N_ALG PROG_A3 - Texto de apoio

Site: EAD Mackenzie
Tema: ALGORITMOS E PROGRAMAÇÃO I {TURMA 01B} 2021/2
Livro: N_ALG PROG_A3 - Texto de apoio

Impresso por: ANDRE SOUZA OCLECIANO .
Data: terça, 28 set 2021, 20:09

Índice

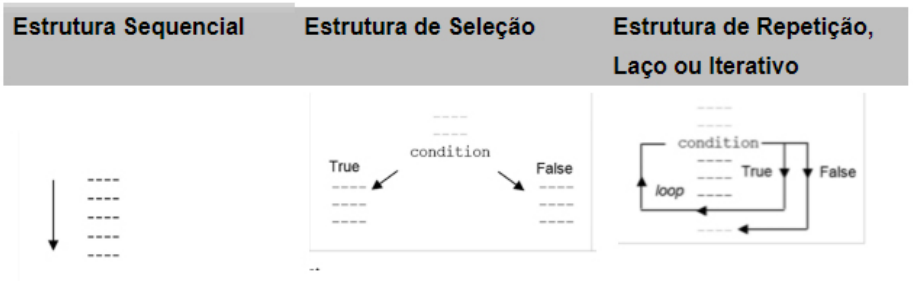
1. ESTRUTURA DE DECISÃO SIMPLES E COMPOSTA

O Que é uma Estrutura de Controle?

A ordem em que as instruções de um programa são executadas é chamada de **controle de fluxo**.

Existem três formas fundamentais de controlar o fluxo de um programa: estrutura de controle sequencial, estrutura de controle de seleção ou condicional e estrutura de controle de repetição, iterativo ou laço.

Figura 1 – Formas de controle do fluxo de um programa



Fonte: DIERBACH (2012)

A **estrutura de controle sequencial**, caracterizada pelos problemas visto até agora, é uma forma implícita de controle, em que as instruções são executadas na ordem em que são escritas.

Conheceremos o segundo tipo de controle de fluxo, a estrutura de seleção ou condicional. Porém, antes, saberemos o que é uma expressão lógica.

2. EXPRESSÕES LÓGICAS

Estudaremos dois operadores usados em **expressões lógicas** (booleanas): **operadores relacionais** e **operadores lógicos**.

Uma **expressão lógica** é uma expressão que resulta em um valor lógico **True** (verdadeiro) ou **False** (falso).

Operadores Relacionais

Os **operadores relacionais** são usados em comparações entre valores. O resultado da comparação será um valor lógico. São eles:

Operadores		Exemplo	Resultado
==	igual a	10 == 10	True
!=	diferente	10 != 10	False
<	menor que	10 < 20	True
>	maior que	“Alan” > “Brenda”	False
<=	menor ou igual a	10 <= 10	True
>=	maior ou igual a	“A” >= “D”	False

Operadores Lógicos

Os **operadores lógicos** podem ser usados para construir expressões lógicas mais complexas combinando comparações. São eles: and (e), or (ou), not (não).

- O operador lógico and resulta verdadeiro somente quando seus operandos forem verdadeiros.
- O operador lógico or resulta verdadeiro quando, pelo menos, um de seus operandos for verdadeiro.
- O operador lógico not inverte o valor do operando.

Tabela verdade dos operadores lógicos:

x	y		x and y	x or y	not x
False	False		False	False	True
True	False		False	True	False
False	True		False	True	
True	True		True	True	

É preciso ser cauteloso ao usar operadores lógicos. Por exemplo, na matemática, para indicar que um valor está dentro de um determinado intervalo escrevemos:

1 <= num <= 10

No entanto, na maioria das linguagens de programação, essa expressão não faz sentido. Para entender por que, vamos assumir que num tem o valor 15.

1 <= num <= 10 ® 1 <= 15 <= 10 ® True <= 10 ® ???

Não faz sentido verificar se **True** é menor ou igual a **10**. A maneira correta de escrever a expressão deve usar o operador lógico **and**

1 <= num and num <= 10

ou

num >= 1 and num <= 10

Precedência dos Operadores

A precedência dos operadores relacionais e lógicos, bem como a associatividade dos operadores, é dada conforme segue a tabela:

Operador	Associatividade
<, >, <=, >=, !=, ==	Esquerda para direita
not	Esquerda para direita
and	Esquerda para direita
or	Esquerda para direita

Uma vez que expressões lógicas podem conter também operadores aritméticos, vistos na aula anterior, podemos estender para a tabela seguinte:

Operador	Associatividade
**	Direita para esquerda
- (negação)	Esquerda para direita
* / // e %	Esquerda para direita
+ e – (substração)	Esquerda para direita
<, >, <=, >=, !=, ==	Esquerda para direita
not	Esquerda para direita
and	Esquerda para direita
or	Esquerda para direita

3. ESTRUTURAS DE SELEÇÃO OU CONDICIONAL

A partir de agora, trabalharemos com as **estruturas condicionais** que permitem que o programa execute diferentes sequências de instruções em diferentes casos, dependendo da avaliação de uma expressão lógica.

Uma **expressão lógica** é uma expressão cujos operadores são lógicos e/ou relacionais e cujos operandos são relações e/ou variáveis do tipo lógico.

Em programação, o uso de condições para permitir a escolha de executar ou não um trecho de programa é muito utilizado, principalmente quando precisamos incluir no programa **condições de controle**, para evitar situações não permitidas que podem resultar em erros. Por exemplo, para evitar divisões por zero.

A estrutura lógica que permite que o fluxo de execução de um algoritmo possa sofrer desvios é conhecida como **estrutura condicional ou de seleção**.

Quando temos apenas um bloco especial de comando ou instruções apenas para quando a expressão lógica for verdadeira, esse tipo de estrutura chama-se **estrutura de condicional simples**.

4. ESTRUTURA CONDICIONAL SIMPLES

Sintaxe:

```
if condição:
    instrução(ões)_verdadeiro
```

Exemplo:

```
if nota >= 6.0:
    print(“Aluno aprovado”)
if nota < 6.0:
    print(“Aluno reprovado”)
```

O texto “**Aluno aprovado**” só será exibido se a condição **nota >= 6.0** for **verdadeira**. Caso contrário, o controle passa para a próxima instrução que tem outra condição **nota < 6.0** a ser avaliada; se esse resultado for **verdadeiro**, então será exibido o texto “**Aluno reprovado**”.

Uma característica singular da linguagem Python é que a quantidade de recuo (**indentação**) está associada a um bloco de instruções.
Essa **indentação** é obrigatória para se definir qual instrução ou quais instruções devem ser executadas quando o resultado lógico da expressão for verdadeiro.

Indentação válida	Indentação inválida

if condição:
  instrução
  instrução
  instrução
	if condição:
	  instrução
	  instrução
	  instrução

Analisando o exemplo anterior, podemos perceber que, para escrever **apenas uma** das mensagens, foram necessárias **duas condições**.

Existe outra estrutura, chamada de **estrutura condicional composta**, em que temos uma única condição e dois caminhos que podem ser seguidos – o caminho do resultado verdadeiro e o caminho do resultado falso.

5. ESTRUTURA CONDICIONAL COMPOSTA

Sintaxe:

```
if condição:
    instrução(ões)_verdadeiro
else:
    instrução(ões)_falso
```

Exemplo:

```
if nota >= 6.0:
    print(“Aluno aprovado”)
else:
    print(“Aluno reprovado”)
```

Nesse caso, será avaliada a condição **nota >= 6.0** e, se o resultado for **verdadeiro**, então será apresentado o texto “**Aluno aprovado”; caso contrário (else)**, será apresentado o texto “**Aluno reprovado**”.

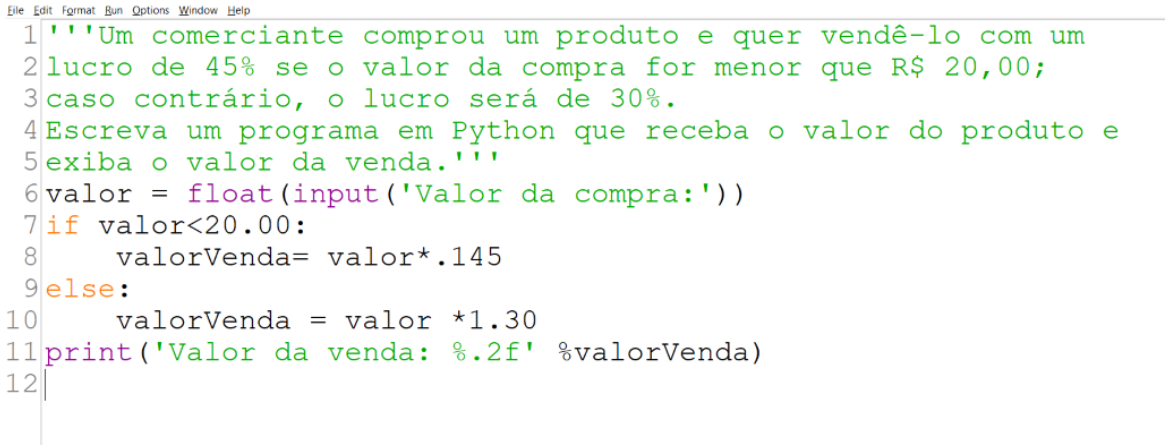
Como na estrutura condicional simples, instrução(ões)_verdadeiro e instrução(ões)_falso devem ser indentadas adequadamente.

Indentações válidas		**Indentações inválidas**	

if condição:
  instrução
  instrução
else:
  instrução
  instrução
	if condição:
	  instrução
	  instrução
	else:
	  instrução
	  instrução
		if condição:
		  instrução
		  instrução
		else:
		  instrução
		  instrução
			if condição:
			  instrução
			  instrução
			else:
			instrução
			instrução

Exemplos:

Um comerciante comprou um produto e quer vendê-lo com um lucro de 45% se o valor da compra for menor que R$ 20,00; caso contrário, o lucro será de 30%. Escreva um programa em Python que receba o valor do produto e exiba o valor da venda.



Escreva um programa que leia um número inteiro de três dígitos e imprima se o algarismo da dezena é par ou ímpar.

File Edit Format Run Options Window Help

```
'''Escreva um programa que leia um número inteiro de 3 dígitos e
imprima se o algarismo da dezena é par ou ímpar. '''
print('Digite um número inteiro de 3 dígitos')
num = int(input())
if num>999 or num<100:
    print('Número fora do intervalo solicitado.')
else:
    dezena = num%100//10
    if dezena%2==0:
        print('dezena par')
    else:
        print('dezena impar')

```

6. REFERÊNCIAS

DIERBACH, C. *Introduction to Computer Science Using Python*: A Computational Problem Solving Focus. New York: Wiley, 2012.

MENEZES, N. N. C. *Introdução à Programação com Python*: algoritmos e lógica de programação para iniciantes. São Paulo: Novatec, 2014.