UniEnsino
CENTRO UNIVERSITÁRIO DO PARANÁ

MATERIAL DE APOIO

DISCIPLINA:

LINGUAGEM DE PROGRAMAÇÃO PYTHON - INICIAL

Material de Apoio para a Disciplina de Linguagem de Programação Python - Inicial

Este material de apoio foi desenvolvido para auxiliar no estudo da disciplina de Linguagem de Programação Python - Inicial, oferecendo resumos concisos de cada aula, recomendações de leitura e explicações adicionais para aprofundar a compreensão dos temas abordados.

Aula 01: Aprofundando no mundo do Python

Resumo: A aula visa consolidar o conhecimento introdutório sobre a linguagem Python, explorando suas características como linguagem de programação poderosa, versátil e de fácil aprendizado. Apresenta os diferentes ambientes de desenvolvimento (IDEs) recomendados para programação em Python, como o Visual Studio Code (VSCode).

Recomendações de Leitura:

- **MATTHES, Eric. Python Crash Course, 2nd Edition. No Starch Press, 2019.** (Capítulos introdutórios)
- Documentação oficial do Python: https://docs.python.org/3/

Explicações Adicionais:

- Python é uma linguagem interpretada, o que significa que o código é executado linha por linha.
- Python possui uma sintaxe simples e legível, o que facilita o aprendizado e a escrita de código.
- Python é amplamente utilizada em diversas áreas, como desenvolvimento web, análise de dados, inteligência artificial e automação.

Aula 02: Dados e Variáveis

Resumo: A aula aborda os conceitos fundamentais de dados e variáveis em Python. São apresentados os diferentes tipos de dados, como inteiros, floats, strings e booleanos, e como declarar e utilizar variáveis para armazenar valores.

Recomendações de Leitura:

- **MENEZES, Nilo Ney Coutinho. Introdução à Programação com Python: Algoritmos e Lógica de Programação. Novatec Editora, 2019.** (Capítulos sobre tipos de dados e variáveis)
- Tutoriais online sobre tipos de dados e variáveis em Python.

Explicações Adicionais:

- Variáveis são nomes que referenciam um espaço na memória onde os dados são armazenados.
- O tipo de dado define o tipo de valor que a variável pode armazenar.
- É importante escolher o tipo de dado correto para garantir a precisão e a eficiência do código.

Aula 03: Cálculos

Resumo: A aula explora os operadores aritméticos em Python, como adição, subtração, multiplicação, divisão e exponenciação. São apresentados exemplos de como utilizar esses operadores para realizar cálculos e manipular valores numéricos.

Recomendações de Leitura:

- **SWEIGART, Al. Automate the Boring Stuff with Python, 2nd Edition. No Starch Press, 2019.** (Capítulos sobre operadores e expressões)
- Materiais online sobre operadores aritméticos em Python.

Explicações Adicionais:

- Python segue a ordem padrão das operações matemáticas (PEMDAS/BODMAS).
- Utilize parênteses para controlar a ordem das operações.
- Operadores de atribuição combinados (+=, -=, *=, /=) simplificam a atualização de variáveis.

Aula 04: Estruturas condicionais

Resumo: A aula introduz as estruturas condicionais em Python, como o if, elif e else. São apresentados exemplos de como utilizar essas estruturas para controlar o fluxo de execução do código com base em condições lógicas.

Recomendações de Leitura:

- **BORGES, Luiz Eduardo. Python para desenvolvedores. Novatec Editora, 2014.** (Capítulos sobre estruturas condicionais)
- Tutoriais online sobre estruturas condicionais em Python.

Explicações Adicionais:

- A estrutura `if` executa um bloco de código se a condição for verdadeira.
- A estrutura `elif` (else if) permite testar múltiplas condições em sequência.
- A estrutura `else` executa um bloco de código se nenhuma das condições anteriores for verdadeira.

Aula 05: Loop em Python

Resumo: A aula explora as estruturas de repetição (loops) em Python, como o for e o while. São apresentados exemplos de como utilizar essas estruturas para repetir um bloco de código várias vezes, iterando sobre listas, strings e outros objetos iteráveis.

Recomendações de Leitura:

- **RAMALHO, Luciano. Fluent Python: Clear, Concise, and Effective Programming. O'Reilly Media, 2015.** (Capítulos sobre iteradores e geradores)
- Materiais online sobre loops em Python.

Explicações Adicionais:

- O loop `for` itera sobre os elementos de uma sequência.
- O loop `while` repete um bloco de código enquanto uma condição for verdadeira.
- Utilize os comandos `break` e `continue` para controlar o fluxo dos loops.

Aula 06: Funções

Resumo: A aula aborda as funções em Python, estruturas que permitem organizar e reutilizar código. São apresentados os conceitos de definição de funções, parâmetros, argumentos, retorno de valores e escopo de variáveis.

Recomendações de Leitura:

- **BEAZLEY, David; JONES, Brian K. Python Cookbook. O'Reilly Media, Incorporated, 2013.** (Capítulos sobre funções)
- Documentação oficial do Python sobre funções: https://docs.python.org/3/tutorial/controlflow.html#defining-functions

Explicações Adicionais:

- Funções permitem modularizar o código e facilitar a manutenção.
- Parâmetros são variáveis que recebem valores quando a função é chamada.
- O comando `return` permite retornar um valor da função.

Aula 07: Módulos

Resumo: A aula explora os módulos em Python, arquivos que contêm código reutilizável. São apresentados os conceitos de importação de módulos, utilização de funções e classes definidas em módulos e criação de seus próprios módulos.

Recomendações de Leitura:

- **SLATKIN, Brett. Effective Python: 90 Specific Ways to Write Better Python. Addison-Wesley Professional, 2019.** (Seções sobre módulos e pacotes)
- Tutoriais sobre como criar e utilizar módulos em Python.

Explicações Adicionais:

- Módulos permitem organizar o código em arquivos separados.
- Utilize o comando `import` para importar módulos.
- Crie seus próprios módulos para organizar e reutilizar seu código.

Aula 08: Strings

Resumo: A aula trata das strings em Python, sequências de caracteres utilizadas para representar texto. São apresentados os diferentes métodos de strings, como fatiamento, concatenação, formatação e busca de padrões.

Recomendações de Leitura:

- Documentação oficial do Python sobre strings: https://docs.python.org/3/library/stdtypes.html#textseq
- Materiais online sobre manipulação de strings em Python.

Explicações Adicionais:

- Strings são imutáveis, o que significa que não podem ser modificadas após a criação.
- Fatiamento permite extrair partes de uma string.
- Formatação permite inserir variáveis em strings.

Aula 09: Arquivos em Python

Resumo: A aula aborda a manipulação de arquivos em Python, incluindo abertura, leitura, escrita e fechamento de arquivos. São apresentados os diferentes modos de abertura de arquivos e as melhores práticas para garantir a segurança dos dados.

Recomendações de Leitura:

- Documentação oficial do Python sobre arquivos: https://docs.python.org/3/tutorial/inputoutput.html#reading-and-writing-files
- Exemplos de código sobre manipulação de arquivos em Python.

Explicações Adicionais:

- Utilize o comando `with` para garantir que o arquivo seja fechado automaticamente.
- Existem diferentes modos de abertura de arquivos (leitura, escrita, adição).
- O tratamento de exceções é importante para lidar com erros de E/S.

Aula 10: Exercícios para testar conhecimentos

Resumo: Esta aula prática visa consolidar o conhecimento adquirido nas aulas anteriores, através da resolução de exercícios que envolvem todos os temas abordados: tipos de dados, variáveis, operadores, estruturas condicionais, loops, funções, módulos e strings.

Recomendações de Leitura:

- Revisar os materiais das aulas anteriores.
- Resolver exercícios de programação em Python.

Explicações Adicionais:

- A resolução de exercícios é fundamental para a fixação do conhecimento.
- Analise os erros e debug o código para identificar e corrigir problemas.
- Peça ajuda a colegas ou professores se tiver dificuldades em resolver os exercícios.

Aula 11: Programação orientada a objeto

Resumo: A aula inicia a introdução à Programação Orientada a Objetos (POO) em Python. São apresentados os conceitos de classes, objetos, atributos e métodos, explicando como utilizar a POO para modelar o mundo real e organizar o código de forma modular.

Recomendações de Leitura:

- **LARMAN, Craig. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. Prentice Hall, 2004.** (Capítulos sobre POO)
- Documentação oficial do Python sobre classes: https://docs.python.org/3/tutorial/classes.html

Explicações Adicionais:

- Classes são modelos para criar objetos.
- Objetos são instâncias de classes.
- Atributos são características dos objetos.
- Métodos são ações que os objetos podem realizar.

Aula 12: Herança

Resumo: A aula aprofunda o estudo da POO, explorando o conceito de herança. São apresentados exemplos de como criar classes filhas que herdam atributos e métodos de classes pais, reutilizando código e criando hierarquias de classes.

Recomendações de Leitura:

- Exemplos de código sobre herança em Python.
- Livros sobre POO que abordam o conceito de herança.

Explicações Adicionais:

- Herança permite criar classes mais especializadas a partir de classes mais gerais.
- A classe filha herda todos os atributos e métodos da classe pai.
- A classe filha pode adicionar novos atributos e métodos ou sobrescrever os existentes.

Aula 13: Lidando com erros

Resumo: A aula aborda o tratamento de erros em Python, apresentando o conceito de exceções e as estruturas try-except-finally. São apresentados exemplos de como capturar e tratar exceções para evitar que o programa seja interrompido inesperadamente.

Recomendações de Leitura:

- Documentação oficial do Python sobre exceções: https://docs.python.org/3/tutorial/errors.html
- Artigos sobre tratamento de exceções em Python.

Explicações Adicionais:

- Exceções representam erros que ocorrem durante a execução do programa.
- A estrutura `try-except` permite capturar e tratar exceções.
- O bloco `finally` é sempre executado, mesmo se ocorrer uma exceção.

Aula 14: Operador Python

Resumo: A aula explora o módulo "operator" em Python, uma biblioteca que fornece funções que correspondem aos operadores intrínsecos da linguagem (aritméticos, lógicos, de comparação). O uso desse módulo pode tornar o código mais flexível e, em alguns casos, mais conciso.

Recomendações de Leitura:

- Documentação do módulo "operator" em Python: https://docs.python.org/3/library/operator.html
- Exemplos de uso do módulo "operator".

Explicações Adicionais:

- Em vez de usar diretamente `+`, `-`, `*`, `/`, etc., pode-se usar `operator.add`, `operator.sub`, etc.
- A flexibilidade adicional vem da possibilidade de passar funções como argumentos para outras funções.
- Algumas operações, como a concatenação de strings (`+`), também possuem funções equivalentes no módulo.

Aula 15: Dados estruturados

Resumo: A aula aborda como trabalhar com dados estruturados em Python, que podem vir em diversos formatos (CSV, JSON, XML, etc.). O foco está nas bibliotecas padrão para ler, processar e transformar esses dados.

Recomendações de Leitura:

- Documentação da biblioteca "csv" em Python: https://docs.python.org/3/library/csv.html
- Documentação da biblioteca "json" em Python: https://docs.python.org/3/library/json.html

Explicações Adicionais:

- "csv" é para arquivos de valores separados por vírgulas, comuns para dados tabulares.
- "json" lida com dados formatados em JSON (JavaScript Object Notation), popular para APIs web.
- A biblioteca "xml.etree.ElementTree" (mencionada na documentação) lida com arquivos XML.

Aula 16: Explorando como conectar a web

Resumo: A aula explora a biblioteca "requests" em Python, que permite fazer requisições HTTP para interagir com serviços web. São abordados os diferentes tipos de requisições (GET, POST, PUT, DELETE) e como enviar dados e receber respostas de APIs web.

Recomendações de Leitura:

- Documentação da biblioteca "requests" em Python: https://requests.readthedocs.io/en/latest/
- Tutoriais sobre como utilizar a biblioteca "requests" para consumir APIs web.

Explicações Adicionais:

- "requests" simplifica a interação com a web, abstraindo a complexidade do protocolo HTTP.
- É importante entender os códigos de status HTTP (200, 400, 500, etc.) para identificar problemas na requisição.
- A biblioteca "json" é frequentemente utilizada para decodificar a resposta da API.

Aula 17: Como fazer Web Scrape

Resumo: A aula introduz o conceito de web scraping, a técnica de extrair dados de páginas web. Apresenta a biblioteca "BeautifulSoup", que facilita o parsing de HTML e XML, e como utilizá-la em conjunto com a biblioteca "requests" para coletar dados de websites.

Recomendações de Leitura:

- Documentação da biblioteca "BeautifulSoup": https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- Tutoriais e exemplos de código sobre web scraping com "BeautifulSoup".

Explicações Adicionais:

- Web scraping pode ser utilizado para coletar dados de sites que não oferecem APIs.
- É importante respeitar os termos de uso do site e evitar sobrecarregar o servidor.
- Web scraping exige cuidado para lidar com mudanças na estrutura do HTML do site.

Aula 18: Análise de dados usando pandas

Resumo: A aula explora a biblioteca "pandas" em Python, que fornece estruturas de dados de alto nível e ferramentas para análise de dados. São apresentados os DataFrames, que permitem armazenar e manipular dados tabulares de forma eficiente, e as principais funções para limpeza, transformação e análise de dados.

Recomendações de Leitura:

- Documentação da biblioteca "pandas": https://pandas.pydata.org/docs/
- Tutoriais sobre análise de dados com "pandas".

Explicações Adicionais:

- DataFrames são semelhantes a tabelas em bancos de dados relacionais ou planilhas.
- "pandas" oferece diversas funções para filtrar, ordenar, agrupar e agregar dados.
- A biblioteca "matplotlib" pode ser utilizada para criar visualizações dos dados.

Aula 19: EXAME FINAL
Avaliação geral do conteúdo do curso.

Este material de apoio é um guia para o estudo da disciplina, combinando teoria e prática para auxiliar no aprendizado e desenvolvimento de habilidades em Linguagem de Programação Python. Boa sorte!