UNAMA

Prof. Fabrício Araújo

Lógica de Programação Algorítmica

Aula 9 - 18/05/2023

https://bit.ly/23_1_logica_n

O que temos para hoje?

- O que vimos até agora;
- Situação problema;
- Funções;
- Recursividade;
- Lista de exercícios;

O que vimos até agora

O que vimos até agora

- Entrada de dados;
- Declaração de variáveis;
- Operações matemáticas;
- Estruturas de sequência;
- Estruturas de seleção;
- Estruturas de repetição;
- Funções;
- Saída de dados;

Situação problema

Situação problema

- É muito comum, durante a construção de um programa, termos códigos que se repetem;
- Imagine que você tem um bloco de código que se repete em 5 partes do seu programa;
- Depois que você escreveu o programa, você verificou que esse código repetido possui um erro;

Situação problema

- Você precisa, então, modificar os 5 locais onde esse código foi repetido para fazer a correção;
- E se, na verdade, não eram 5 locais, mas sim 6 e você esqueceu o último local onde estava o código errado?

Funções

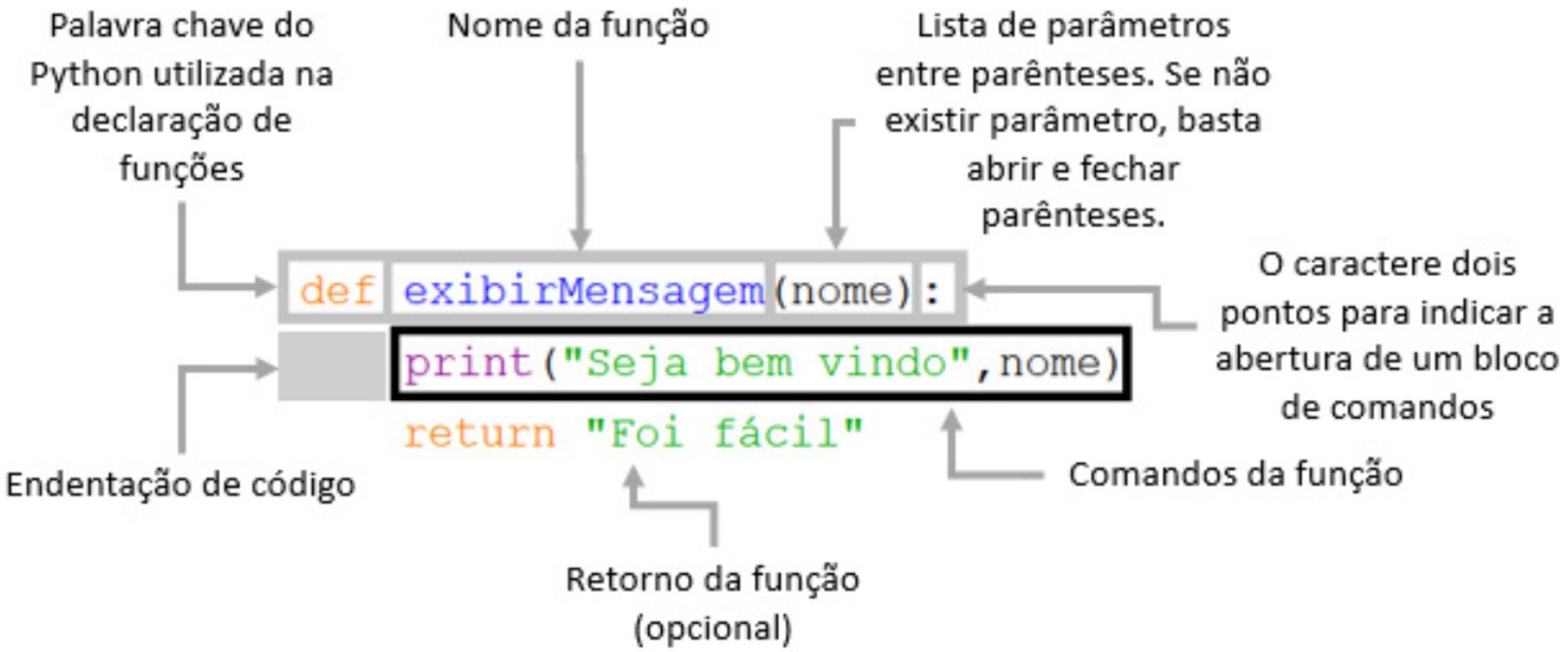
Funções

- Função é um nome para um conjunto de comandos que pode ser chamado várias vezes em pontos diferentes do programa, sem a necessidade de repetição de código;

Funções

- As funções também servem para deixar o código mais fácil de ser lido;

Funções


Palavra chave do Python utilizada na declaração de funções
Nome da função
Lista de parâmetros entre parênteses. Se não existir parâmetro, basta abrir e fechar parênteses.
def exibirMensagem(nome):
print("Seja bem vindo",nome)
return "Foi fácil"
O caractere dois pontos para indicar a abertura de um bloco de comandos
Endentação de código
Comandos da função
Retorno da função (opcional)

Funções

slide12.py > ...

```python
def exibirMensagem(nome):
    print("Seja vem vindo "+nome)
    return "Foi fácil"
```

Funções

- O que acontece?

Funções

- As funções precisam ser chamadas;

Funções

slide16.py > ...

```python
def exibirMensagem(nome):
    print("Seja vem vindo "+nome)
    return "Foi fácil"

mensagem = exibirMensagem("Fabricio")
print(mensagem)
```

Funções

- Escreva uma função que receba como parâmetros dois números e escreva o resultado da soma entre eles;

Funções

slide17.py > ...

```
1    def soma(x,y):
2        z = x + y
3        print(z)
4
5    soma(2,3)
```

Funções

- Escreva uma função que receba como parâmetros dois números e retorne o resultado da soma entre eles;

Funções

slide19.py > ...

```
1   def soma(x,y):
2       z = x + y
3       return z
4
5   resultado = soma(8,9)
6   print(resultado)
```

Funções

- Funções importantes que já vimos: input, print, range...

Funções

- Exercício: crie uma função que receba como parâmetro um número e retorne se esse número é par;

Funções

slide22.py > ...

```python
def is_par(n):
    if n % 2 == 0:
        return True
    else:
        return False

x = input("Digite um número: ")
x = int(x)
print(str(x)+" é par? "+str(is_par(x)))
```

Funções

- Exercício: crie uma função que receba como parâmetro um número e retorne a soma de todos os números de 0 até esse número (incluindo esse número);

Funções

slide24.py > ...

```python
def somatoria(n):
    soma = 0
    for i in range(n+1):
        soma = soma + i
    return soma

n = input("Digite um número: ")
n = int(n)
soma = somatoria(n)
print("A somatória de todos os número de 0 até "+str(n)+" é: "+str(soma))
```

Funções

- Exercício: crie uma função que receba como parâmetro um número e retorne se este número é primo;

Funções

slide26.py > ...

```python

def is_primo(n):
    contador = 2
    while contador < n:
        if n % contador == 0:
            return False
        contador = contador + 1

    return True

n = input("Digite um número: ")
n = int(n)
print(str(n)+" é primo? "+str(is_primo(n)))
```

Recursividade

Recursividade

- Funções recursivas são funções que chamam a si mesmas de maneira direta ou indireta;
- Não há nenhum benefício em termos de desempenho ao usar funções recursivas em Python, já que laços podem resolver o problema com mais eficiência.

Recursividade

- Uma função recursiva tem duas partes:
  - Condição de parada: onde a recursão para e os valores são calculados;
  - Condição de recursão: onde a recursão acontece;

Recursividade

- Exemplo: desenvolva uma função recursiva que calcule o fatorial de um número;

Recursividade

slide31.py > ...

```python
def fatorial(n):
    if n == 1 or n == 0:
        return 1
    return n * fatorial(n - 1)
retorno = fatorial(5)
print(retorno)
```

Lista de exercícios

Lista de exercícios

- Acessar o arquivo lista4.pdf no teams da disciplina;
- Resolver as questões;

Obrigado!

Alguma pergunta?

Contato:

- 040601692@prof.unama.br