Comando de seleção

SSC0301

Prof. Márcio Delamaro

Problema

Escreva um programa para computar uma raiz da função $f(x) = x^3 - x^2 - 13x + 8$ usando 10 iterações do método de Newton-Raphson. Use um comando de entrada para ler o chute inicial.

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
f1 = lambda x: 3 * x ** 2 - 2 * x - 13
x0 = float(input('Digite o chute inicial: '))
```

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
f1 = lambda x: 3 * x ** 2 - 2 * x - 13
x0 = float(input('Digite o chute inicial: '))
x1 = x0 - f(x0)/f1(x0)
```

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
f1 = lambda x: 3 * x ** 2 - 2 * x - 13
x0 = float(input('Digite o chute inicial: '))
x1 = x0 - f(x0)/f1(x0)
x2 = x1 - f(x1)/f1(x1)
```

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
f1 = lambda x: 3 * x ** 2 - 2 * x - 13
x0 = float(input('Digite o chute inicial: '))
x1 = x0 - f(x0)/f1(x0)
x2 = x1 - f(x1)/f1(x1)
...
x10 = x9 - f(x9)/f1(x9)
erro = abs(x10-x9)
print("Solução: {:.7f}. Erro: {:.7f}".format(x10,erro))
```

ICMC USP SÃO CARLOS SSC

Problema

- Escreva um programa para computar uma raiz da função $f(x) = x^3 - x^2 - 13x + 8$ usando 10 iterações do método da bisseção.

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
a = -4
b = -3
```

Solução

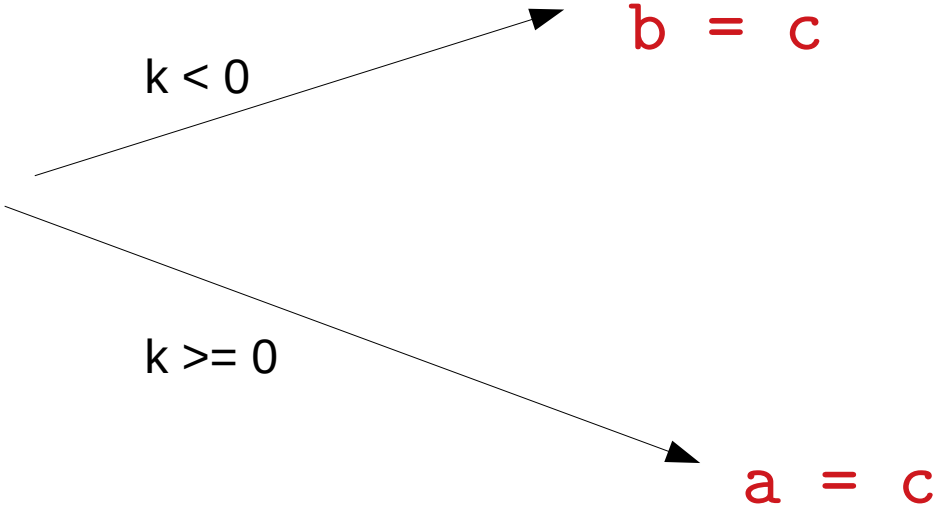
```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
a = -4
b = -3
c = (a+b)/2
k = f(a) * f(c)
```

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
a = -4
b = -3
c = (a+b)/2
k = f(a) * f(c)
????????
```

Solução

```
f = lambda x: x ** 3 - x ** 2 - 13 * x + 8
a = -4
b = -3
c = (a+b)/2
k = f(a) * f(c)
????????
```



Tipo booleano

- Representa um valor lógico, verdadeiro ou falso
  - True
  - False
- Usado em expressões/condições
- Guardado em variáveis

Exemplos

```
>>> a = 10.01
>>> a == 10
False
>>> a > 10
True
>>> a != 10
True
>>> 'abc' < 'def'
True
>>> x = a > 10
>>> x
True
>>> y = True
```

ICMC USP SÃO CARLOS SSC

Operadores relacionais

Operador	Significado	Exemplo	Resultado
==	Igual	10 == 10.1 "abc"== "abc"	False True
!=	Diferente	10 != 10.1 "abc"!= "abc"	True False
<	Menor	10.1 < 10 "abc” < "def"	False True
>	Maior	10.1 > 10 "abc” > "def"	True False
<=	Menor ou Igual	10.1 <= 10 "abc” <= "abc"	False True
>=	Maior ou Igual	10.1 >= 10 "abc” >= "def"	True False

Operadores lógicos

- Combinar expressões booleanas

- Valor da variável idade está entre 18 e 60

Operadores lógicos

- Combinar expressões booleanas

- Valor da variável idade está entre 18 e 60
  - idade tem que ser maior do que 18 e menor do que 60

Operadores lógicos

- Combinar expressões booleanas

- Valor da variável idade está entre 18 e 60
  - idade tem que ser maior do que 18 e menor do que 60
- Idade não está no intervalo 18 e 60

Operadores lógicos

- Combinar expressões booleanas
-
- Valor da variável idade está entre 18 e 60
  - idade tem que ser maior do que 18 e menor do que 60
- Idade não está no intervalo 18 e 60
  - idade tem que ser menor que 18 ou maior que 60

Exemplos

```
>>> x = 0.5
>>> x >= 0.0 and x <= 1.0
True
>>> x = 2.0
>>> x >= 0.0 and x <= 1.0
False
>>> x = 0.5
>>> x < 0.0 or x > 1.0
False
>>> x = 2.0
>>> x < 0.0 or x > 1.0
True
>>>
```

ICMC USP SÃO CARLOS SSC

Exemplos

```
>>> x = 0.5
>>> y = 1.5
>>> x < 0.0 or y > 1.0
True
>>> x == 0.0 or x != y
True
>>> not (x >= 0.0 and x <= 1.0)
False
>>> x = 2.0
>>> not (x >= 0.0 and x <= 1.0)
True
>>>
```

Tabela verdade

Operação	a	b	Resultado
a and b	False	False	False
	False	True	
	True	False	
	True	True	
a or b	False	False	
	False	True	
	True	False	
	True	True	
not a	False	-	
	True	-	

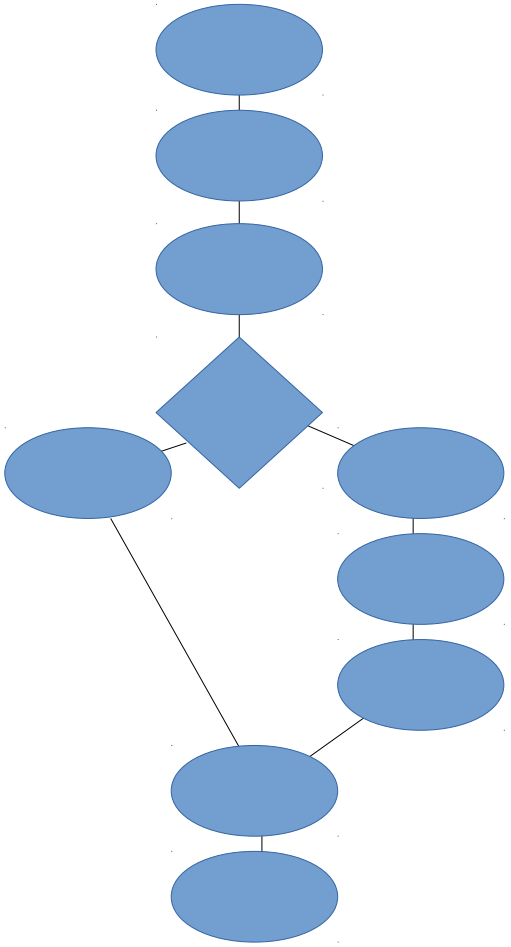
Tabela verdade

Operação	a	b	Resultado
a and b	False	False	False
	False	True	False
	True	False	False
	True	True	True
a or b	False	False	False
	False	True	True
	True	False	True
	True	True	True
not a	False	-	True
	True	-	False

Execução sequencial

- Os comandos no nosso programa são executados sequencialmente
- Ver o exemplo do bhaskara
  - O que acontece se delta é negativo?
- Como fazer eu quizer tratar esse problema?
  - Se o delta for negativo eu aviso o usuário

Desvio na execução



Comando if/else

```
if < expressão booleana > :
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
else :
    comando executado se expressão for falsa
    comando executado se expressão for falsa
```

Comando if/else

```
if < expressão booleana > :
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
else :
    comando executado se expressão for falsa
    comando executado se expressão for falsa
```

Comando if/else

```
if < expressão booleana > :
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
else :
    comando executado se expressão for falsa
    comando executado se expressão for falsa
```

Comando if/else

```
if < expressão booleana > :
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
else :
    comando executado se expressão for falsa
    comando executado se expressão for falsa
```

Bhaskara

```
delta = b ** 2 - 4 * a * c

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)

print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)


print('Final do programa')
```

Bhaskara

```
.........

delta = b ** 2 - 4 * a * c

# Verfica se delta é negativo
if ??? :

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)
print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)


print('Final do programa')
```

ICMC USP SÃO CARLOS SSC

Bhaskara

```
.........

delta = b ** 2 - 4 * a * c

# Verfica se delta é negativo
if delta < 0 :

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)
print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)


print('Final do programa')
```

Bhaskara

```
……..

delta = b ** 2 - 4 * a * c

# Verfica se delta é negativo
if delta < 0 :
    print(“Essa equação não tem raiz real”)

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)
print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)


print('Final do programa')
```

Bhaskara

```
……..

delta = b ** 2 - 4 * a * c

# Verfica se delta é negativo
if delta < 0 :
    print(“Essa equação não tem raiz real”)
else:

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)
print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)


print('Final do programa')
```

Bhaskara

……

```
delta = b ** 2 - 4 * a * c

# Verfica se delta é negativo
if delta < 0 :
    print(“Essa equação não tem raiz real”)
else:
    x1 = (-b + math.sqrt(delta)) / (2 * a)
    x2 = (-b - math.sqrt(delta)) / (2 * a)
    print("O valor da 1a raiz é ", x1)
    print("O valor da 2a raiz é ", x2)


print('Final do programa')
```

if sem else

- O comando `else` não é obrigatório
- Nesse caso, se a condição for falsa, nenhum comando é executado
- A execução continua depois do `if`

if sem else

```
if < expressão booleana > :
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
else :
    comando executado se expressão for falsa
    comando executado se expressão for falsa
```

if sem else

```
if < expressão booleana > :
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
    comando executado se expressão for verdadeira
```

if sem else

- O comando `else` não é obrigatório
- Nesse caso, se a condição for falsa, nenhum comando é executado
- A execução continua depois do `if`
- `bhaskara` – verificar se a equação é quadrática

if sem else

```
……

a = float(input('Digite o valor de a: '))
b = float(input('Digite o valor de a: '))
c = float(input('Digite o valor de c: '))

#verifica se é equação quadrática
if a == 0:
      ????????
……
```

if sem else

```
……
import sys
a = float(input('Digite o valor de a: '))
b = float(input('Digite o valor de a: '))
c = float(input('Digite o valor de c: '))

#verifica se é equação quadrática
if a == 0:
    print('Essa equação não é quadrática.')
    print('Terminando a execução')
    sys.exit()

delta = b ** 2 - 4 * a * c

……
```

Observação

- O programa só termina porque foi chamada a função `sys.exit()`
- Escreva um programa leia um número *x* digitado pelo usuário e compute a expressão
  - x^2 - $\sqrt{x}$ se o valor de x for não negativo
  - se x for negativo, use na expressão o valor positivo de x

Exemplo if sem else

```
x = float(input("Digite o valor de x: "))
if x < 0:
    x = -x

res = x**2 - math.sqr(x)
print("Resultado: {:.4f}".format(res))
```

Praticar

- Faça um programa que leia o valor da hora de trabalho (em reais) e número de horas trabalhadas no mês, e calcule o valor a ser pago ao funcionário, adicionando 10% sobre o valor calculado. Antes de dar o resultado final, desconte 3% de imposto, se o valor total for menor ou igual a 2800 reais. Se for maior, desconte 5%.

Praticar

```
valor_hora = float(input('Valor da hora trabalhada: '))
horas_mes = float(input('Número de horas trabalhadas: '))

salario = valor_hora * horas_mes * 1.1

if salario <= 2800:
   imposto = salario * 0.03
else:
   imposto = salario * 0.05

print('Valor a receber: R${:.2f}'.format(salario-imposto))
```

Praticar

```
valor_hora = float(input('Valor da hora trabalhada: '))
horas_mes = float(input('Número de horas trabalhadas: '))

salario = valor_hora * horas_mes * 1.1

if salario <= 2800:
    imposto = salario * 0.03
else:
    imposto = salario * 0.05

print('Valor a receber: R${:.2f}'.format(salario-imposto))
```

Praticar

```
valor_hora = float(input('Valor da hora trabalhada: '))
horas_mes = float(input('Número de horas trabalhadas: '))

salario = valor_hora * horas_mes * 1.1

imposto = salario * 0.05
if salario <= 2800:
    imposto = salario * 0.03

print('Valor a receber: R${:.2f}'.format(salario-imposto))
```

Praticar

- Implemente o método de bhaskara, que verifique se a equação é de segundo grau e que verifique se ela tem raízes reais. Não utilize a função `sys.exit()`. Use apenas dois ifs para fazer os testes.

Comandos aninhados

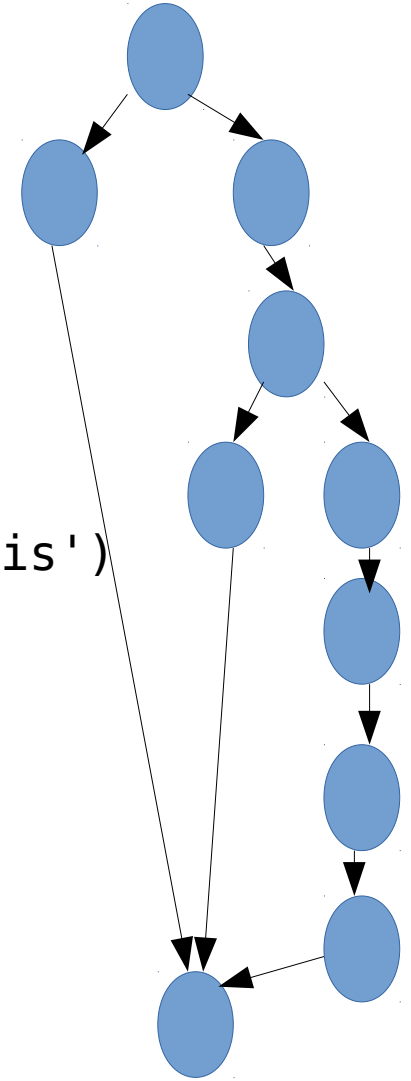
- Dentro de um comando if podemos ter qualquer tipo de comando
- Incluindo outros comandos if
- bhaskara: se a = 0 não é quadratica. Caso contrario, calcula o delta e verifica se tem raízes reais. Se não tiver, avisa. Se tiver, calcula.

Comandos aninhados

```
……
#verifica se é equação quadrática
if a == 0:
    print('Essa equação não é quadrática.')
else:
    delta = b ** 2 - 4 * a * c

    # Verfica se delta é negativo
    if delta < 0:
        print('Essa equação não possui raízes reais')
    else:
        x1 = (-b + math.sqrt(delta)) / (2 * a)
        x2 = (-b - math.sqrt(delta)) / (2 * a)

        print('A 1a raiz é {:.4f}'.format(x1))
        print('A 2a raiz é {:.4f}'.format(x2))

print('Final do programa')
```

Comandos aninhados

```
……
#verifica se é equação quadrática
if a == 0:
    print('Essa equação não é quadrática.')
else:
    delta = b ** 2 - 4 * a * c

    # Verfica se delta é negativo
    if delta < 0:
        print('Essa equação não possui raízes reais')
    else:
        x1 = (-b + math.sqrt(delta)) / (2 * a)
        x2 = (-b - math.sqrt(delta)) / (2 * a)

        print('A 1a raiz é {:.4f}'.format(x1))
        print('A 2a raiz é {:.4f}'.format(x2))

print('Final do programa')
```



Praticar

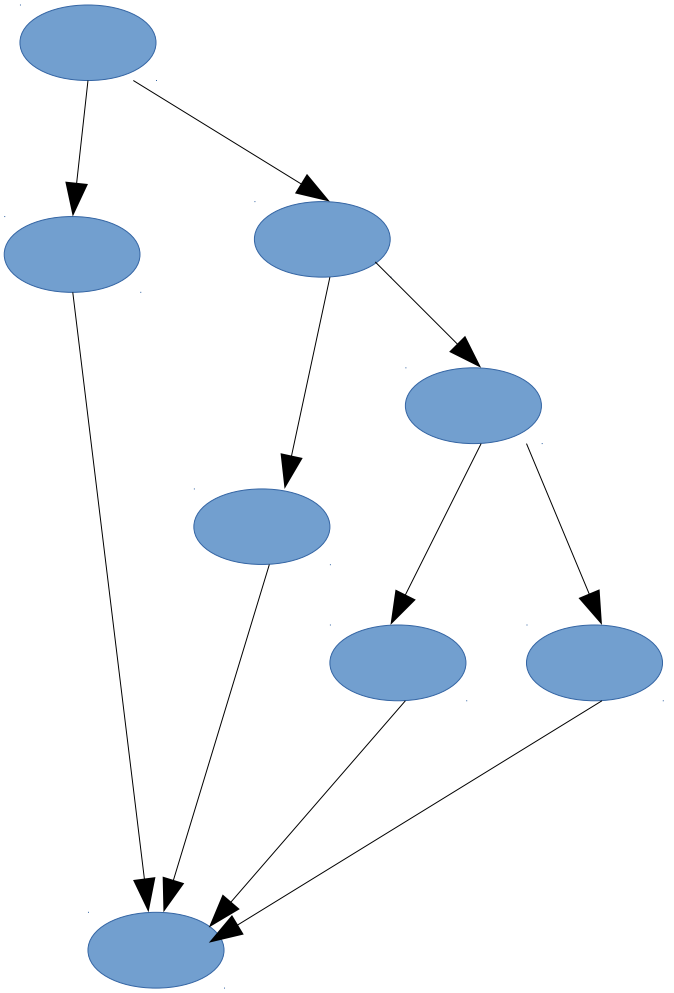
- Escreva um programa que lê um número inteiro e diz se ele corresponde a um ano bissexto ou não. Use os ifs aninhados para fazer isso.
- Se for múltiplo de quatro, mas não pode ser múltiplo de 100, a não ser que seja múltiplo de 400.

Praticar

```
ano = int(input('Digite o ano: '))

if ano % 400 == 0:
    print('É bissexto')
else:
    if ano % 100 == 0:
        print('Não é bissexto')
    else:
        if ano % 4 == 0:
            print('É bissexto')
        else:
            print('Não é bissexto')

print('Fim do programa')
```

Praticar

```
ano = int(input('Digite o ano: '))

if ano % 400 == 0:
    print('É bissexto')
else:
    if ano % 100 == 0:
        print('Não é bissexto')
    else:
        if ano % 4 == 0:
            print('É bissexto')
        else:
            print('Não é bissexto')

print('Fim do programa')
```



if/elif/else

- Quando existe um encadeamento de varias condições, existe uma alternativa
-

```
if < expressão booleana 1> :
    comando executado se expressão 1 for verdadeira
elif < expressão booleana 2>:
    comando executado se expressão 2 for verdadeira
elif < expressão booleana 3> :
    comando executado se expressão 3 for verdadeira
else:
    comando executado se expressões forem falsas
```

if/elif/else

- Quando existe um encadeamento de varias condições, existe uma alternativa
-

```
if < expressão booleana 1> :
    comando executado se expressão 1 for verdadeira
elif < expressão booleana 2>:
    comando executado se expressão 2 for verdadeira
elif < expressão booleana 3> :
    comando executado se expressão 3 for verdadeira
else:
    comando executado se expressões forem falsas
```

Todos têm a mesma indentação. Facilita a visualização.

Praticar

- Reimplemente o programa do ano bissexto usando if/elif/else
- Escreva um programa que lê três números que representam os lados de um triângulo e diga que tipo de triângulo ele é: equilátero, isóceles ou escaleno.

Bissexto de novo

```
ano = int(input('Digite o ano: '))

if ano % 400 == 0:
   print('É bissexto')
elif ano % 100 == 0:
   print('Não é bissexto')
elif ano % 4 == 0:
   print('É bissexto')
else:
   print('Não é bissexto')

print('Fim do programa')
```

Tipo do triângulo

```
a = int(input('Digite o lado 1: '))
b = int(input('Digite o lado 2: '))
c = int(input('Digite o lado 3: '))

if a == b and b == c:
   print('Equilatero')
elif a == b or a == c or b == c:
   print('Isóceles')
else:
   print('Escaleno')
```

Praticando

- Se você ainda não fez isso no seu programa, lembre que existem números que não formam um triângulo. Por exemplo 2, 2, 5.
- No bhaskara, faça com que seu programa mostre apenas um valor, caso as duas raízes sejam iguais.

Voltando ao problema

- Escreva um programa para computar uma raiz da função $f(x) = x^3 - x^2 - 13x + 8$ usando 10 iterações do método da bisseção.