisível por', x)
```

Break e Continue

- É possível continuar para a próxima iteração do loop sem passar pelas próximas linhas
  - Comando *continue*
  - Útil para pular linhas indesejadas quando uma condição específica é encontrada
  - Evitar
    - Difícil de encontrar no código
- Continua apenas o loop mais interno*

Break e Continue

```
x = 3
for n in range(1, 10):
  if n % x == 0:
    print(n, 'é divisível por', x)
    continue
  if n % 2 == 0:
    print(n, 'é divisível por 2')
  else:
    print(n, 'não é divisível')
```

Exemplos

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
5. https://docs.python.org/3/tutorial/introduction.html
6. https://docs.python.org/3/tutorial/controlflow.html
7.