Introdução

Olá! Iremos iniciar nossos estudos de Programação Orientada a Objeto! Como veremos nos próximos capítulos, essa nova estratégia de programação é bastante focada na modularização dos projetos. Sendo assim, você irá notar que os códigos serão bem diferentes do que você pode ter estudado em módulos mais introdutórios, e isso poderá impactar também a nossa forma de estudar.

## 1. Como estudar este material?

Uma recomendação que segue válida é sempre executar os códigos de exemplo. A grande diferença é que muitos dos códigos serão a criação de alguma classe. Experimente criar objetos de cada classe e experimentar com eles: modifique seus atributos, use seus métodos de diferentes maneiras, experimente fazer alterações pontuais nos métodos e veja como o comportamento dos objetos é afetado... E se nada disso fez sentido até o momento, fique tranquilo, tudo fará em breve! ;)

Não deixe de fazer também os exercícios propostos pelo seu professor. Criar as suas próprias classes é fundamental para desenvolver o tipo de raciocínio que você precisará para modelar problemas no futuro.

## 2. Onde executar os códigos?

Você pode seguir utilizando a IDE e/ou notebook que você já vinha usando em módulos anteriores, seguindo a orientação do seu professor. Caso algum dia você esteja sem acesso a um computador com suas ferramentas instaladas no horário da aula, segue algumas recomendações de editores online:

* [Repl.it](https://replit.com/languages/python3): uma IDE online bastante simples, com visualização da saída do código logo ao lado do editor.
* [OnlineGDB](https://www.onlinegdb.com/): uma IDE online com suporte a múltiplas linguagens de programação diferentes. Não se esqueça de selecionar "Python3" no menu antes de começar os seus trabalhos.
* [Jupyter](https://jupyter.org/try-jupyter/retro/notebooks/?path=notebooks/Intro.ipynb): O Jupyter oferece uma área em seu site para você experimentá-lo sem precisar instalar. Ele será executado direto no seu navegador e possui acesso às bibliotecas padrão de ciência de dados.
* [Google Colab](https://colab.research.google.com/): É um serviço de notebooks compatível com Jupyter, mas integrado à sua conta Google. O seu trabalho será salvo em seu Google Drive.

## 3. Material extra e referências

Além do material básico de Python, que segue válido (e será citado novamente aqui), acrescentamos material específico para amadurecer o seu raciocínio orientado a objeto.

### Sites

* [Wiki Python Brasil](https://wiki.python.org.br/ProgramacaoOrientadaObjetoPython): esse grupo publicou um manual inteiro em português sobre programação orientada a objeto em Python.
* [Refactoring Guru](https://refactoring.guru/pt-br/design-patterns): um site inteiro especializado em design patterns: modelagens já documentadas para situações específicas. Ele possui versão em português e exemplos em Python. Muito útil quando você já estiver dominando o básico deste módulo.
* [GeeksForGeeks](https://www.geeksforgeeks.org/python-oops-concepts/): um ótimo site de tutoriais de programação. Possui um material bacana e bastante resumido de programação orientada a objeto em Python. Disponível apenas em inglês.
* [PythonTutorial](https://www.pythontutorial.net/python-oop/): outro site focado em tutoriais, com uma seção bastante rica voltada exclusivamente para programação orientada a objetos. Apenas em inglês.
* [Documentação Oficial do Python](https://docs.python.org/pt-br/3/): disponível em português, é ponto de parada obrigatório para todos os programadores. Ela explica detalhadamente todos os recursos da linguagem e possui uma referência bastante completa de todas as bibliotecas incluídas na linguagem.
* [RealPython.com](https://realpython.com/): um site riquíssimo em cursos, artigos e tutoriais. Alguns são gratuitos, outros são pagos. Eles possuem uma newsletter que envia dicas de Python regularmente por e-mail.
* [StackOverflow](https://stackoverflow.com/): talvez você não tenha utilidade imediata para ele. Mas conforme você for programando e pesquisando erros, frequentemente cairá neste site. Ele é um site de perguntas e respostas sobre programação com filtros bastante eficientes para garantir a prevalência de perguntas relevantes e respostas corretas.

### Livros

* Código Limpo - Robert Cecil Martin ("Uncle Bob"): leitura obrigatória para quem pensa em fazer sistemas maiores. Apesar dos exemplos serem em Java, é útil para programadores de diversas linguagens.
* Pense em Python (2ª edição) - Allen B. Downey: esse livro é bastante introdutório, e possui uma linguagem extremamente acessível. É ideal para quem está começando e usa vários exemplos em código. Ele é disponibilizado gratuitamente e pode ser [lido online](https://penseallen.github.io/PensePython2e/). A versão impressa pode ser comprada no site da [Editora Novatec](https://novatec.com.br/livros/pense-em-python/).
* Think Python (2nd edition) - Allen B. Downey: é a versão original do livro acima. Quem tem facilidade com a língua inglesa pode preferir ler o original. Assim como na tradução, é possível [ler gratuitamente online](https://greenteapress.com/wp/think-python-2e/) ou adquirir a versão impressa no mesmo link.
* Python Basics (4th edition) - RealPython.com: disponível apenas em inglês, é um livro escrito pela equipe que mantém o site RealPython.com. Esse livro é bastante introdutório, mirando estudantes iniciantes. Mas ele percorre uma variedade muito maior de tópicos que o Think Python, e acaba oferecendo mais aplicações.
* Python Tricks - Dan Bader: disponível apenas em inglês, escrito por um dos editores do RealPython.com. Esse livro é um pouco mais avançado, e seria mais interessante para quem já chegou aqui sabendo Python ou então após a conclusão do módulo. Ele é um livro de aprofundamento, que irá ajudar quem já sabe o básico de Python a explorar mais a fundo os diferentes recursos que a linguagem oferece.
* Python Fluente - Luciano Ramalho: assim como o livro anterior, ele é recomendável para quem já está confortável com a linguagem e gostaria de se aprofundar. Foi escrito por um brasileiro e possui traduções para diversas línguas. É uma leitura obrigatória para programadores Python experientes.
* Toda a coleção do Al Sweigart: esse autor escreve diversos livros totalmente baseados em projetos e atividades. Eles são bem legais para complementar os estudos justamente pela oportunidade de ver tudo funcionando na prática. Seus livros são em inglês, mas estão todos disponíveis gratuitamente no próprio [site do autor](https://inventwithpython.com/), e assim como em outros casos, também podem ser comprados em versões impressas.
* Como Pensar Como um Cientista da Computação - Brad Miller e David Ranum: a tradução completa do livro em português pode ser lida gratuitamente [aqui](https://panda.ime.usp.br/pensepy/static/pensepy/index.html). Ele conta com um capítulo inteiro sobre programação orientada a objeto.
* How to Think Like a Computer Scientist: Interactive Edition - Brad Miller e David Ranum: a edição original em inglês do livro citado acima também pode ser lida gratuitamente [neste site](https://runestone.academy/ns/books/published/thinkcspy/index.html).

Há uma infinidade de sites, blogs e fóruns com bastante material interessante. Em alguns capítulos, citaremos alguns links específicos para que você possa se aprofundar em certos tópicos.