

Universidade Federal de Viçosa
Departamento de Informática
Centro de Ciências Exatas e Tecnológicas



INF 100 – Introdução à Programação

Comando de Seleção (ou condicionais)

se ... senão ...

if ... else ...

Exercício

- Faça um programa que leia do teclado os valores a , b e c de uma equação do 2º grau
$$ax^2 + bx + c = 0$$
e depois calcule e escreva na tela as raízes da equação.
- Exercício: analise o problema e monte um algoritmo contendo a solução inicial. Suponha que $a \neq 0$ e que a equação possui raízes reais.



Exercício

leia a, b, c

$$\Delta = b^2 - 4ac$$

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

escreva x1

escreva x2



Exercício

- Agora traduza o algoritmo para a linguagem Python. No lugar de:

$\sqrt{\quad}$

você pode usar:

`delta ** 0.5`



Programa em Python

```
print('Equação do 2o grau  $ax^2 + bx + c = 0$ ')
print('Entre com os valores:')
a = float (input('a = '))
b = float (input('b = '))
c = float (input('c = '))
delta = b*b - 4*a*c
x1 = (-b + delta**0.5) / (2 * a)
x2 = (-b - delta**0.5) / (2 * a)
print('x1 = ', x1 )
print('x2 = ', x2 )
```



Exemplo de Programa em Python

Mas o que fazer nas seguintes situações:

- $a = 0$ (não é equação do 2º grau)
- $\Delta < 0$ (equação não possui raízes reais)
- ?



Comandos Condicionais

Nos algoritmos, podemos escrever:

```
se condição:  
  comando1  
  comando2  
  ...
```

o que faz com que os comandos só sejam executados se a condição dada for Verdadeira.



Comandos Condicionais

Nos algoritmos, podemos escrever:

se condição:
comando₁
comando₂

...



Outros comandos pode vir depois do comando **se**. Como podemos definir quais comandos dependem da condição?

Opções:

- usando **indentação**;
- usando terminadores explícitos.



Comandos Condicionais

Nos algoritmos, podemos escrever:

```
se condição:  
  comando1  
  comando2  
  ...
```

OU:

```
se condição:  
  comando1  
  comando2  
  ...  
fim_se
```

Outros comandos pode vir depois do comando **se**. Como podemos definir quais comandos dependem da condição?

Opções:

- usando indentação;
- **usando terminadores explícitos.**



Comandos Condicionais

- Compare os seguintes trechos de algoritmo:

se condição:
comando₁
comando₂
comando₃

se condição:
comando₁
comando₂
comando₃

- Considerando que a indentação é usada para definir a dependência com o comando condicional, então os dois trechos diferem na semântica – qual é a diferença???



Comandos Condicionais

- Compare os seguintes trechos de algoritmo:

se *condição*:
 → *comando*₁
 *comando*₂
 *comando*₃

se *condição*:
 *comando*₁
 *comando*₂
 *comando*₃


- No primeiro caso:
 - 2 comandos dependem da *condição*.
 - *comando*₃ sempre será executado, independente da *condição*.



Comandos Condicionais

- Compare os seguintes trechos de algoritmo:

se condição:
comando₁
comando₂
comando₃

se condição:
 *comando₁*
comando₂
comando₃

- No segundo caso:
 - *comando₁* depende da *condição*.
 - *comando₂* e *comando₃* sempre serão executados, independentes da *condição*.



Comandos Condicionais

- Abaixo usamos um terminador explícito 'fim_se' para definir a dependência dos comandos condicionais, em vez da indentação.

```
se condição:  
    comando1  
    comando2  
fim_se  
comando3
```

```
se condição:  
    comando1  
fim_se  
comando2  
comando3
```



Comandos Condicionais

- Ao escrever os algoritmos, você pode usar a indentação ou os terminadores explícitos para definir a dependência dos comandos.
- Algumas linguagens de programação usam a indentação (e.g. Python) e outras usam terminadores explícitos (C, C++, Java...).



Comandos Condicionais

Uma versão estendida do comando é:

se condição:
 comando₁
senão:
 comando₂

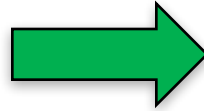
o que significa que, se a condição for verdadeira, ***comando₁*** será executado; caso contrário, ***comando₂*** será executado.



Sintaxe em Python

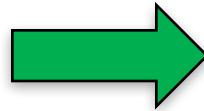
Algoritmo

se condição:
comando₁



if condição:
comando₁

se condição:
comando₁
senão:
comando₂



if condição:
comando₁
else:
comando₂



Comandos Condicionais

- Operadores lógicos de comparação e seus equivalentes em Python

Operador	Equivalente em Python
=	==
≠	!=
>	>
<	<
≥	>=
≤	<=



Traduzindo para Python

```
se a ≠ 0:  
    b = 1  
    c = 2  
fim_se  
d = 3
```



```
if a != 0:  
    b = 1  
    c = 2  
d = 3
```

```
se a ≥ 0:  
    b = 1  
senão:  
    c = 2  
    d = 3  
fim_se
```



```
if a >= 0:  
    b = 1  
else:  
    c = 2  
    d = 3
```



Exercício

- O que será escrito na tela pelo trecho de código ao lado?

```
a=2
b=1
if a < b:
    a=3
else:
    b=0
    a=4
print( a, b )
a=1
b=2
if a < b:
    a=3
else:
    b=0
a=4
print( a, b )
```



Exercício

leia a, b, c

$$\Delta = b^2 - 4ac$$

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

escreva x1

escreva x2

Estender esse algoritmo para as seguintes situações:

- $a = 0$ (não é equação do 2º grau)
- $\Delta < 0$ (equação não possui raízes reais)



$a = 0$ (não é equação do 2º grau)

leia a, b, c

se a = 0:

 escreva 'Não é equação do 2º grau'

senão:

$$\Delta = b^2 - 4ac$$

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

escreva x1

escreva x2



$a = 0$ (não é equação do 2º grau)

leia a, b, c

se **a = 0:**

↳ escreva 'Não é equação do 2º grau'

senão:

↳
$$\Delta = b^2 - 4ac$$

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

escreva x1

escreva x2



$a = 0$ (não é equação do 2º grau)

```
print('Equação do 2o grau  $ax^2 + bx + c = 0$ ')
print('Entre com os valores:')
a = float(input('a = '))
b = float(input('b = '))
c = float(input('c = '))
if a == 0:
    print('Não é equação do 2o grau.')
else:
    delta = b*b - 4*a*c;
    x1 = (-b + delta**0.5) / (2 * a)
    x2 = (-b - delta**0.5) / (2 * a)
    print('x1 = ', x1)
    print('x2 = ', x2)
```



Exemplo de Programa em Python

Mas o que fazer nas seguintes situações:

- $a = 0$ (não é equação do 2º grau) **resolvido!**
- $\Delta < 0$ (equação não possui raízes reais)

Outras situações:

- $\Delta = 0$ (equação possui uma única raiz real)
- $\Delta > 0$ (equação possui duas raízes reais)




$\Delta < 0$ (equação não possui raízes reais)

```
print('Equação do 2o grau ax² + bx + c = 0')
print('Entre com os valores:')
a = float (input('a = '))
b = float (input('b = '))
c = float (input('c = '))
if a == 0:
    print('Não é equação do 2o grau.')
else:
    delta = b*b - 4*a*c
    if delta < 0:
        print('Nenhuma raiz real.')
    else:
        x1 = (-b + delta**0.5) / (2 * a)
        x2 = (-b - delta**0.5) / (2 * a)
        print('x1 = ', x1 )
        print('x2 = ', x2 )
```



$\Delta < 0$ (equação não possui raízes reais)

```
print('Equação do 2o grau ax² + bx + c = 0')
print('Entre com os valores:')
a = float (input('a = '))
b = float (input('b = '))
c = float (input('c = '))
if a == 0:
    print('Não é equação do 2o grau.')
else:
    delta = b*b - 4*a*c
    if delta < 0:
        print('Nenhuma raiz real.')
    else:
        x1 = (-b + delta**0.5) / (2 * a)
        x2 = (-b - delta**0.5) / (2 * a)
        print('x1 = ', x1 )
        print('x2 = ', x2 )
```



Note as indentações

$\Delta < 0$ (equação não possui raízes reais)

```
print('Equação do 2o grau ax² + bx + c = 0')
print('Entre com os valores:')
a = float (input('a = '))
b = float (input('b = '))
c = float (input('c = '))
if a == 0:
    print('Não é equação do 2o grau.')
else:
    delta = b*b - 4*a*c
    if delta < 0:
        print('Nenhuma raiz real.')
    else:
        x1 = (-b + delta**0.5) / (2 * a)
        x2 = (-b - delta**0.5) / (2 * a)
        print('x1 = ', x1 )
        print('x2 = ', x2 )
```



Observações Importantes

- Comandos `if...else` aninhados podem criar situações que parecem ambíguas. O uso de comentários ajuda muito no entendimento do código.
- Um erro comum é usar `=` em vez de `==` ao comparar dois valores. O compilador irá indicar erro de sintaxe nesse caso.



Exercício

- Faça um programa que leia a nota final de um aluno, e escreva na tela uma mensagem dizendo se ele passou direto, se ficou de Prova Final, ou se foi reprovado (de acordo com as normas da UFV).



Uma possível solução...

leia nota

se nota < 40:

escreva 'Reprovado.'

senão

se nota < 60:

escreva 'Prova Final.'

senão

escreva 'Aprovado.'



Outra possível solução...

leia nota

se nota \geq 60:

escreva 'Aprovado.'

senão

se nota \geq 40:

escreva 'Prova Final.'

senão

escreva 'Reprovado.'



Traduzindo para Python

```
nota = float (input('Nota final: '))
if nota >= 60:
    print('Aprovado.')
else:
    if nota >= 40:
        print('Prova Final.')
    else:
        print('Reprovado.')
```



Exercício

- Estenda o programa para que ele verifique se a nota digitada está dentro do intervalo válido [0..100].



Exercício

```
leia nota
se 0 ≤ nota ≤ 100:
    se nota ≥ 60:
        escreva 'Aprovado.'
    senão
        se nota ≥ 40:
            escreva 'Prova Final.'
        senão
            escreva 'Reprovado.'
senão
    escreva 'Nota inválida.'
```



Traduzindo para Python

```
nota = float (input('Nota final: '))
if 0 <= nota <= 100:
    if nota >= 60:
        print('Aprovado.')
    else:
        if nota >= 40:
            print('Prova Final.')
        else:
            print('Reprovado.')
else:
    print('Nota inválida.')
```



Exercício

- Mas...

A expressão:	É equivalente a...
$0 \leq \text{nota} \leq 100$	$\text{nota} \geq 0 \text{ e } \text{nota} \leq 100$

- ... o que nos dá outra possível solução:



Exercício

leia nota

se nota ≥ 0 e nota ≤ 100 :

se nota ≥ 60 :

escreva 'Aprovado.'

senão

se nota ≥ 40 :

escreva 'Prova Final.'

senão

escreva 'Reprovado.'

senão

escreva 'Nota inválida.'



Estruturas de Seleção

- Operadores lógicos de conexão (e, ou, não)

Operador	Equivalente em Python
e	and
ou	or
não	not



Traduzindo para Python

```
nota = float (input('Nota final: '))
if nota >= 0 and nota <= 100:
    if nota >= 60:
        print('Aprovado.')
    else:
        if nota >= 40:
            print('Prova Final.')
        else:
            print('Reprovado.')
else:
    print('Nota inválida.')
```



Outra solução

leia nota

se nota < 0 ou nota > 100:

escreva 'Nota inválida.'

senão

se nota \geq 60:

escreva 'Aprovado.'

senão

se nota \geq 40:

escreva 'Prova Final.'

senão

escreva 'Reprovado.'



Traduzindo para Python

```
nota = float (input('Nota final: '))
if nota < 0 or nota > 100:
    print('Nota inválida.')
else:
    if nota >= 60:
        print('Aprovado.')
    else:
        if nota >= 40:
            print('Prova Final.')
        else:
            print('Reprovado.')
```



Juntando else + if: comando **elif**

```
nota = float (input('Nota final: '))
if nota < 0 or nota > 100:
    print('Nota inválida.')
elif nota >= 60:
    print('Aprovado.')
elif nota >= 40:
    print('Prova Final.')
else:
    print('Reprovado.')
```



Considerando também de faltas...

```
nota = float (input('Nota final: '))
if nota < 0 or nota > 100:
    print('Nota inválida.')
else:
    faltas = int (input('Nº faltas: '))
    pctFaltas = faltas/30 # só aulas teóricas
    if nota < 40 or pctFaltas > 0.25:
        print('Reprovado.')
    elif nota < 60:
        print('Prova Final.')
    else:
        print('Aprovado.')
```



Operadores Lógicos

- Operador **e** (*and*)
- a e b resultará em verdadeiro somente se a for verdadeiro e b também for verdadeiro.

a	b	a and b
V	V	V
V	F	F
F	V	F
F	F	F



Operadores Lógicos

- O operador **and** pode ser visto como uma multiplicação, onde cada valor verdadeiro é codificado como sendo 1, e cada valor falso é codificado como sendo 0.

<i>a</i>	<i>b</i>	<i>a and b</i>
1	1	1
1	0	0
0	1	0
0	0	0



Operadores Lógicos

- Operador **ou** (*or*)
- a ou b resultará em verdadeiro se qualquer dos valores a ou b for verdadeiro. Ou seja, o resultado só será falso se todos os operandos forem iguais a falso.

a	b	a or b
V	V	V
V	F	V
F	V	V
F	F	F



Operadores Lógicos

- O operador **or** pode ser visto como uma soma. Neste caso, qualquer soma diferente de zero resulta em verdadeiro.

<i>a</i>	<i>b</i>	<i>a or b</i>
1	1	1
1	0	1
0	1	1
0	0	0



Operadores Lógicos

- Operador **não** (*not*)
- Este operador é unário, ou seja, ele atua em um operando de cada vez

a	$\text{not } a$
V	F
F	V



Exemplo

```
if idade < 18:  
    print('Menor de idade.')  
else:  
    print('Maior de idade.')
```

```
if not (idade >= 18)  
    print('Menor de idade.')  
else:  
    print('Maior de idade.')
```



Propriedades de expressões lógicas

not (a or b) é equivalente a not a and not b
not (a and b) é equivalente a not a or not b

Exemplo de expressões equivalentes:

not (nota >= 0 and nota <= 100)

not (nota >= 0) or not (nota <= 100)

nota < 0 or nota > 100



Expressões Lógicas

- O valor de um teste lógico falso/verdadeiro pode ser armazenado em uma variável:

```
x = 1
y = 2
a = y < 5
b = x == y
c = (x < y) and a
print('2 < 5:', a )
print('x == y:', b )
print('(x < y) and a:', c )
```



Precedência dos Operadores

Prioridade	Operador(es) e Comando =	Exemplo
1	**	x ** 3
2	- (unário)	-x
3	* / // %	x / y
4	+ -	x - y
5	< <= > >= == !=	x < y
6	not	not x < y
7	and	x > 10 and y < 0
8	or	x > 10 or y < 0
9	=	x = 2



Exemplos

Expressão	Resultado
$(2 \geq 1) \text{ or } (5 \neq 4)$	
$\text{not } (2 < 4)$	
$(4 == 2 + 2) \text{ and } (3 > 8)$	
$\text{not } ((5 > 9) \text{ or } (3 == 1 + 2)) \text{ and } (2 \leq 7)$	
$\text{not } (5 > 9) \text{ or } (3 == 1 + 2) \text{ and } (2 \leq 7)$	
$(8 \% 2 == 4) \text{ or } (3 / 2 > 1.6)$	



Exemplos

Expressão	Resultado
$(2 \geq 1) \text{ or } (5 \neq 4)$	True
$\text{not } (2 < 4)$	False
$(4 == 2 + 2) \text{ and } (3 > 8)$	False
$\text{not } ((5 > 9) \text{ or } (3 == 1 + 2)) \text{ and } (2 \leq 7)$	False
$\text{not } (5 > 9) \text{ or } (3 == 1 + 2) \text{ and } (2 \leq 7)$	True
$(8 \% 2 == 4) \text{ or } (3 / 2 > 1.6)$	False



Exercício "Conceitos"

- Faça um programa que leia a nota final (inteira) de um aluno, e escreva na tela uma mensagem contendo a situação do aluno de acordo com a seguinte regra:

Nota	Situação
Entre 90 e 100	Conceito 'A'
Entre 75 e 89	Conceito 'B'
Entre 60 e 74	Conceito 'C'
Menor que 60	Reprovado



Exercício "IMC"

- O Índice de Massa Corporal (IMC) é um método aproximado para se medir o grau de obesidade de uma pessoa. É calculado como sendo a massa da pessoa (em kg) dividido pelo quadrado da altura da mesma (em metros).
- Faça um programa que peça ao usuário seus dados de peso e altura, calcule seu IMC e exiba em tela o resultado, junto com a classificação de acordo com a tabela a seguir.



Exercício "IMC"

IMC	Classificação
Abaixo de 18,5	Abaixo do peso ideal
Entre 18,5 e 24,9	Peso ideal
Entre 25,0 e 29,9	Sobrepeso
30,0 ou mais	Obeso



Exercício "Bafômetro"

- Bafômetro (ou etilômetro): aparelho que permite determinar a concentração de bebida alcoólica em uma pessoa, analisando o ar exalado dos pulmões.
- Taxa de alcoolemia: quantidade de álcool existente no sangue de um indivíduo, em determinado momento, expressa em gramas de álcool por litro de sangue (g/l). Ex.: uma alcoolemia de 0,5 g/l é o mesmo que dizer que existem 0,5 g de álcool por litro de sangue.



Exercício "Bafômetro"

- Alcoolemia (valor aproximado):

Taxa de alcoolemia = $\frac{\text{Álcool consumido (g)}}{(\text{Peso corporal (kg)} \times \text{coeficiente})}$

0,7 nos homens em jejum

0,6 nas mulheres em jejum

1,1 durante as refeições (ambos os sexos)

Obs.: Limite até Jan/2013: 0,2 g/l; Hoje: **0,0 g/l**



Exercício "Bafômetro"

- Desenvolva um programa que permita calcular e exibir na tela a taxa de alcoolemia do sangue de pessoas de acordo com o número de copos de cerveja ingeridos por uma pessoa.
- Considere que cada copo pequeno de cerveja (150 ml) possui 4,8 gramas de álcool.



Exercício "xyz"

- Suponha que x , y e z são valores numéricos (reais), e m é a média desses valores.
- Escreva um programa em Python para calcular n , o número de valores maiores que m .
- Exemplo: Suponha que $x = 5$, $y = 7$, $z = 18$. Então $m = 10$. Nesse caso, somente z é maior que m , e temos, portanto, $n = 1$.

