



Algoritmos e Programação com Linguagem Python

Prof. Adriano Silva

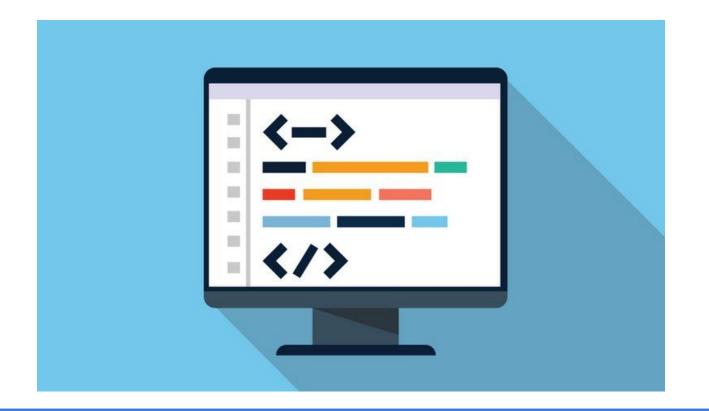
adrianovss@gmail.com



Ementa

- Correção do Exercício 03
- Correção da Atividade 02 + Revisão
- Orientação a Objetos
- Manipulação de Arquivos
- Erros e Exceções





Correção do Exercício 03 & Atividade 02 | Revisão





E continuando...



Conceitos:

- Classe
- Instância
- Objeto
- Atributos e Métodos
- Herança
- Abstração
- Encapsulamento
- o Polimorfismo
- Associação (Composição e Agregação)
- Generalização e Especialização
- Demonstrações no Google Colab



- O statement class indica uma uma definição de uma nova classe.
- A função __init__(self, args...) define o construtor da classe, sendo o primeiro argumento obrigatoriamente a própria instância da classe (self).
- Classes podem ter variáveis de classe que são compartilhadas entre todas as instâncias dessa mesma classe e variáveis de instância que são exclusivas a cada instância.
- Instâncias podem receber atributos dinamicamente.



- O encapsulamento consiste em "esconder" os atributos de uma classe de forma que não possam ser acessados diretamente.
- Em Python não é comum a criação de métodos *get/set* para atributos caso eles não tenham a real necessidade de serem privados.
- A necessidade de privar o acesso a um atributo se dá quando o controle do seu valor é necessário.
- Para tornar um atributo privado, basta iniciar seu nome com um duplo underscore __attrib. Os statements @property e @attr.setter é a forma "Pythonica" de definir métodos getter e setter.



- Em Python não existe o conceito de interfaces para definição de comportamentos, pois tem suporte a herança múltipla.
- Para a definição de classes abstratas, Python provê uma infraestrutura denominada ABC (Abstract Base Classes).
- Para tornar uma classe abstrata, basta estender a classe ABC.
- O statement @abstractmethod instrui o interpretador de que esse método não está implementado e caso uma nova instância dessa classe seja criada, um erro de execução será retornado.
- O método super() é utilizado para acesso ao objeto pai.



- Sobrescrita de métodos (overriding) é permitida, bastando para isso escrever na subclasse o mesmo nome do método com a mesma quantidade de parâmetros.
- Sobrecarga de métodos (overloading) não é possível em Python, em virtude da tipagem dinâmica de variáveis. Porém, é possível simular esse comportamento através de Default Parameters.



Exercícios





- O objeto file é responsável por operações de leitura e escrita (input/output) e também são conhecidos como file-like ou streams.
- Dependendo da forma como é criado, o objeto file pode mediar escrita em tempo real em disco, outros tipos de armazenamentos ou dispositivos de comunicação como buffers em memória, sockets, pipes, etc.
- Files são divididos em três categorias definidas no módulo io:
 - raw binary files
 - buffered binary files
 - text files



- Python provê uma função embutida para denominada open() para criar ou acessar objetos do tipo file:
 - open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)
 - Retorna o objeto file correspondente ou lança uma exceção do tipo OSError caso algum erro ocorra no acesso ao recurso solicitado.



- Pode se abrir um arquivo somente para leitura ou para leitura e escrita:
 - o f = open('arquivo', 'r') (default)
 - o f = open('arquivo', 'w')
- Pode-se especificar o encoding do arquivo no terceiro parâmetro (como por exemplo UTF-8).
- Ao se abrir um arquivo que não existe no modo leitura, um erro é lançado (*FileNotFoundError*) caso seja especificado escrita, o arquivo será criado.
- Um arquivo aberto, sempre deve ser fechado através do método close().



- Métodos para leitura de dados de um arquivo:
 - read()
 - readline()
 - readlines()
- Métodos para escrita de dados em um arquivos:
 - write()
 - writelines()
- Para saber mais: https://docs.python.org/3/library/io.html#module-io



Exercícios





Erros e Exceções

- Há dois tipos de erros em Python:
 - Syntax Errors
 - Exceptions
- Erros de Sintaxe ou erros de conversões são mais comuns (quando estamos aprendendo a linguagem). O interpretador irá informar onde o erro foi encontrado;
- Exceções são eventos que ocorrem quando a sintaxe está correta, porém no momento da execução alguma situação anormal ocorre:
 - Divisão por zero;
 - Conversão inválida de tipos;
 - Arquivo não encontrado.



Erros e Exceções (continuação...)

- Statements **TRY** e **EXCEPT**;
- É permitido a criação de exceções definidas pelo usuário, herdando-se a classe Exception; *
- O statement FINALLY permite que operações denominadas clean-up sejam executadas independente da ocorrência de uma exceção;
- O statement RAISE é utilizado para forçar o lançamento de uma exceção, muito utilizado (principalmente) para o tratamento de exceções de negócio;
- Todas as exceções embutidas no Python, estendem da Base Exception:
 - https://docs.python.org/3/library/exceptions.html#bltin-exceptions



Exercícios





AQUI TEM ENGENHARIA