FIAP SCHOOL

# ENSINO MÉDIO TÉCNICO EM INFORMÁTICA

FIAP SCHOOL

# ANÁLISE DE SISTEMAS

Professor Me. Eng. Rodolfo Magliari de Paiva

# Estruturas Lógicas, Condicionais e de Repetição em Python

Estruturas **lógicas, condicionais e de repetição** são conceitos fundamentais em programação que permitem **controlar o fluxo** de execução de um programa, além de garantirem que as operações sejam executadas corretamente.

Dentre suas diversas aplicações, são muito úteis para testar modelos.

Vamos entender cada um deles.

**Estruturas Lógicas:** As estruturas lógicas em Python são usadas para avaliar condições e tomar decisões com base em operadores lógicos.

Os principais operadores lógicos são:

- **and (e):** Verdadeiro se todas as condições forem verdadeiras.
- **or (ou):** Verdadeiro se pelo menos uma condição for verdadeira.
- **not (não):** Inverte o valor da condição (verdadeiro vira falso e vice-versa).

**Estruturas Condicionais:** As estruturas condicionais permitem que o programa execute diferentes blocos de código dependendo de uma condição.

Em Python, as principais são:

- **if (se):** Executa um bloco de código se a condição for verdadeira.
- **else (senão):** Executa um bloco de código se a condição do if for falsa.
- **elif (senão se):** Permite testar múltiplas condições em sequência.

Exemplo 1:
Escreva um programa em Python que verifique se uma pessoa pode dirigir com base na sua idade e na posse de carteira de motorista.

Resposta:

```
idade = 18
tem_carteira = True

if idade >= 18 and tem_carteira:
    print("Pode dirigir.")
else:
    print("Não pode dirigir.")
```

Exemplo 2:
Escreva um programa em Python que determine a situação de um aluno com base na sua nota e frequência nas aulas.

Resposta:

```
nota = 7.5
frequencia = 50

if nota >= 7 and frequencia >= 75:
    print("Aprovado!")
elif nota >= 5 and frequencia >= 75:
    print("Recuperação.")
else:
    print("Reprovado.")
```

Exemplo 3:
Crie um programa em Python que verifique o estoque de café e tome uma decisão com base na disponibilidade.

Resposta:

```
tem_cafe = False

if not tem_cafe:
    print("Precisamos fazer mais café!")
else:
    print("Já temos café suficiente.")
```

Exemplo 4:
Crie um programa em Python que verifique se um número é par ou positivo.

Resposta:

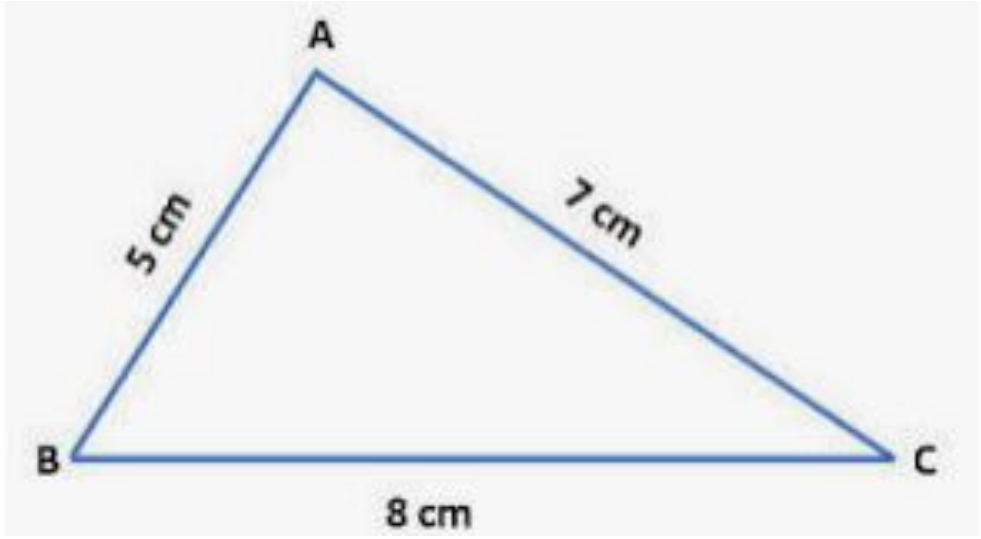
```
numero = int(input("Digite um número: "))

if numero > 0 or numero % 2 == 0:
    print("O número é positivo ou par")
else:
    print("O número não é positivo nem par")
```

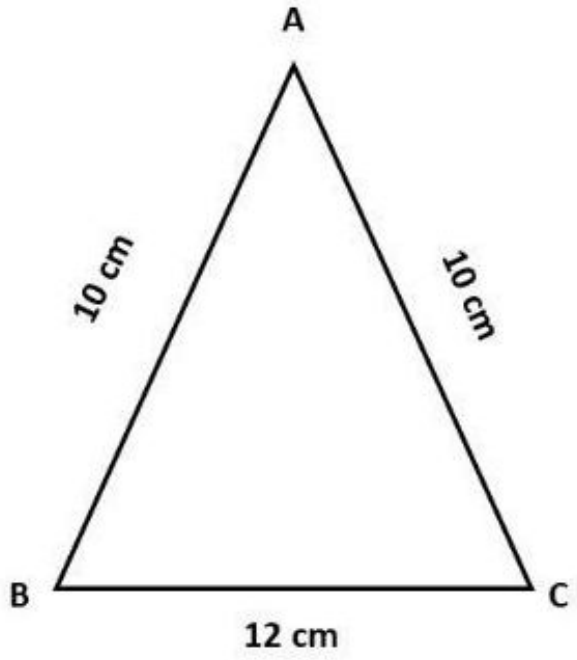
Exercício 1:

Escreva um programa em Python que verifique se o triângulo é escaleno, caso contrário, responda que não é um triângulo escaleno.

a)




b)



Resposta:
a)

```
a = 5
b = 7
c = 8

if a != b and b != c and a != c:
   print("É escaleno!")
else:
   print("Não é escaleno.")
```

b)

```
a = 10
b = 10
c = 12

if a != b and b != c and a != c:
   print("É escaleno!")
else:
   print("Não é escaleno.")
```

Exercício 2:
Escreva um programa em Python que verifique se um número x é positivo ou negativo e execute a seguinte lógica:
Se x for positivo ou zero, calcule e imprima a raiz quadrada de x.
Se x for negativo, imprima a mensagem: "O número é negativo".

a) $\sqrt{-16}$

b) $\sqrt{81}$

Resposta:

a)

```
x = -16

if x >= 0:
   print(x**(1/2))
else:
   print("O número é negativo")
```

b)

```
x = 81

if x >= 0:
   print(x**(1/2))
else:
   print("O número é negativo")
```

**Estruturas de Repetição:** Também conhecidas como Laços, as estruturas de repetição permitem que um bloco de código seja executado várias vezes (loops).

Em Python, as principais são:

- **for (para):** É usado para iterar sobre uma sequência (como uma lista, string ou range) um número conhecido de vezes.
- **while (enquanto):** Repete um bloco de código enquanto uma condição for verdadeira.

Exemplo 1:
Escreva um programa em Python que imprima os números de 0 a 4, cada um precedido pela mensagem "Número:" utilizando a estrutura for.

Resposta:

```
for i in range(5):
    print("Número:", i)
```

Exemplo 2:
Escreva um programa em Python que imprima os números de 0 a 4, cada um precedido pela mensagem "Número:", utilizando o laço de repetição while.

Resposta:

```
n = 0

while n < 5:
    print("Número:", n)
    n += 1
```

Exercício 1:
Escreva um programa em Python que calcule e exiba o cubo dos números inteiros de 1 a 5, utilizando a estrutura for.

Resposta:

```
for numero in range(1, 6):
    print(numero ** 3)
```

Exercício 2:
Escreva um programa em Python que imprima os números de 1 a 100, pulando de 10 em 10, ou seja, exibindo apenas os valores 1, 11, 21, 31, ..., 91, utilizando a estrutura while.

Resposta:

```
numero = 1

while numero < 100:
    print(numero)
    numero += 10
```

Existem diversas outras funções e estruturas em Python para controle de fluxo, como match-case, break, continue, map, filter, e compreensões de lista, além das funções já vistas, que permitem gerenciar a execução do código de forma eficiente e adaptável a diferentes cenários de programação.

Além das Estruturas Lógicas, Condicionais e de Repetição, existem também as Estruturas de Tratamento de Exceções, que servem para gerenciamento de erros e para sinalização de problemas durante a execução do programa.

# OBRIGADO!

Contato:
profrodolfo.paiva@fiap.com.br

FIAP SCHOOL