

LÓGICA E LINGUAGEM DE PROGRAMAÇÃO

Curso Técnico Integrado em Informática
Lucas Sampaio Leite



Estruturas de repetição

- Imagine a seguinte situação: Você precisa criar um programa que calcule a média de um aluno.
- Este programa é bastante simples, bastaria:
 - Ler as notas digitadas pelo usuário no teclado;
 - Calcular a média dessas notas;
 - Imprimir o resultado.

Estruturas de repetição

- Imagine a seguinte situação: Você precisa criar um programa que calcule a média de um aluno.
- Este programa é bastante simples, bastaria:
 - Ler as notas digitadas pelo usuário no teclado;
 - Calcular a média dessas notas;
 - Imprimir o resultado.

Mas... e se tivermos que calcular 5, 10 ou até 100 alunos?
Você escreveria os mesmos comandos várias vezes para cada um deles?

Estruturas de repetição

- Imagine a seguinte situação: Você precisa criar um programa que calcule a média de um aluno.
- Este programa é bastante simples, bastaria:
 - Ler as notas digitadas pelo usuário no teclado;
 - Calcular a média dessas notas;
 - Imprimir o resultado.

```
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}")
```

Estruturas de repetição

- Imagine a seguinte situação: Você precisa criar um programa que calcule a média de um aluno.
- Este programa é bastante simples, bastaria:
 - Ler as notas digitadas pelo usuário no teclado;
 - Calcular a média dessas notas;
 - Imprimir o resultado.

```
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: "))
```

Como poderíamos fazer para calcular a média de todos os alunos de uma turma em um mesmo programa?

Estruturas de repetição

- 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).

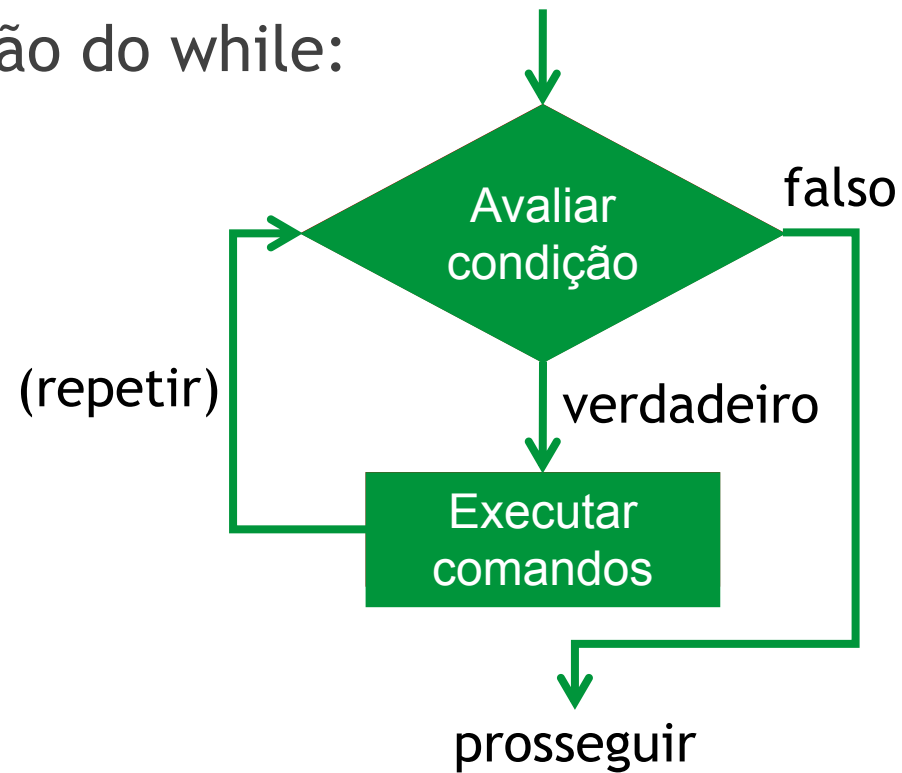
Estruturas de repetição

- 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.



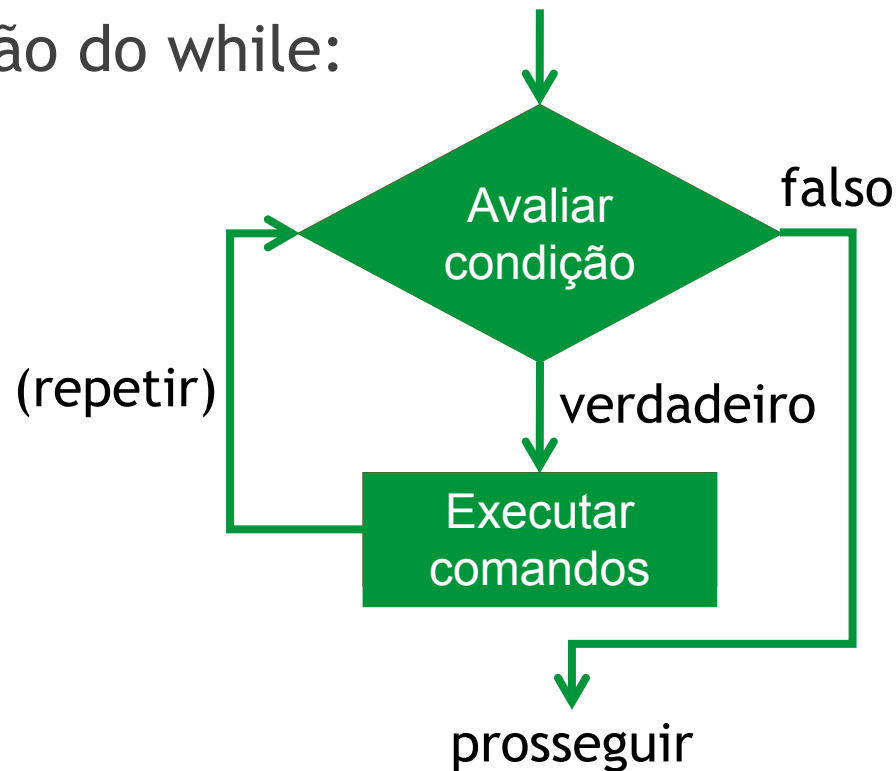
Estruturas de repetição (while)

- 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)

- 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)

- 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)

Comando antes do while

Condição do while

Bloco de comandos do
while ("Corpo do laço")

Comando após o while

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

Estruturas de repetição (while)

```
contador = 1

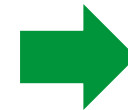
while contador < 10:
    print(f"Contando: {contador}")
    contador += 1
print("Contagem encerrada")
```

Qual a saída do
programa?

Estruturas de repetição (while)

```
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)

Passo 1: teste da
condição de parada.

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

Estruturas de repetição (while)

Passo 2: Caso a condição seja verdadeira, execute os comandos do bloco do while e volte ao passo 1.


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


Estruturas de repetição (while)

```
contador = 1

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

Passo 3: Caso a condição seja falsa, continue a execução com os comandos após o while.



Estruturas de repetição (while)

```
contador = 0

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

print("Contagem encerrada")
```

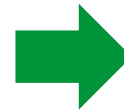
Qual a saída do
programa?

Estruturas de repetição (while)

```
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?

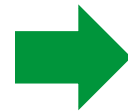
Estruturas de repetição (while)

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

Qual a saída do
programa?

Estruturas de repetição (while)

```
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?

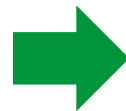
Estruturas de repetição (while)

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

Qual a saída do
programa?

Estruturas de repetição (while)

```
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)

```
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)

```
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)

- Outros exemplos:

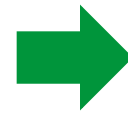
```
contador = 0
while contador < 10:
    print(contador)
    contador = contador + 1
```

Qual a saída do
programa?

Estruturas de repetição (while)

- Outros exemplos:

```
contador = 0
while contador < 10:
    print(contador)
    contador = contador + 1
```

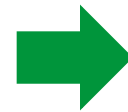


0
1
2
3
4
5
6
7
8
9

Estruturas de repetição (while)

- Outros exemplos:

```
contador = 0
while contador < 10:
    print(contador)
    contador = contador + 1
```



0
1
2
3
4
5
6
7
8
9

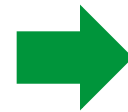
Qual a saída do
programa?

```
i = 10
while i >= 1:
    print(i)
    i = i - 1
```

Estruturas de repetição (while)

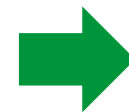
- Outros exemplos:

```
contador = 0
while contador < 10:
    print(contador)
    contador = contador + 1
```



0
1
2
3
4
5
6
7
8
9

```
i = 10
while i >= 1:
    print(i)
    i = i - 1
```



10
9
8
7
6
5
4
3
2
1

Estruturas de repetição (while)

- Outros exemplos:

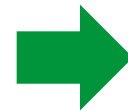
```
i = 10
while i >= 0:
    print(i)
    i -= 2
```

Qual a saída do
programa?

Estruturas de repetição (while)

- Outros exemplos:

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

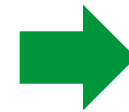


```
10  
8  
6  
4  
2  
0
```

Estruturas de repetição (while)

- Outros exemplos:

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



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

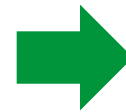
Qual a saída do
programa?

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


Estruturas de repetição (while)

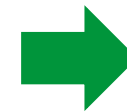
- Outros exemplos:

```
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)

- 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
```

Estruturas de repetição (while)

- Em alguns casos, pode ser necessário encerrar a repetição antes que a condição termine naturalmente.
- Para isso, usamos o comando `break`, que interrompe imediatamente o laço e faz o programa seguir para a próxima instrução após o laço.

```
while True:
    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":
        break
```

Estruturas de repetição (while)

- Em certas situações, pode ser útil ignorar apenas uma repetição do laço, sem interromper toda a estrutura.
- Para isso, utilizamos o comando `continue`, que pula imediatamente para a próxima iteração, sem executar o restante do bloco atual.

```
i = 0
while i < 100:
    i+=1
    if i % 5 == 0:
        continue
    print(i)

print()
```

O que o programa faz?

Exercícios

1. Escreva um programa que imprime todos os numeros de 0 até 50, incluindo-os.
2. Modifique o programa anterior de forma que este imprima apenas os números que são pares.
3. Escreva um programa para contar a quantidade de números pares entre dois números quaisquer fornecidos pelo usuário?
4. Escreva um programa para calcular o fatorial de um número fornecido pelo usuário.

Exercícios

5. Faça um programa que peça dois números, base e expoente, calcule e mostre o primeiro número elevado ao segundo número. Não utilize a função de potência da linguagem ou o operador de exponenciação.
6. Desenvolva um gerador de tabuada, capaz de gerar a tabuada de qualquer número inteiro entre 1 a 10. O usuário deve informar de qual numero ele deseja ver a tabuada. A saída deve ser como o exemplo abaixo:

```
Tabuada de 5:  
5 x 1 = 5  
5 x 2 = 10  
...  
5 x 10 = 50
```

Exercícios

7. A prefeitura de uma cidade deseja fazer uma pesquisa entre seus habitantes. Faça um algoritmo para coletar e armazenar dados sobre o salário e número de filhos de cada habitante e após as leituras, escrever:
- a) Média de salário da população
 - b) Média do número de filhos
 - c) Maior salário dos habitantes

Obs.: O final da leituras dos dados se dará com a entrada de um “salário negativo”.

Dúvidas



LÓGICA E LINGUAGEM DE PROGRAMAÇÃO

Curso Técnico Integrado em Informática
Lucas Sampaio Leite

