UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA

DIM0320 Algoritmos e Programação de Computadores

#FUNÇÕES

ELIEZIO SOARES

ELIEZIOSOARES@DIMAP.UFRN.BR

Modularização / Decomposição

- Os programas de computador que resolvem problemas reais, comumente são muito maiores do que os programas apresentados nestes primeiros passos.
- A melhor maneira de desenvolver, manter e evoluir um programa grande (complexo) é construí-lo como uma composição de pequenas partes, ou módulos.
- Cada módulo sendo responsável por tarefas menores dilui-se a complexidade do programa original.

Funções

- Funções são módulos de um programa de computador.
- Os programas são escritos combinando novas funções desenvolvidas pelos programadores com funções da biblioteca padrão do Python.
- Uma função realiza uma rotina com começo, meio e fim e podem/devem ser executadas quantas vezes forem necessárias.

- Exemplo:
  - Exibir informações na tela.
  - Ler um arquivo do disco.
  - Realizar uma operação matemática.
  - Exibir um menu de opções.
  - Solicitar uma entrada ao usuário.

Funções

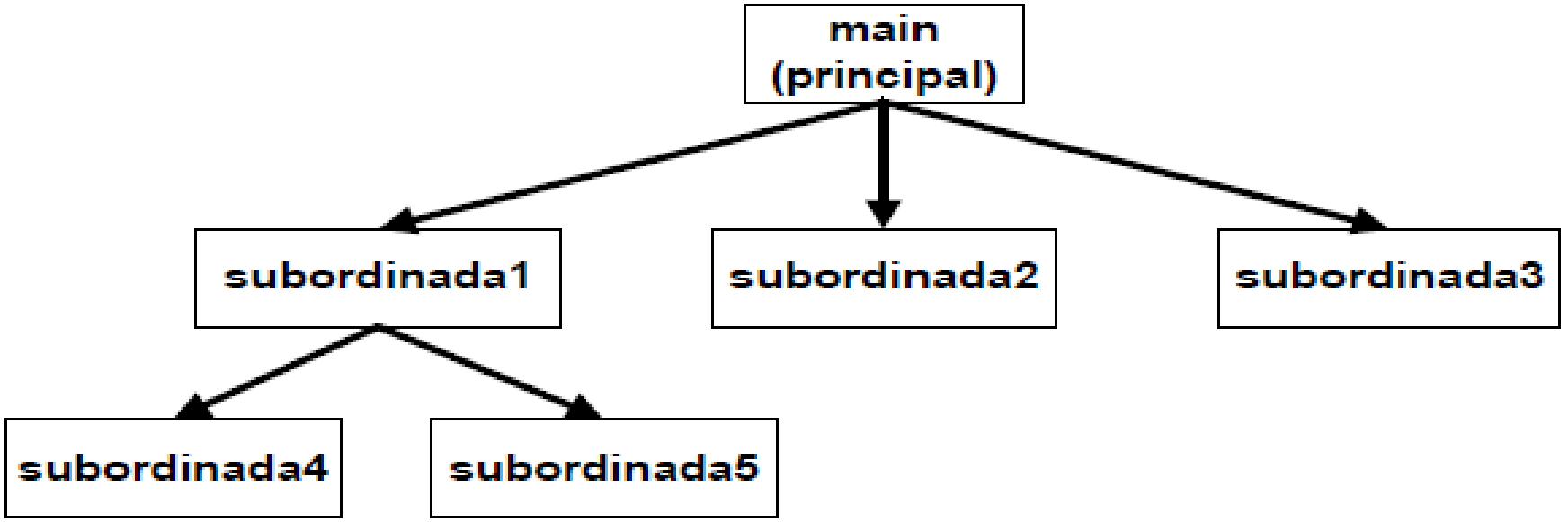
- As funções são ativadas / invocadas / chamadas / executadas por uma **chamada de função.**

- Uma chamada de função especifica o nome da função a ser executada e fornece informações (argumentos / parâmetros) exigidas pela função para realizar a tarefa a que se propõe.

Funções

- Uma função pode chamar outra função através do nome da função seguido pelo parêntese esquerdo, depois os argumentos (ou vários separados por vírgulas) e o parêntese direito.

print(" %f " %5.0);



Trabalhando o conceito...

- Cada um será uma função e pode fazer o que bem entender com os parâmetros para retornar o resultado correto.

- Função Somar()
- Função Subtrair()
- Função dividir()

- Exemplo:
  - Calcular a média de um aluno.

Definição de funções

- Nós já utilizamos **chamadas de função**:
  - int(‘50’)
  - float(’55.0’)
  - print(“Olá!”)
  - input(“Fala: ”)

Definição de funções

- Sintaxe:

```
def    nome-da-função (lista-de-parâmetros) :
        //instruções
```

- Nome-da-função: Qualquer identificador válido.
- Lista-de-parâmetros: É uma lista separada por vírgula contendo declarações de variáveis que receberão valores na invocação da função.

Definição de funções

- Considere implementar uma função **quadrado** para calcular os quadrados dos números inteiros.

- Essa função recebe um parâmetro inteiro e retorna o número recebido multiplicado por ele mesmo.

```
def quadrado(numero):
    return numero*numero
```

Devolução de controle para o invocador

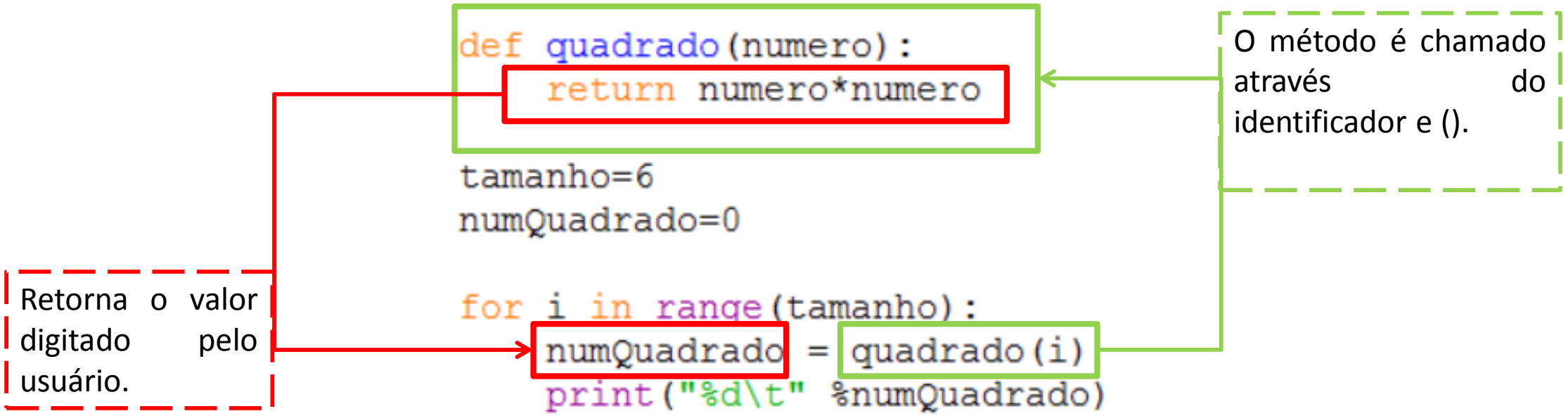
- O controle é devolvido ao se atingir a última instrução identada em relação a palavra **def**.
- Através da interrupção da execução com a palavra **return;**
  - **Exemplo:**

```
def imprimeQuadrado(numero):
    if numero==1:
        return
    print("%d\t" %(numero*numero))
```

- Através do retorno de um valor com a palavra **return** seguida por uma variável, valor, ou expressão.

```
def quadrado(numero):
    return numero*numero
```

Invocando Funções



Exemplo Completo

```
def imprimeQuadrado(numero):
    if numero<=1:
        return
    print("%d\t" %(numero*numero))

def quadrado(numero):
    return numero*numero

tamanho=6
numQuadrado=0

for i in range(tamanho):
    imprimeQuadrado(i)
```

Construindo um código...

- Passo I:
  - Definir a função exibeMenu()
    - A função método exibeMenu mostra as opções para a escolha do usuário.
    - A função deve ler a opção do usuário e retornar um inteiro com o número digitado.

```
def exibeMenu():
    print("##### MENU #####\n")
    print("0- SAIR\n")
    print("1- SOMAR\n")
    opcao = int(input("Escolha uma opcao: "))
    return opcao
```

Construindo um código...

- Passo II:
  - Definir a função somar(numero1, numero2)
    - A função somar recebe dois números inteiros como parâmetro.
    - A função deve somar os dois números e retornar o valor resultante.

```
def somar(numero1, numero2):
    resultado = numero1+numero2
    return resultado
```

Construindo um código...

- Passo III:
  - Implementar o programa principal
    - O programa principal será executado após a definição das funções.
    - Usaremos as variáveis opcao (para armazenar a escolha do usuário), num1 (para armazenar o primeiro número digitado para a operação), num2 (para armazenar o segundo número digitado para a operação) e resultado (para armazenar o valor após a operação escolhida).
    - Construa um laço para exibir o menu e realizar operações até que o usuário escolha a opção 0 que equivale a sair do programa.
    - Execute o método somar se a escolha do usuário for 1 (equivale a opção Somar).

```
def exibeMenu():
    print("##### MENU #####\n")
    print("0- SAIR\n")
    print("1- SOMAR\n")
    opcao = int(input("Escolha uma opcao: "))
    return opcao

def somar(numero1, numero2):
    resultado = numero1+numero2
    return resultado

i=0
opcao=1
num1=0
num2=0
resultado=0

while opcao!=0:
    opcao = exibeMenu()
    if opcao <= 0:
        break

    num1= float(input("Informe o primeiro numero para a operacao: "))
    num2= float(input("Informe o segundo numero para a operacao: "))
    if opcao == 1:
        resultado = somar(num1, num2)
    print("Resultado: %f\n\n" %resultado)
```

Dúvidas

Exercício 1

1. Altere a função exibeMenu() para exibir as opções:
   - Sair
   - Somar
   - Subtrair
   - Multiplicar
   - Dividir

2. Implemente as seguintes funções:
   - Somar(numero1, numero2)
   - Subtrair(numero1, numero2)
   - Multiplicar(numero1, numero2)
   - Dividir(numero1, numero2)

3. Altere o programa principal para executar os métodos acima conforme escolha do usuário.

Exercício 2

- Construa um programa que manipule uma lâmpada. O programa deve exibir as seguintes opções ao usuário: (0)Sair; (1) Acender luz; (2) Apagar luz; (3) Consultar estado atual;

- Para isso o programa deve implementar as funções:
  - exibeOpcoes()
    - Escreve na tela o menu de opções.
  - acenderLampada()
    - Altera a lâmpada para acesa.
  - apagarLampada()
    - Altera a lâmpada para apagada.
  - exibirStatus()
    - Informa o estado atual da lâmpada.

Exercício 3

- Adapte o programa anterior (crie um novo projeto) para funcionar com 20 lâmpadas. Cada lâmpada será identificada por um número sequencial. Para acender, apagar, ou consultar o estado de uma lâmpada será necessário informar o identificador da lâmpada desejada.

- Para isso o programa deve implementar as funções:
  - exibeOpcoes()
    - Escreve na tela o menu de opções.
  - acenderLampada(int idLampada)
    - Altera a lâmpada para acesa.
  - apagarLampada(int idLampada)
    - Altera a lâmpada para apagada.
  - exibirStatus(int idLampada)
    - Informa o estado atual da lâmpada.
  - exibirTodas()
    - Informa o estado atual de todas as lâmpadas.

Escopo das Variáveis

"É um contexto limitado aos quais valores e expressões estão associados."

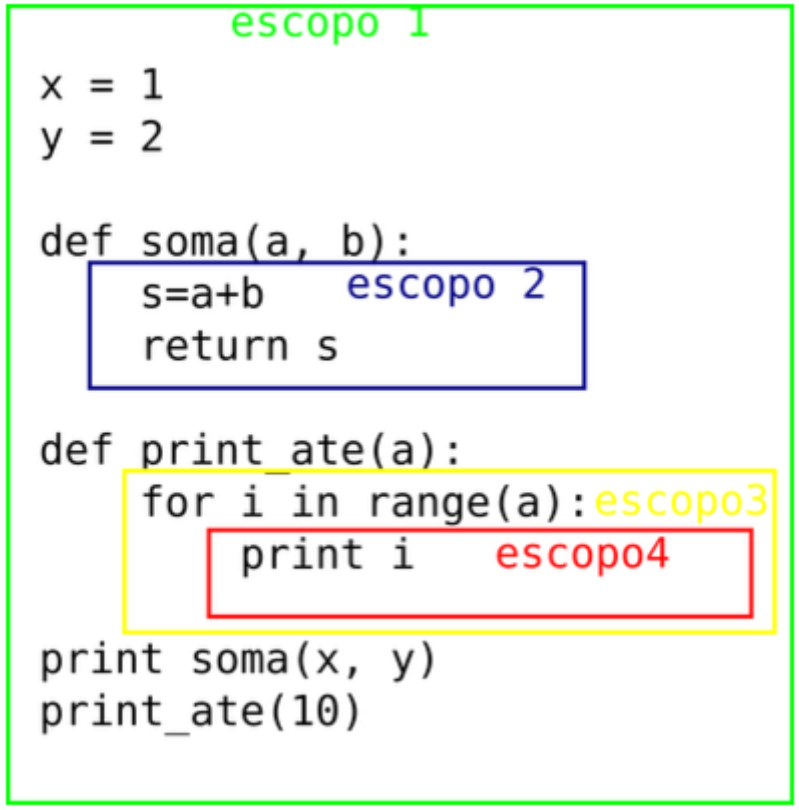
- Python utiliza escopo estático
  - "No escopo estático, o nome é ligado a uma coleção de comandos de acordo com sua posição no programa-fonte."

Escopo Local x Escopo Global

- Escopo Local:
  - O primeiro tipo de variáveis que veremos são as variáveis locais. Estas são aquelas que só tem validade dentro do bloco no qual são declaradas.

- Escopo Global:
  - Variáveis globais são declaradas, como já sabemos, fora de todas as funções do programa. Elas são conhecidas e podem ser alteradas por todas as funções do programa.

Escopo Local x Escopo Global



Acessando variáveis globais

```
def func():
    x=1
    print(x)

x = 42
func()
print(x)
```

```
def func():
    global x
    x=1
    print(x)

x = 42
func()
print(x)
```

Dúvidas

Exercício 2

- Construa um programa que manipule uma lâmpada. O programa deve exibir as seguintes opções ao usuário: (0)Sair; (1) Acender luz; (2) Apagar luz; (3) Consultar estado atual;

- Para isso o programa deve implementar as funções:
  - exibeOpcoes()
    - Escreve na tela o menu de opções.
  - acenderLampada()
    - Altera a lâmpada para acesa.
  - apagarLampada()
    - Altera a lâmpada para apagada.
  - exibirStatus()
    - Informa o estado atual da lâmpada.

Exercício 3

- Adapte o programa anterior (crie um novo projeto) para funcionar com 20 lâmpadas. Cada lâmpada será identificada por um número sequencial. Para acender, apagar, ou consultar o estado de uma lâmpada será necessário informar o identificador da lâmpada desejada.

- Para isso o programa deve implementar as funções:
  - exibeOpcoes()
    - Escreve na tela o menu de opções.
  - acenderLampada(int idLampada)
    - Altera a lâmpada para acesa.
  - apagarLampada(int idLampada)
    - Altera a lâmpada para apagada.
  - exibirStatus(int idLampada)
    - Informa o estado atual da lâmpada.
  - exibirTodas()
    - Informa o estado atual de todas as lâmpadas.