





FIAP

Aula 4

COMPUTATIONAL THINKING USING PYTHON



AGENDA

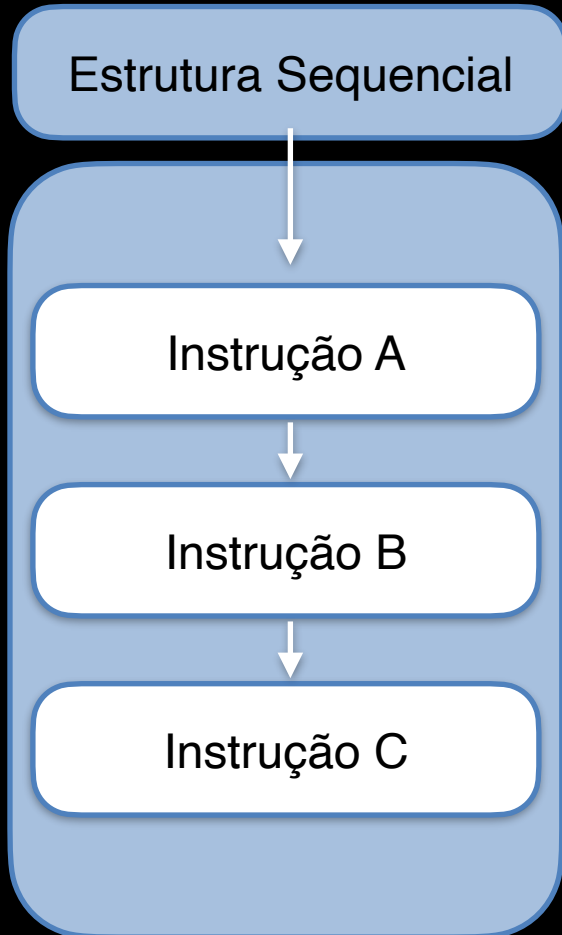


1. Estruturas de repetição aninhadas
2. Interrupções de laços
3. Simular estrutura de repetição *repeat...until* usando *while*
4. Utilizar laços para validar entrada de dados
5. Entender a necessidade das estruturas aninhadas
6. Resolver problemas com repetições aninhadas

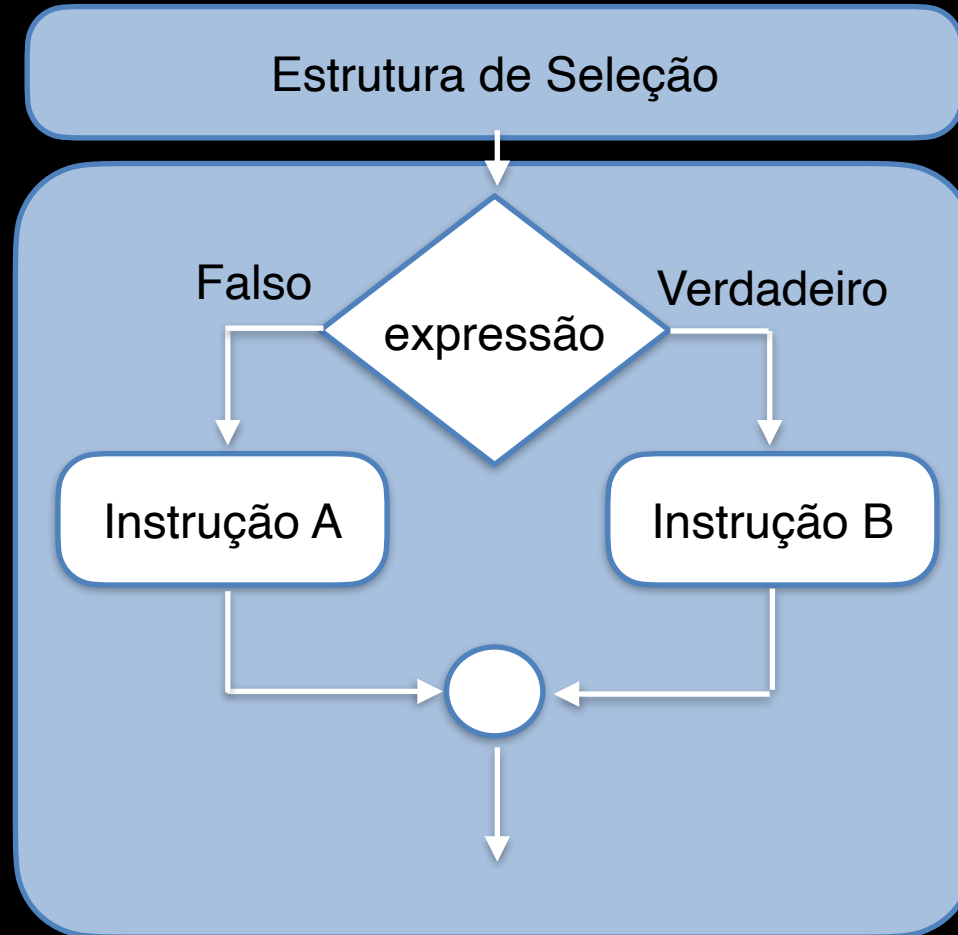


Representações

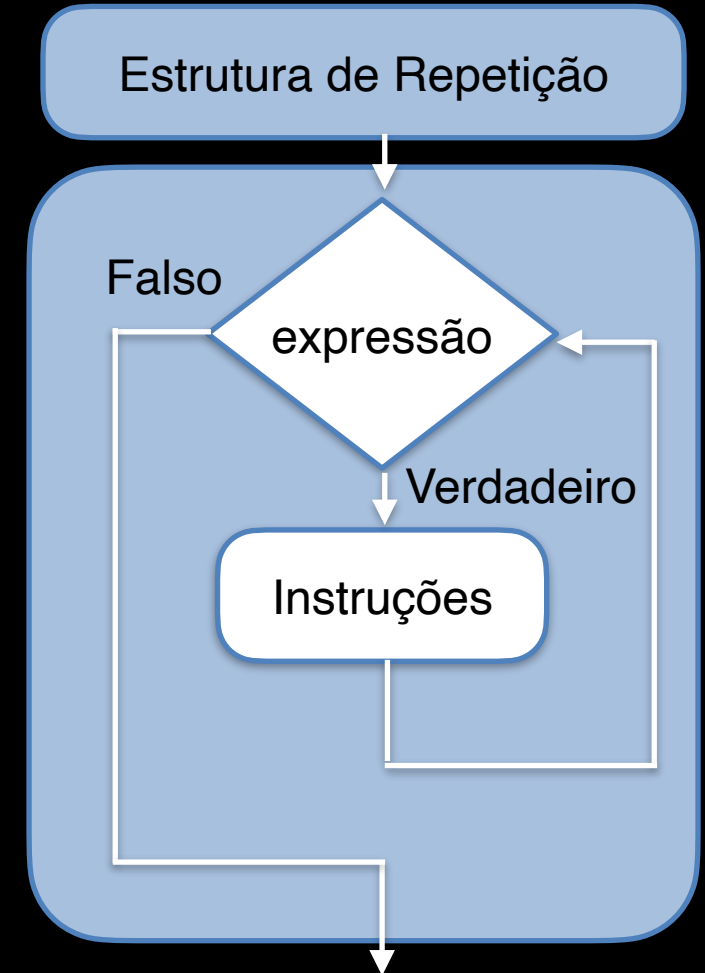
Estrutura Sequencial



Estrutura de Seleção



Estrutura de Repetição



Nesta aula iremos nos aprofundar nas estruturas de repetição e veremos como **encerrar um *loop*** por meio do comando ***break*** e como interromper apenas uma rodada com o comando ***continue***

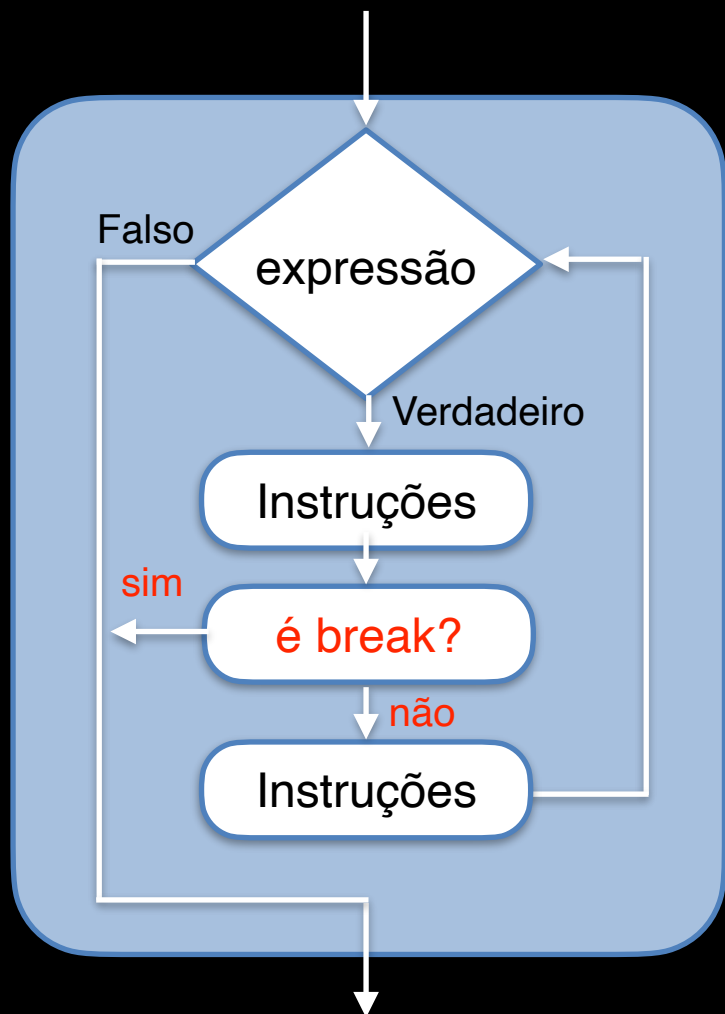


Comando *break*

A execução de um comando *break* encerra o *loop* mais interno em que está contido. Normalmente este comando está condicionado a um comando *if*

Representação do comando break em fluxograma

simplificada



Se o comando **break** for executado, o laço é encerrado e o fluxo de execução será direcionado para a próxima instrução fora dele.

É comum que comando **break** esteja dentro de uma estrutura de seleção, sendo executado se uma condição específica for satisfeita

Exemplo 1

Variável contadora

```
executa = input('Executar o bloco: '))
contador = 0
while executa == 'sim':
    contador += 1
    executa = input('Executar o bloco: ')
print(f'0 bloco foi executado {contador} vezes')
```

Variável contadora

```
contador = 10
while contador > 0:
    print(contador)
    contador -= 1
print('Fogo!')
```

Exercício

Crie um programa que peça letras como entrada, uma por vez, até que seja lida a letra 'x', ao final, o programa deve exibir a quantidade de letras lidas sem contabilizar 'x'.

Lembre-se: Python diferencia letras maiúsculas de minúsculas.



Variável acumuladora



Variável acumuladora é utilizada para **acumular** valores que, em geral, não são constantes, ou seja, o incremento ou decremento pode ser variável.

Exercício

Crie um programa que receba como entrada os preços de itens comprados em um supermercado por um cliente, ao final, o programa deverá exibir o total da compra. Para informar que não há mais itens a serem comprados, o cliente deve digitar o valor -1.

Solução válida

```
total = 0
preco = float(input('Preço do item: '))
while preco != -1:
    total += preco
    preco = float(input('Preço do item: '))

print(f'Total da compra: R$ {total:.2f}')
```

Exercício

Crie um programa que receba como entrada o crédito e depois o preço de itens comprados por esse cliente. O programa deverá parar de solicitar novos preços quando o crédito disponível for insuficiente para pagar por um deles. Ao final exiba o total da compra e o crédito restante

Solução válida

```
credito = float(input('Seu crédito: '))
total = 0 #variável acumuladora
preco = float(input('Preço do item: '))
while credito >= preco:
    total += preco
    credito -= preco
    preco = float(input('Preço do item: '))
print(f'Total da compra: R$ {total:.2f}')
print(f'Crédito restante: R$ {credito:.2f}')
```



Variável *flag* booleana

Variável *flag* sinaliza com um valor booleano se o laço deve ou não ser encerrado

Solução válida

```
total = 0
quero_comprar = True
while quero_comprar:
    preco = float(input('Preço: '))
    total += preco
    opcao = input('Continuar comprando (s/n)? ')
    if opcao != 's':
        quero_comprar = False

print(f'Total da compra: R$ {total:.2f}')
```




Laço infinito



Laço infinito é uma estrutura de repetição em que a condição associada ao laço é sempre **verdadeira**

Exemplos de laço infinito

```
n = 0
```

```
while n <= 10:  
    print(n)
```

```
print('Fora do while')
```

```
n = 0
```

```
while n >= 0:  
    print(n)  
    n += 2
```

```
print('Fora do while')
```

Combinação de estruturas de controle de fluxo



Exemplo

```
n = 10
cont = 1
while cont <= 10:
    if (cont % 2 == 0):
        print(f'{cont} é par')
    else:
        print(f'{cont} é ímpar')
    cont+=1
```


Obrigado e até a próxima aula!