PROGRAMAÇÃO I

Curso Técnico Subsequente em Informática

Lucas Sampaio Leite

Operadores aritméticos

INSTITUTO FEDERAL
Baiano

Operação	Operador
Adição	+
Subtração	-
Multiplicação	*
Divisão	/

Operadores aritméticos

INSTITUTO FEDERAL
Baiano

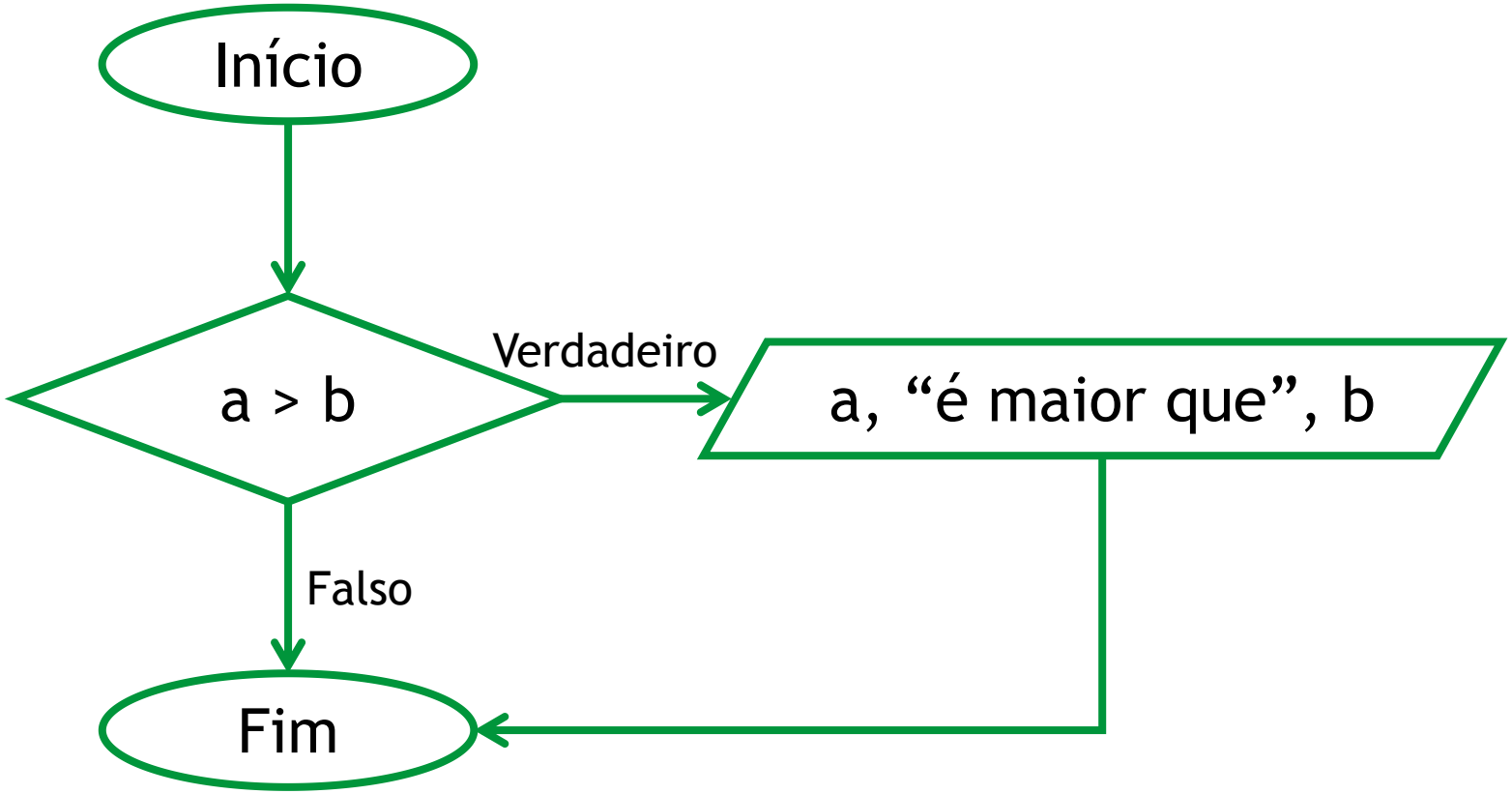
Operação	Operador
Exponenciação	**
Parte inteira do resultado da divisão	//
Módulo	%

Ordem de precedência dos operadores aritméticos

INSTITUTO FEDERAL
Baiano

Operador	Ordem de resolução na expressão
()	1º
**	2º
*, /, //, %	3º
+, -	4º

Estruturas condicionais



Estruturas condicionais

INSTITUTO FEDERAL
Baiano

- Comandos condicionais permitem controlar o fluxo de execução de um programa, escolhendo qual instrução deve ser executada a seguir.
- A execução de uma instrução depende do resultado de uma condição, representada por uma expressão booleana.
  - Expressões booleanas são aquelas que resultam em Verdadeiro (True) ou Falso (False).
- Essas expressões podem ser construídas com o uso de operadores relacionais e operadores lógicos:
  - Operadores relacionais são utilizados para realizar comparações (como ==, !=, <, >, <=, >=.
  - Operadores lógicos permitem combinar expressões booleanas, aplicando lógica (and, or, not).

Operadores relacionais

INSTITUTO FEDERAL
Baiano

Operador	Referente a:
==	Igual a
!=	Diferente
>=	Maior ou igual
>	Maior que
<	Menor que
<=	Menor ou igual

Operadores lógicos

INSTITUTO FEDERAL
Baiano

Operador	Referente a:
and	e
or	ou
not	não

Operadores lógicos

INSTITUTO FEDERAL
Baiano

- and (E lógico):
  - O resultado é verdadeiro apenas se todos os operandos forem verdadeiros.
  - Se algum operando for falso, o resultado será falso.
- or (OU lógico):
  - O resultado é verdadeiro se pelo menos um dos operandos for verdadeiro.
  - O resultado será falso apenas se todos os operandos forem falsos.
- not (negação):
  - Inverte o valor lógico de uma proposição.
  - Se a proposição for verdadeira, torna-se falsa.
  - Se for falsa, torna-se verdadeira.

Tabela verdade

INSTITUTO FEDERAL
Baiano

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Operadores lógicos

INSTITUTO FEDERAL
Baiano

- O que será impresso?

main.py > ...

```python
a = True
b = False
c = a and b
d = a or b
print(c)
print(d)
print(not(c))
print(not(d))
```

teste.py > ...

```python
c = 23
d = 27
a = (c < 20) or (d > c)
b = (c < 20) and (d < c)
print('a= {}; b= {}.'.format(a, b))
```

Estruturas condicionais (if)

INSTITUTO FEDERAL
Baiano

- O comando if é uma estrutura condicional que permite executar um bloco de código apenas se uma determinada condição booleana for verdadeira.
- Sintaxe:

```
if <condição>:
    bloco verdadeiro
```

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

if numero%2 == 0:
    print(f"{numero} é par.")

if numero%2 == 1:
    print(f"{numero} é ímpar.")
```

Estruturas condicionais (if)

INSTITUTO FEDERAL
Baiano

```
idade = 17

if idade >= 16:
    print("Você está apto a votar.")
```

```
Você está apto a votar.
```

Estruturas condicionais (if-else)

INSTITUTO FEDERAL
Baiano

- O comando else é usado junto com o if para definir um bloco alternativo de código que será executado caso a condição do if seja falsa.
- Sintaxe:

```
if <condição>:
    bloco verdadeiro
else:
    bloco falso
```

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

if numero%2 == 0:
    print(f"{numero} é par.")
else:
    print(f"{numero} é ímpar.")
```

Estruturas condicionais (if-else)

INSTITUTO FEDERAL
Baiano

```
idade = 15

if idade >= 16:
    print("Você está apto a votar.")
else:
    print("Você ainda não pode votar.")
```

```
Você ainda não pode votar.
```

Estruturas condicionais (if-elif-else)

INSTITUTO FEDERAL
Baiano

- O comando elif (abreviação de "else if") é utilizado para testar múltiplas condições em uma estrutura condicional. Ele é avaliado somente se o if anterior for falso, e seu bloco será executado se a condição do elif for verdadeira.

main.py > ...

```
1   a = 5
2   if a > 0:
3       print('valor positivo')
4   else:
5       if a < 0:
6           print('valor negativo')
7       else:
8           print('valor nulo')
```

=

main.py > ...

```
1   a = 5
2   if a > 0:
3       print('valor positivo')
4   elif a < 0:
5       print('valor negativo')
6   else:
7       print('valor nulo')
```

Estruturas condicionais (if-elif-else)

INSTITUTO FEDERAL
Baiano

```
num1 = int(input("Digite o primeiro número: "))
num2 = int(input("Digite o segundo número: "))

if num1 == num2:
    print("Os números digitados são iguais.")
elif num1 > num2:
    print("O primeiro número é maior que o segundo.")
else:
    print("O segundo número é maior que o primeiro.")
```

Estruturas condicionais (if-elif-else)

INSTITUTO FEDERAL
Baiano

```
nota = 7.5

if nota >= 9:
    print("Excelente desempenho!")
elif nota >= 7:
    print("Bom trabalho!")
elif nota >= 5:
    print("Você passou, mas pode melhorar.")
else:
    print("Reprovado. Estude mais para a próxima.")
```

```
Bom trabalho!
```

Exercícios rápidos

INSTITUTO FEDERAL
Baiano

- Leia a idade e o tempo de serviço de um trabalhador e escreva se ele pode ou não se aposentar. As condições para aposentadoria são:
  - Ter pelo menos 65 anos,
  - Ou ter trabalhado pelo menos 30 anos,
  - Ou ter pelo menos 60 anos e trabalhado pelo menos 25 anos.

Exercícios rápidos

INSTITUTO FEDERAL
Baiano

- Faça um programa que leia 2 notas de um aluno, verifique se as notas são válidas e exiba na tela a média destas notas. Uma nota válida deve ser, obrigatoriamente, um valor entre 0.0 e 10.0, onde caso a nota não possua um valor válido, este fato deve ser informado ao usuário e o programa termina.

Estruturas de repetição

INSTITUTO FEDERAL
Baiano

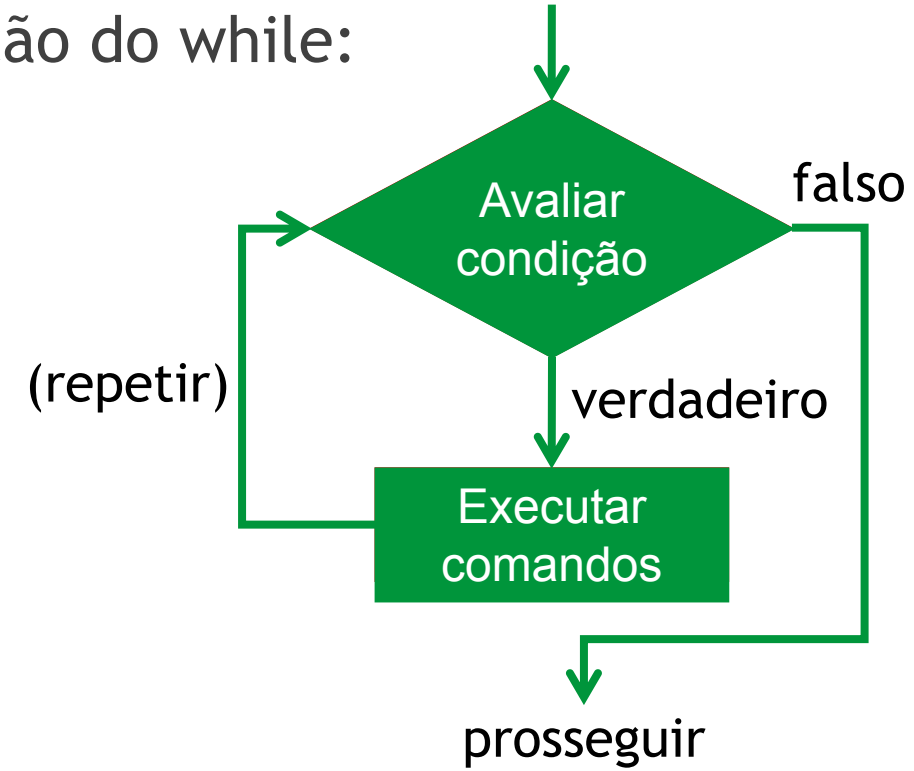
- Perceba o seguinte: em muitos casos, precisamos executar os mesmos comandos várias vezes, como ao calcular a média de uma turma inteira.
- Repetir o código manualmente seria ineficiente e cansativo.
- As linguagens de programação oferecem mecanismos que automatizam repetições, conhecidos como estruturas de repetição ou, em inglês, loops.
- No Python, contamos com duas principais formas de criar repetições:
  - while → ideal para quando não sabemos quantas vezes o código deve se repetir.
  - for → perfeito para repetições com quantidade conhecida ou ao percorrer coleções (como listas).

Estruturas de repetição

- Repetição condicional: executa um bloco de código enquanto uma condição lógica for verdadeira.
  - Utilizamos o comando while.
- Repetição contável: executa um bloco de código um número definido de vezes, geralmente com base em um contador.
  - Utilizamos o comando for.

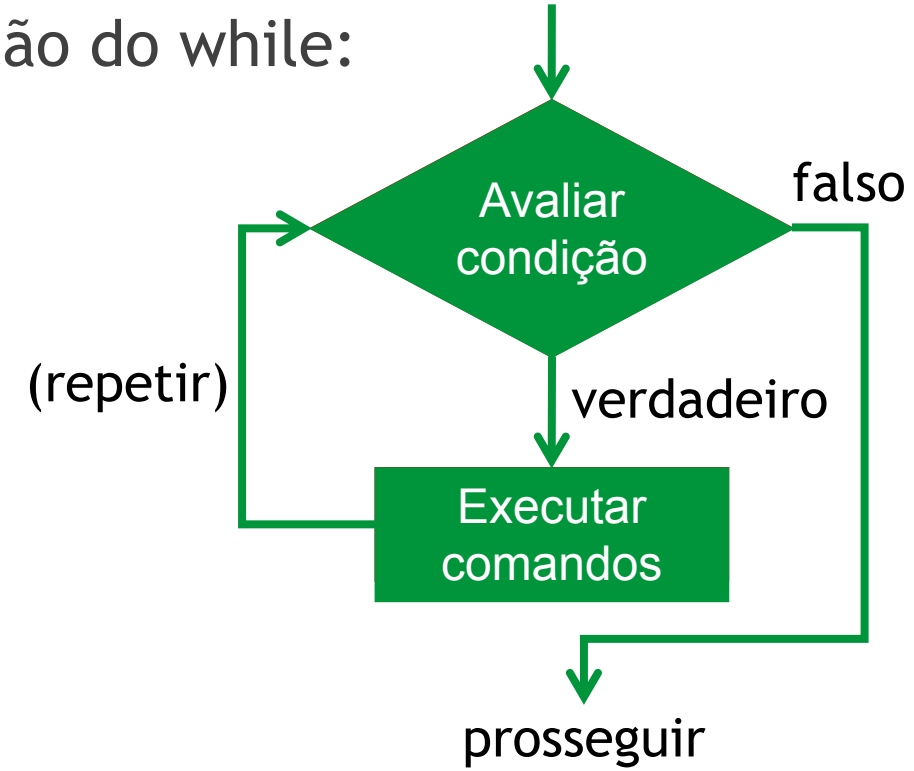
Estruturas de repetição (while)

- Fluxograma de representação do while:



Estruturas de repetição (while)

- Fluxograma de representação do while:



Observe que há a possibilidade de nunca se executar os comandos caso a primeira avaliação da condição já resulte em falso.

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

- A estrutura while possui a seguinte sintaxe:

```
while <condição>:
    comandos
```

- Podemos interpretar assim: “Enquanto a condição booleana for verdadeira, execute o bloco de comandos abaixo.”
- Isso significa que o bloco de código será repetido sempre que a condição for verdadeira.

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

- A estrutura while possui a seguinte sintaxe:

```
while <condição>:
    comandos
```

- Podemos interpretar assim: “Enquanto a condição booleana for verdadeira, execute o bloco de comandos abaixo.”
- Isso significa que o bloco de código será repetido sempre que a condição for verdadeira.

Atenção!
Algo dentro do laço deve alterar o valor da condição, caso contrário o laço nunca será interrompido — e o programa pode entrar em um loop infinito.

Estruturas de repetição (while)

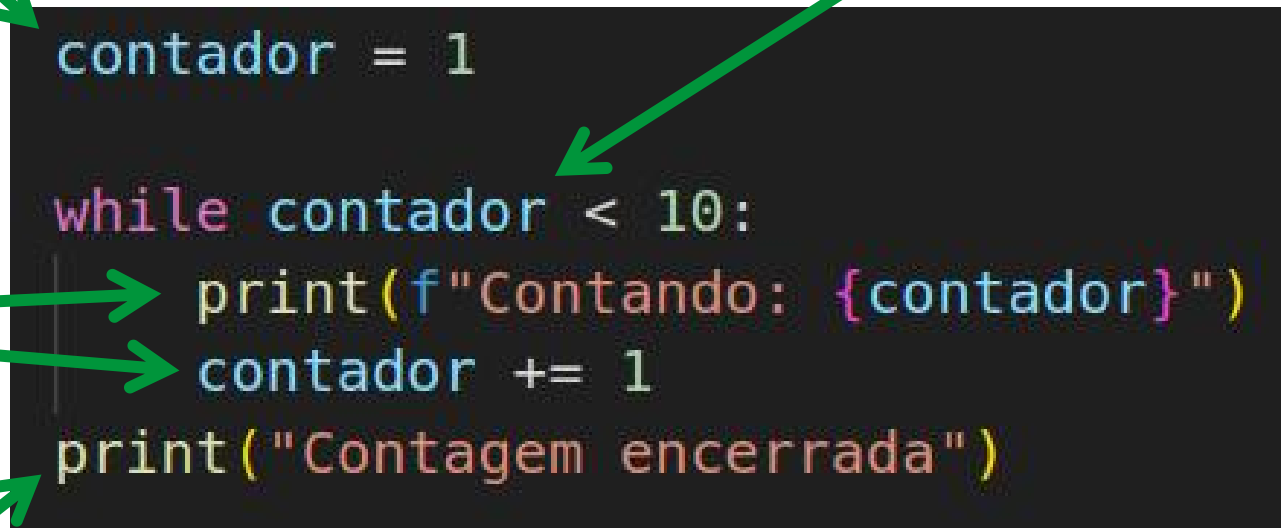
INSTITUTO FEDERAL
Baiano

Comando antes do while

Condição do while

Bloco de comandos do while (“Corpo do laço)

Comando após o while



Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
contador = 1

while contador < 10:
    print(f"Contando: {contador}")
    contador += 1
print("Contagem encerrada")
```

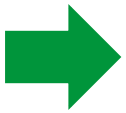
Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
contador = 1

while contador < 10:
    print(f"Contando: {contador}")
    contador += 1
print("Contagem encerrada")
```



```
Contando: 1
Contando: 2
Contando: 3
Contando: 4
Contando: 5
Contando: 6
Contando: 7
Contando: 8
Contando: 9
Contagem encerrada
```

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
contador = 0

while contador <= 50:
    print(contador)
    contador = contador + 5

print("Contagem encerrada")
```

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
contador = 0

while contador <= 50:
    print(contador)
    contador = contador + 5

print("Contagem encerrada")
```

```
0
5
10
15
20
25
30
35
40
45
50
Contagem encerrada
```

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
i = 1
while i != i:
    print(i)
    i += 1
```

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
i = 1
while i != i:
    print(i)
    i += 1
```

A execução do programa nunca vai entrar na repetição (no laço).

Condição será sempre false!!!

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
i = 1
while i == i:
    print(i)
    i += 1
```

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```
i = 1
while i == i:
    print(i)
    i += 1
```

A execução do programa entra na repetição e nunca sai dela (laço infinito).

Condição será sempre True!!!

Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

```python
senha_correta = "python123"
tentativa = input("Digite a senha: ")

while tentativa != senha_correta:
    print("Senha incorreta. Tente novamente.")
    tentativa = input("Digite a senha: ")

print("Acesso liberado!")
```

O que o programa faz?

Estruturas de repetição (while)

```python
senha_correta = "python123"
tentativa = input("Digite a senha: ")

while tentativa != senha_correta:
    print("Senha incorreta. Tente novamente.")
    tentativa = input("Digite a senha: ")

print("Acesso liberado!")
```

```
Digite a senha: python321
Senha incorreta. Tente novamente.
Digite a senha: python
Senha incorreta. Tente novamente.
Digite a senha: python123
Acesso liberado!
```

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

- Outros exemplos:

```
i = 10
while i >= 0:
    print(i)
    i -= 2
```

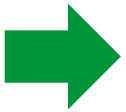
Qual a saída do programa?

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

- Outros exemplos:

```
i = 10
while i >= 0:
    print(i)
    i -= 2
```



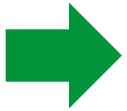
```
10
8
6
4
2
0
```

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

- Outros exemplos:

```
i = 10
while i >= 0:
    print(i)
    i -= 2
```



```
10
8
6
4
2
0
```

Qual a saída do programa?

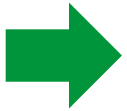
```
a = 0
b = 2
while a <= b:
    print(f"{a} <= {b}")
    a += 1
```

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

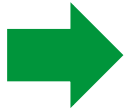
- Outros exemplos:

```
i = 10
while i >= 0:
    print(i)
    i -= 2
```



```
10
8
6
4
2
0
```

```
a = 0
b = 2
while a <= b:
    print(f"{a} <= {b}")
    a += 1
```



```
0 <= 2
1 <= 2
2 <= 2
```

Estruturas de repetição (while)

INSTITUTO FEDERAL
Baiano

- Resolução do problema das médias com a estrutura de repetição while:

```
repetir = True

while repetir:
    nota1 = float(input("Digite a primeira nota do aluno: "))
    nota2 = float(input("Digite a segunda nota do aluno: "))
    nota3 = float(input("Digite a terceira nota do aluno: "))

    media = (nota1 + nota2 + nota3) / 3

    print(f"A média do aluno é: {media:.2f}")

    resposta = input("Deseja calcular uma nova média? (s/n): ")
    if resposta == "n":
        repetir = False
```

Exercícios rápidos

- Peça ao usuário para digitar vários números positivos.
- Quando um número negativo for digitado, o loop deve parar.
- Calcule e exiba a média dos números positivos digitados.

Exercícios rápidos

INSTITUTO FEDERAL
Baiano

- Peça ao usuário um número inteiro positivo.
- Use while para exibir o número invertido.
- Exemplo:
  - Digite um número: 1234
  - Saída: 4321

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- O for é uma estrutura de repetição, assim como o while, mas costuma ser mais usada quando se sabe exatamente quantas vezes o bloco de código deve ser executado.
- Sintaxe do for com a função range():

```
for i in range(<n>):
    comandos
```

- A função range() gera uma sequência de números inteiros e é muito usada com o for para repetir algo várias vezes ou iterar com base em contadores.
- Ao utilizar range(n), a variável de controle (geralmente chamada de i) começa com o valor 0 e é incrementada de 1 em 1 até atingir n - 1.

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- Sintaxe do for com a função range():

```
for i in range(<n>):
    comandos
```

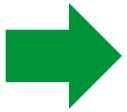
- Para usar o for, é sempre necessário indicar uma variável iteradora, que assumirá um valor diferente a cada repetição do laço.
- Também é necessário indicar os limites de iteração, ou seja, de onde a contagem começa, onde termina e, se desejado, o valor do passo (incremento ou decremento).

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- Exemplo:

```
for i in range(5):
    print(i)
```



```
0
1
2
3
4
```

Neste caso, o laço for imprime na tela o valor atual de i a cada iteração.
A variável i começa em 0 e é incrementada de um em um até atingir o valor 4.

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- É possível definir um valor inicial diferente de 0 para a contagem.
- Para isso, basta utilizar a função range() com dois parâmetros: o valor inicial e o valor final.
- Sintaxe do for com a função range():

```
for i in range(<início>, <fim>):
    comandos
```

Estruturas de repetição (for)

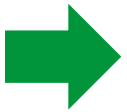
INSTITUTO FEDERAL
Baiano

- É possível definir um valor inicial diferente de 0 para a contagem.
- Para isso, basta utilizar a função range() com dois parâmetros: o valor inicial e o valor final.
- Sintaxe do for com a função range():

```
for i in range(<início>, <fim>):
    comandos
```

- Exemplo:

```
for i in range(2, 6):
    print(i)
```



```
2
3
4
5
```

Neste caso, o laço for imprime na tela o valor de i a cada iteração.
A variável i inicia em 2 e é incrementada de um em um até atingir o valor 5.

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- A função range() também pode ser usada para controlar a execução do laço for, utilizando três parâmetros: range(m, n, p).
- Essa forma gera uma sequência de números inteiros que começa em m, vai até n - 1 e é incrementada de p em p.

```
m = 1
n = 100
p = 2
for i in range(m, n, p):
    print(i)
```

```
for i in range(1, 100, 2):
    print(i)
```

O que será impresso?

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- Também é possível utilizar um passo negativo na função range() para fazer a contagem regressiva, ou seja, contar de trás para frente.
- Exemplo:

```
for i in range(5, 0, -1):
    print(i)
```

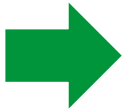
O que será impresso?

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- Também é possível utilizar um passo negativo na função range() para fazer a contagem regressiva, ou seja, contar de trás para frente.
- Exemplo:

```
for i in range(5, 0, -1):
    print(i)
```



```
5
4
3
2
1
```

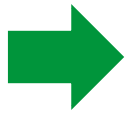
Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- A variável iteradora também pode assumir diretamente os valores de uma string ou dos elementos de uma lista:
- Exemplo:

```
string = "Programação é 10"

for char in string:
    print(char)
```



```
P
r
o
g
r
a
m
a
ç
ã
o

é

1
0
```

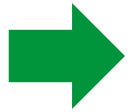
Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- A variável iteradora também pode assumir diretamente os valores de uma string ou dos elementos de uma lista:
- Exemplo:

```
lista = [2, 4, 6.5, 8, 10]

for elemento in lista:
    print(elemento)
```



```
2
4
6.5
8
10
```

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- Pode-se usar range(len(lista)) para percorrer a lista acessando os elementos por seus índices.
- Exemplo:

```
nomes = ["Ana", "Carlos", "Maria", "João"]

for i in range(len(nomes)):
    print(f"Índice {i}: {nomes[i]}")
```

```
Índice 0: Ana
Índice 1: Carlos
Índice 2: Maria
Índice 3: João
```

Estruturas de repetição (for)

INSTITUTO FEDERAL
Baiano

- Pode-se usar enumerate(lista) para percorrer a lista obtendo simultaneamente índice e valor.
- Exemplo:

```
nomes = ["Ana", "Carlos", "Maria", "João"]

for i, nome in enumerate(nomes):
    print(f"Índice {i}: {nome}")
```

```
Índice 0: Ana
Índice 1: Carlos
Índice 2: Maria
Índice 3: João
```

Exercícios rápidos

INSTITUTO FEDERAL
Baiano

- Strings, assim como listas, são sequências (iteráveis). Portanto, é possível percorrer seus caracteres tanto acessando-os pelos índices quanto utilizando a função enumerate. Crie uma string e percorra todos os caracteres empregando as duas abordagens.

Exercícios rápidos

INSTITUTO FEDERAL
Baiano

- Escreva um programa que leia um número inteiro positivo N e calcule a soma de todos os números pares de 1 até N (inclusive).

Exercícios rápidos

- Peça ao usuário para digitar uma palavra e mostre-a invertida usando apenas um laço for.

Exercícios rápidos

INSTITUTO FEDERAL
Baiano

- Escreva um programa que leia dois números inteiros a e b e exiba todos os números primos entre a e b utilizando for.

Dúvidas

INSTITUTO FEDERAL
Baiano

PROGRAMAÇÃO I

Curso Técnico Subsequente em Informática

Lucas Sampaio Leite