

Introdução à Programação de Computadores

Prof. Max Vieira Santiago
max.santiago@dcc.ufmg.br

DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

U F *m* G

Professor

- Graduação
 - Sistemas de Informação.
- Pós-graduação
 - Análise de Sistemas.
- Pós-graduação
 - Análise de Negócios.
- Pós-graduação
 - Estatística.
- Mestrado
 - Ciências da Computação.
- Pós-graduação
 - Ciência de Dados e Analytics.
- Pós-graduação
 - Inteligência Artificial e Machine Learning.
- Doutorado
 - Inteligência Artificial e Machine Learning.

- Squad BI Industrial
 - ArcelorMittal (Ciência de Dados) – 23 anos.

Introdução à Programação de Computadores

Ementa:

- Metodologia de desenvolvimento de programas. Programação em Linguagem de Alto-Nível. Comandos Básicos. Modularização. Estruturas de dados. Introdução à bibliotecas científicas.

Objetivo:

- Ajudar a desenvolver o raciocínio lógico e a capacidade de abstração usando um sistema de computação e linguagem de programação de alto nível (Python) como ferramenta de apoio

Programa:

- | | |
|--|--|
| ✓ Sistemas de Computação | ✓ Procedimentos e Funções |
| ✓ Fases da Resolução de um Problema via Computador | ✓ Manipulação de Arquivos |
| ✓ Estruturas Básicas | ✓ Estruturas de dados |
| ✓ Comandos Condicionais | ✓ Introdução a bibliotecas científicas |
| ✓ Comandos de Repetição | |

Estrutura das Aulas

AULAS EXPOSITIVAS

Objetivos:

- Introduzir conceitos
- Apresentar exemplos básicos

Alunos acompanharão com:

- Slides da aula

Fixação do conteúdo:

- **Exercícios interativos** no final de cada aula

AULAS INTERATIVAS

Objetivos:

- Fixar conceitos
- Obter experiência em programação
- Resolver problemas práticos

Alunos farão exercícios de programação relacionados ao conteúdo teórico da aula anterior associados a problemas reais

Professor e monitores estarão disponíveis para auxiliar nas dúvidas de resolução das atividades

AVALIAÇÃO

Objetivos:

- Avaliar as habilidades de programação do aluno
- Verificar o grau de evolução do aprendizado do aluno e da turma

Alunos farão exercícios de programação relacionados ao conteúdo teórico das aulas anteriores

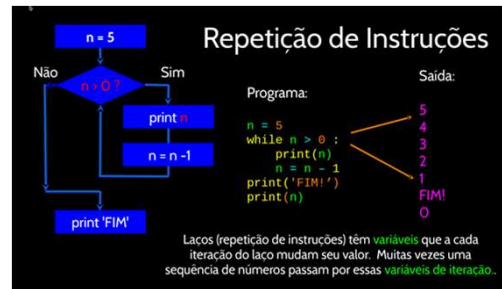
Avaliação será realizada de forma automática

Aulas Expositivas

Exemplos

Laço de Repetição Condicional

- Um laço de repetição pode ser chamado de "Condicional" porque continuam rodando até que uma condição lógica torna-se *False*
- Utilizamos o comando `while` para criar laços de repetição condicional



Um Laço Infinito

```
n = 5
while n > 0 :
    print('Enxaguar')
    print('Enxaguar')
print('Secar!')
```

O que está errado com esse laço?

Funções

- Há dois tipos de *funções* em Python.
- *Funções pré-definidas* que são as fornecidas como parte do Python - `input()`, `print()`, `type()`, `float()`, `int()`, `str()` ...
- *Funções* que nós *mesmos* definimos e então as usamos
- Nós tratamos os nomes das *funções* pré-definidas como "novas" *palavras reservadas* (i.e., nós as evitamos como nomes de variáveis)

Definição de Função

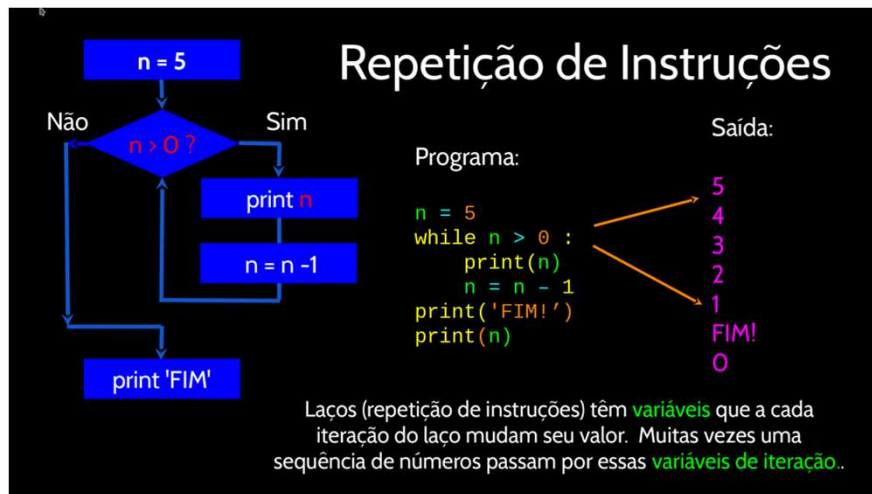
- No Python uma *função* é algum código reutilizável que leva(m) *argumento(s)* como entrada(s), faz alguma computação e então retorna um resultado ou resultados
- Definimos uma *função* usando a palavra reservada `def`
- Chamamos/invocamos a *função* usando o nome da função, parênteses, e *argumentos* em uma expressão

Construindo sua própria função

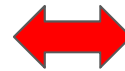
- Nós criamos uma nova *função* usando a palavra reservada `def` o nome da função e *parâmetros* opcionais entre parênteses
- Nós indentamos o corpo da função
- Isso *define* a função *mas não* executa o corpo da função

```
def print_lyrics():
    print("I'm a lumberjack, and I'm okay.")
    print("I sleep all night and I work all day.")
```

Aulas Expositivas (conceitos)



Professor



Aula6.ipynb Python 3

Slide 5 - Repetição de Instruções

Laços (repetição de instruções) têm variáveis que a cada iteração do laço mudam seu valor. Muitas vezes uma sequência de números passam por essas variáveis de iteração.

Exemplo:

```
In [3]: 1 n = 5
2 while n > 0 :
3     print(n)
4     n = n - 1
5     print('Blastoff!')
6     print(n)

5
4
3
2
1
Blastoff!
0
```

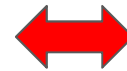
Aluno

Aulas Expositivas (exercícios)

Exercício

Escreva uma função chamada `num_pares` que receba como parâmetro um número inteiro `n` e retorne a quantidade de números pares entre `[2, n]`

Professor



```
4 print('Enxaguar')
5 print('Secar!')
```

Exercício

Escreva uma função chamada `num_pares` que receba como parâmetro um número inteiro `n` e retorne a quantidade de números pares entre `[2, n]`

```
In [ ]: 1 # Escreva sua solução aqui
        2
        3
        4
        5
        6
        7
        8
        9
        10
```

```
In [ ]: 1
        2
        3
        4
        5
        6
```

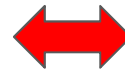
Aluno

Aulas Expositivas (exercícios)

Solução

```
def num_pares(n):  
    quantidade = 0  
    for i in range(2, n + 1):  
        if i % 2 == 0:  
            quantidade = quantidade + 1  
    return quantidade
```

Professor



```
5 print('Secar!')
```

Exercício

Escreva uma função chamada `num_pares` que receba como parâmetro um número inteiro `n` e retorne a quantidade de números pares entre `[2, n]`

```
In [2]: 1 # Escreva sua solução aqui  
2  
3 def num_pares(n):  
4     quantidade = 0  
5     for i in range(2, n + 1):  
6         if i % 2 == 0:  
7             quantidade = quantidade + 1  
8     return quantidade  
9  
10  
11  
12  
13  
14  
15
```

Aluno

Aulas Interativas - Fixação de Conceitos

Descrição

Lista de envios

Similaridades

Atividade de teste

Enviar

Editar

Visualizar envios

Nota

Lista de envios anteriores

problema1.py

1

2

3

4

5

6

7

8

9

-*- coding: utf-8 -*-

raio = float(input('Digite o valor do raio da circunferência: '))

pi = 3.1415

perimetro = 2 * pi * raio

area = pi * (raio ** 2)

volume = 4 * pi * ((raio ** 3) / 3)

Nota proposta: 100 / 100

Comentários

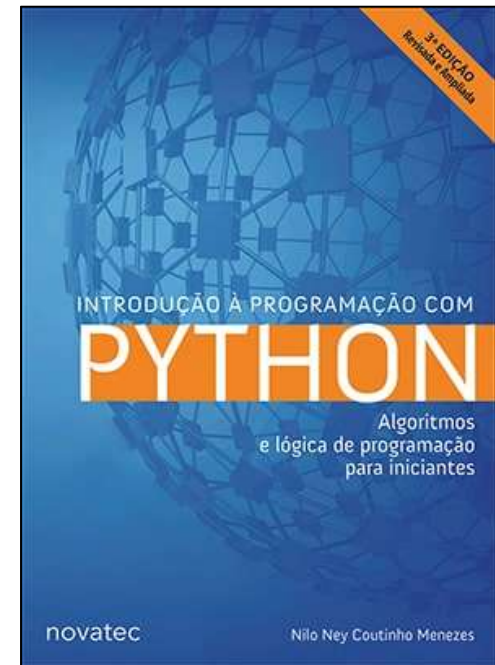
Summary of tests

4 tests run/ 4 tests

Execução

Python

- Todo o curso será dado na linguagem Python
- Livro texto:
 - Python for Informatics: Exploring Information (disponível gratuitamente em <http://www.pythonlearn.com/book.php>)
- Outros materiais:
 - MENEZES, Nilo Ney Coutinho. **Introdução à programação com Python–2ª edição: Algoritmos e lógica de programação para iniciantes**. Novatec Editora, 2016.
 - A Internet é cheia de material!
 - The Python Tutorial: <https://docs.python.org/2/tutorial/>
 - <http://wiki.python.org.br/DocumentacaoPython>

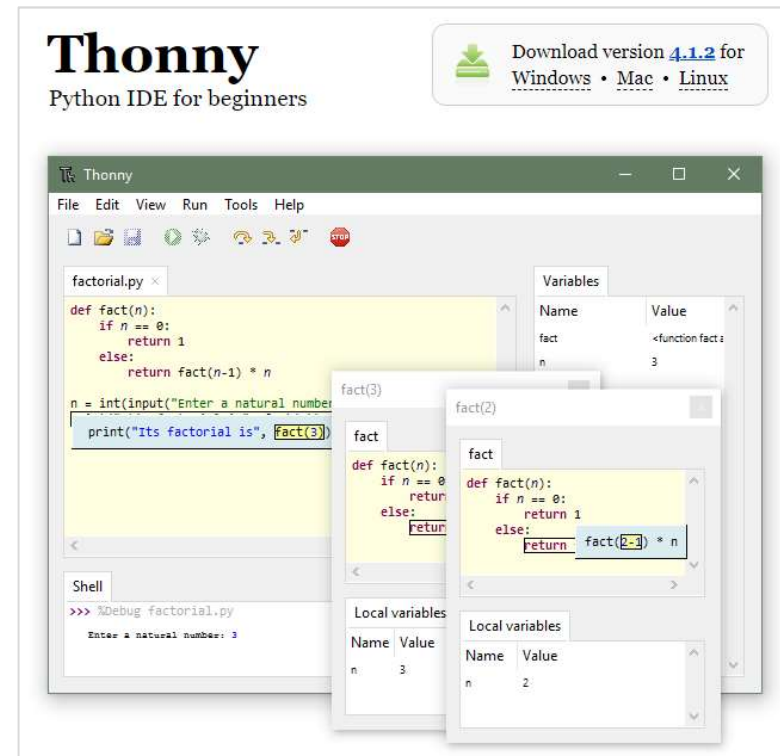


Python

- A escolha de uma linguagem para aprender a programar é um problema complicado
 - Com o tempo, alunos podem usar o conhecimento para aprender outras linguagens
 - Depois de bastante prática, chavear linguagens é algo simples
- Amplamente utilizada em cursos introdutórios
 - Alto nível
 - Sintaxe agradável
 - Organização do código (whitespaces)
 - Biblioteca poderosa

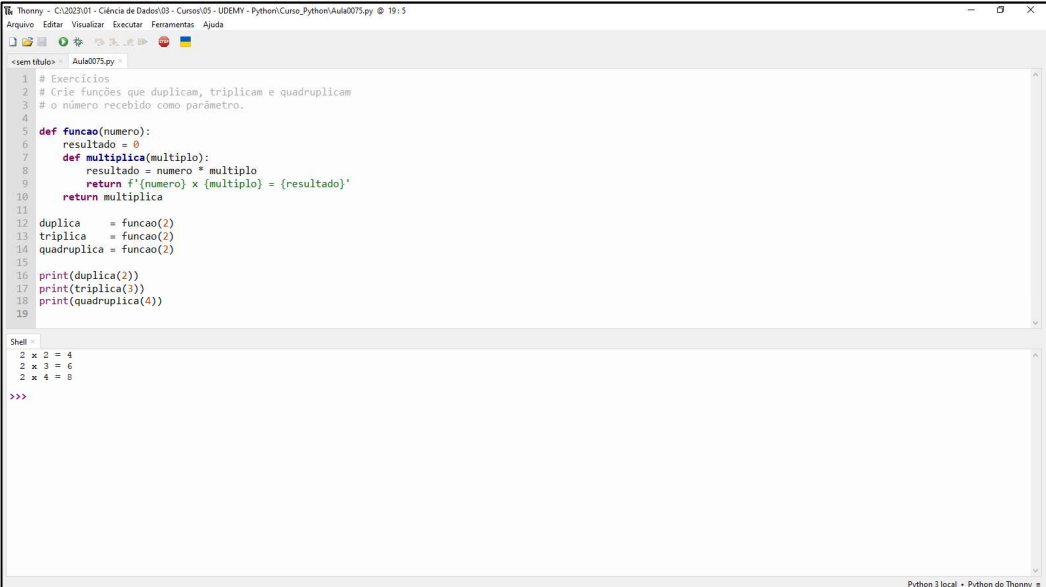
Thonny – IDE (Integrated Development Environment)

- Permite criar e compartilhar documentos que contêm
 - Código
 - Equações
 - Visualizações
 - texto narrativo.
 - <https://thonny.org/>
- Os usos mais comuns são
 - limpeza e transformação de dados
 - simulação numérica
 - modelagem estatística
 - visualização de dados
 - aprendizado de máquina



Thonny na sala de aula

- Lista de exercícios
- Notas de aula
- Demos



The screenshot shows the Thonny Python IDE interface. The top menu bar includes 'Arquivo', 'Editar', 'Visualizar', 'Executar', 'Ferramentas', and 'Ajuda'. The main editor window displays a Python script named 'Aula0075.py' with the following code:

```
1 # Exercícios
2 # Crie funções que duplicam, triplicam e quadruplicam
3 # o número recebido como parâmetro.
4
5 def funcao(numero):
6     resultado = 0
7     def multiplica(multiplo):
8         resultado = numero * multiplo
9         return f'{numero} x {multiplo} = {resultado}'
10    return multiplica
11
12 duplica = funcao(2)
13 triplica = funcao(3)
14 quadruplica = funcao(4)
15
16 print(duplica(2))
17 print(triplica(3))
18 print(quadruplica(4))
19
```

Below the editor, the 'Shell' window shows the output of the script:

```
>>>
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
>>>
```

The status bar at the bottom right indicates 'Python 3 local - Python do Thonny'.

Perguntem !



Avaliação

Avaliação

- 8 Avaliações (10 pontos cada) – **Nota_AVs**
 - Apenas as 7 maiores notas serão consideradas (**SEM substitutiva**)
 - Total: **70 pontos**
- 1 Projeto Final – **Nota_Projeto_Final**
 - Total: **10 pontos**
- 10 Atividades Práticas (2 pontos cada) – **Nota_APs**
 - Total: **20 pontos**
- **Nota Final: Nota_AVs + Nota_Projeto_Final + Nota_APs**

Avaliações (AVs)

- Avaliação – AV1 - 30/08/2023 (quarta-feira) - 10 pontos
- Avaliação – AV2 - 11/09/2023 (segunda-feira) - 10 pontos
- Avaliação – AV3 - 20/09/2023 (quarta-feira) - 10 pontos
- Avaliação – AV4 - 02/10/2023 (segunda-feira) - 10 pontos
- Avaliação – AV5 - 11/10/2023 (quarta-feira) - 10 pontos
- Avaliação – AV6 - 23/10/2023 (segunda-feira) - 10 pontos
- Avaliação – AV7 - 01/11/2023 (quarta-feira) - 10 pontos
- Avaliação – AV8 - 20/11/2023 (segunda-feira) - 10 pontos

Avaliações Práticas (APs)

- | | | |
|----------------------------|------------------------|------------|
| ● Avaliação Prática – AP1 | - 28/08/2023 (segunda) | - 2 pontos |
| ● Avaliação Prática – AP2 | - 06/09/2023 (quarta) | - 2 pontos |
| ● Avaliação Prática – AP3 | - 18/09/2023 (segunda) | - 2 pontos |
| ● Avaliação Prática – AP4 | - 27/09/2023 (quarta) | - 2 pontos |
| ● Avaliação Prática – AP5 | - 09/10/2023 (segunda) | - 2 pontos |
| ● Avaliação Prática – AP6 | - 18/10/2023 (quarta) | - 2 pontos |
| ● Avaliação Prática – AP7 | - 30/10/2023 (segunda) | - 2 pontos |
| ● Avaliação Prática – AP8 | - 06/11/2023 (segunda) | - 2 pontos |
| ● Avaliação Prática – AP9 | - 13/11/2023 (segunda) | - 2 pontos |
| ● Avaliação Prática – AP10 | - 29/11/2023 (quarta) | - 2 pontos |

Por Enquanto

- <https://github.com/MaxVieiraSantiago/UFMG/tree/master/IPC>