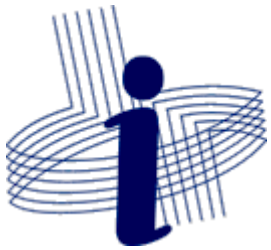
Universidade Federal de Viçosa
Centro de Ciências Exatas e Tecnológicas
Departamento de Informática



INF 100 – Introdução à Programação

Funções
(parte 1)

Motivação

- Exemplo de cálculo de $\sqrt{x}$ usando o algoritmo proposto por Heron de Alexandria (Método de Newton):

```
1. Leia x
2. Faça r = x/2  # chute inicial para a raiz
3. Faça r = (r + x/r) / 2
4. Se |r² - x| > ε, retorne ao passo 3
5. Escreva r
```

onde **ε** = um erro estabelecido qualquer, por exemplo, 10^{-10}.

Motivação

- Possível implementação em Python:

```
x = float( input('Entre com o valor de x: '))
r = x/2  # chute inicial para a raiz
while abs( r*r - x ) > 1e-10:
    r = (r + x/r) / 2
print('Raiz de', x, '=', r )
```

Funções em Python

- Por isso usamos funções:

```
import math

x = float( input('Entre com o valor de x: '))
r = math.sqrt( x )
print('Raiz de', x, '=', r )
```

Funções em Python

- Outros exemplos de funções da biblioteca `math`:
  - **`log(x)`**
  - **`log10(x)`**
  - **`exp(x)`**
  - **`sqrt(x)`**
  - **`tan(x)`**
  - **`sin(x)`**
  - **`cos(x)`**
  - **…**

Identificando oportunidades para reuso de código...

Prática 05

```
# Este programa calcula o dia e mês do
# domingo de páscoa entre um ano inicial
# e um ano final fornecidos pelo usuário

ano1 = int( input('Digite o ano inicial (1582 a 2499): '))
while ano1 < 1582 or ano1 > 2499:
    ano1 = int( input('Digite o ano inicial (1582 a 2499): '))

print()

ano2 = int( input('Digite o ano final (1582 a 2499)  : '))
while ano2 < 1582 or ano2 > 2499:
    ano2 = int( input('Digite o ano final (1582 a 2499)  : '))

print('\nAno     Data da Páscoa')
print('----------------------')
ano = ano1
while ano <= ano2:
    ...
```

Identificando oportunidades para reuso de código...

Prática 05

```
# Este programa calcula o dia e mês do
# domingo de páscoa entre um ano inicial
# e um ano final fornecidos pelo usuário

print()

print('\nAno     Data da Páscoa')
print('----------------------')
ano = ano1
while ano <= ano2:
    ...
```

Identificando oportunidades para reuso de código...

```
while True:
    n = int( input('Entre com a quantidade de alunos: '))
    if n < 2 or n > 50:
        print('Valor deve estar entre 2 e 50')
    else:
        break
soma = 0
for i in range(0, n):
    while True:
        x = int( input('Entre com a nota do próximo aluno: '))
        if x < 0 or x > 100:
            print('Valor deve estar entre 0 e 100')
        else:
            break
    soma = soma + x
media = soma / n
print('Média das notas:', media )
```

...

Identificando oportunidades para reuso de código...

```
soma = 0
for i in range(0, n):
```

```
    soma = soma + x
media = soma / n
```

```
# Este programa calcula o dia e mês do
# domingo de páscoa entre um ano inicial
# e um ano final fornecidos pelo usuário

ano1 = leiaInt('Digite o ano inicial (1582 a 2499): ', 1582, 2499)
print()
ano2 = leiaInt('Digite o ano final (1582 a 2499)  : ', 1582, 2499)

print('\nAno     Data da Páscoa')
print('----------------------')
ano = ano1
while ano <= ano2:
    ...
```

O código para a função leiaInt será apresentado mais adiante...

```
n = leiaInt('Entre com a quantidade de alunos: ', 2, 50)
soma = 0
for i in range(0, n):
    x = leiaInt('Entre com a nota do próximo aluno: ', 0, 100)
    soma = soma + x
media = soma / n
print('Média das notas:', media )
```

O código para a função leiaInt será apresentado mais adiante...

Definição de uma função em Python

```
def nome( lista_de_parâmetros ):
    <comandos>
    return <valor(es) de retorno>
```

- **nome**: nome usado para chamar (usar) a função.
- **lista_de_parâmetros**: dados (constantes, variáveis) passados para a função.

```
# Este programa calcula o dia e mês do
# domingo de páscoa entre um ano inicial
# e um ano final fornecidos pelo usuário

def leiaInt( msg, vmin, vmax ):
    v = int( input( msg ))
    while v < vmin or v > vmax:
        v = int( input( msg ))
    return v

ano1 = leiaInt('Digite o ano inicial (1582 a 2499): ', 1582, 2499)
print()
ano2 = leiaInt('Digite o ano final (1582 a 2499)  : ', 1582, 2499)

print('\nAno     Data da Páscoa')
print('----------------------')
ano = ano1
while ano <= ano2:
    ...
```

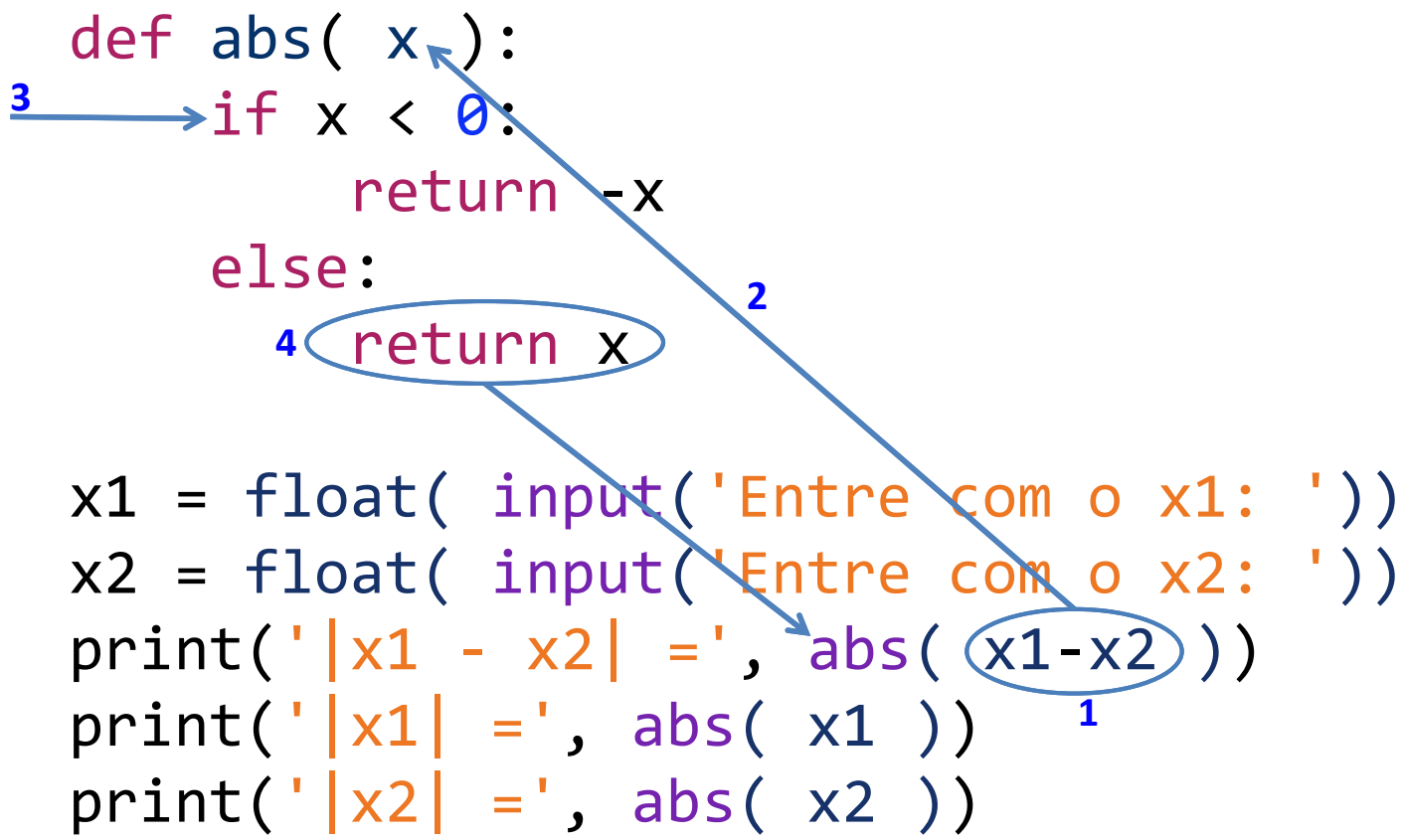
```
def leiaInt( msg, vmin, vmax ):
    v = int( input( msg ))
    while v < vmin or v > vmax:
        print('Valor deve estar entre', vmin, 'e', vmax )
        v = int( input( msg ))
    return v

n = leiaInt('Entre com a quantidade de alunos: ', 2, 50)
soma = 0
for i in range(0, n):
    x = leiaInt('Entre com a nota do próximo aluno: ', 0, 100)
    soma = soma + x
media = soma / n
print('Média das notas:', media )
```

Exercício

- Defina uma função chamada **`abs()`** que aceita como parâmetro um valor x qualquer e retorna o módulo ou valor absoluto de x.

Exercício – programa completo



```
def leiaInt( msg, vmin, vmax ):
    v = int( input( msg ))
    while v < vmin or v > vmax:
        v = int( input( msg ))
    return v
```

Esta função não mostra uma mensagem de erro quando o usuário entra com um valor fora do intervalo.

```
def leiaInt( msg, vmin, vmax ):
    v = int( input( msg ))
    while v < vmin or v > vmax:
        print('Valor deve estar entre', vmin, 'e', vmax )
        v = int( input( msg ))
    return v
```

Esta função mostra!

Como implementar uma função mais genérica, que pode mostrar ou não a mensagem de erro, de acordo com a minha necessidade?

```
def leiaInt( msg, vmin, vmax, mostraErro ):
    v = int( input( msg ))
    while v < vmin or v > vmax:
        if mostraErro:
            print('Valor deve estar entre', vmin, 'e', vmax )
        v = int( input( msg ))
    return v

ano1 = leiaInt('Ano inicial (1582 a 2499): ', 1582, 2499, False)
print()
ano2 = leiaInt('Ano final (1582 a 2499)  : ', 1582, 2499, False)

...

n = leiaInt('Entre com a quantidade de alunos: ', 2, 50, True)
soma = 0
for i in range(0, n):
   x = leiaInt('Entre com a nota do próximo aluno: ', 0, 100, True)
   soma = soma + x
media = soma / n
print('Média das notas:', media )
```

Parâmetros podem assumir valores ”default”:

```
def leiaInt( msg, vmin, vmax, mostraErro=True ):
    v = int( input( msg ))
    while v < vmin or v > vmax:
        if mostraErro:
            print('Valor deve estar entre', vmin, 'e', vmax )
        v = int( input( msg ))
    return v

ano1 = leiaInt('Ano inicial (1582 a 2499): ', 1582, 2499, False)

...

n = leiaInt('Entre com a quantidade de alunos: ', 2, 50 )
```

mostraErro=False

mostraErro=True

Mais exemplos

```
...
def media( a, b ):
    m = (a + b) / 2
    return m

m = media( 5.5, 7.8 )
print('Média =', m )
```

Mais exemplos

```
...
def media( a, b ):
    return (a + b) / 2

m = media( 5.5, 7.8 )
print('Média =', m )
```

Mais exemplos

```
...
def maior( a, b ):
    if a > b:
        return a
    else:
        return b

x = float( input('x = '))
y = float( input('y = '))
print('Maior valor =', maior( x, y ))
```

Funções em Python

- **Erros comuns:**

```
def f( x, y ):
    r = x*x + y
```

(esqueceu do *return*)

```
def funcao( x, y ):
    return x*x + y

def pi():
    return 3.1415926535897

print( pi )
print( funcao( 12 ))
```

pi()

(faltou um parâmetro)

Exercício 1

- Implemente uma função que recebe como parâmetros três valores reais a, b e c, e retorna o maior deles. Use duas abordagens:
  a) Uma função `maior3( a, b, c )` totalmente independente;
  b) Uma função `maior3( a, b, c )` que usa a função `maior( a, b )` já feita antes.

Exercício 1a

```
def maior3( a, b, c ):
    if a > b and a > c:
        return a
    else:
        if b > c:
            return b
        else:
            return c
```

Exercício 1a

```
def maior3( a, b, c ):
    m = a
    if b > m: m = b
    if c > m: m = c
    return m
```

Exercício 1b

```
def maior( a, b ):
    if a > b: return a
    else: return b

def maior3( a, b, c ):
    return maior( maior( a, b ), c )
```

Exercício 2

- Implemente uma função que recebe como parâmetro um valor inteiro n e retorna o valor de $n!$. A função pode supor que n será sempre inteiro e não negativo, sem ter que verificar essa condição.

Exercício 2

```
# versão 1
def fatorial( n ):
    fat = 1
    while n > 1:
        fat = fat * n
        n = n - 1
    return fat
```

Exercício 2

```
# versão 2
def fatorial( n ):
    fat = 1
    for i in range(2, n+1):
        fat = fat * i
    return fat
```

Para Casa

Faça os exercícios disponíveis no PVANet.