

# Contrato Pedagógico

Professor MSc Wylliams Barbosa Santos

wylliamss@gmail.com

<http://about.me/wylliams>

Modelagem de Programação Orientada a Objetos



# Agenda

- **Um passo à frente**
- **Objetivos**
- **Ementa**
- **Bibliografia**
- **Metodologia**
- **Avaliações**
- **Responsabilidades**
- **Assinatura do Contrato**
- **Calendário**



**UFRPE**

Universidade  
Federal Rural  
de Pernambuco



**“Um passo à frente e você não  
está mais no mesmo lugar”**

1º  
Semestre

- Introdução à Programação

2º  
Semestre

- Laboratório de Programação
- Algoritmos e Estruturas de Dados

3º  
Semestre

- Modelagem e Programação Orientada a Objetos
- Fundamentos de Engenharia de Software

# Objetivos

**Modelagem conceitual: Abstração X Representação. O Modelo de Objetos: Classes e Objetos, Comunicação por troca de mensagens. Herança e Polimorfismo. Programação OO. Técnicas e Métodos.**



- Modelagem de Sistemas
  - UML
  - Abstração
  - Representação
- Programação Orientada a Objetos
  - Conceito
  - Fundamentos de programação orientada a objetos
  - Classes e Instâncias
  - Super classe e subclasse
  - Duração e escopo
  - Métodos e mensagens (construtores e destrutores)
  - Composição e Herança
  - Polimorfismo
  - Componentes gráficos e sua organização
  - Tratamento de eventos
  - Threads
  - Manipulação de exceções

- **Implementação de Programas Utilizando os Conceitos e Técnicas da Orientação a Objetos**
  - Estruturas de armazenamento para variáveis de instâncias
- **Vinculações dinâmicas de mensagens e métodos**
- **Tipos e estruturas dinâmicas de dados**
  - Conceito
  - Tipos
  - Características e utilizações de listas, pilhas e filas
- **Depuração e Documentação de Programas**
  - Depuração modular
  - Teste de integração
  - Validação/ dados de controle
- **Introdução à documentação: diagrama de requisitos, diagrama de classes, diagrama de seqüência**

- **Cormen, Thomas H. et. al. Algoritmos: Teoria e Prática. Editora Campus, 2002.**
- **Riccioni, Paulo Roberto. Introdução a Objetos Distribuídos com CORBA. Visual Books, 2000.**
- **Melo, A.C.V. e Silva, F.S.C. Princípios de Linguagem de Programação. Edgar Blúcher Ltda, 2003.**
- **Van Rossum, Guido. Tutorial de Python. Disponível gratuitamente em <http://python.org/>, 2004.**
- **Conallen, Jim. Desenvolvendo Aplicações Web com UML. Editora Campus, 2003.**
- **Deitel, Harvey M. et. al. Java como Programar. Bookman, 2003.**

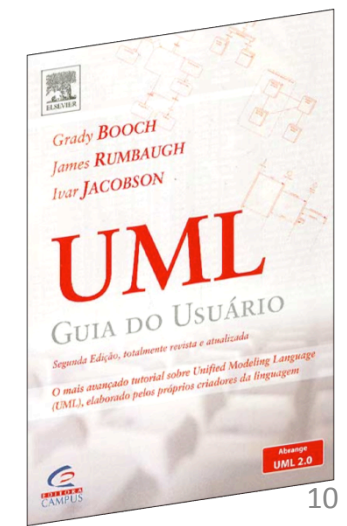
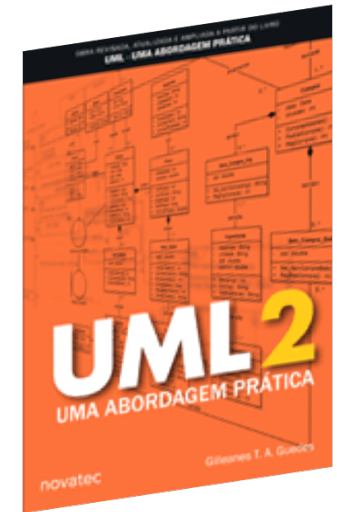
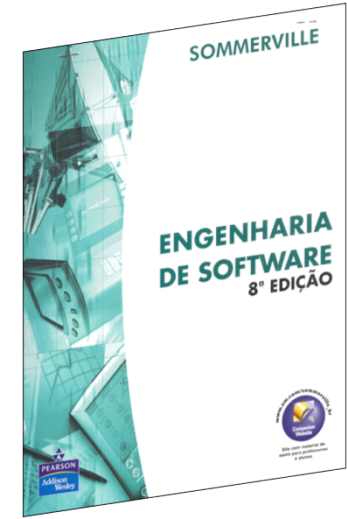


# Bibliografia

- Deitel, Harvey M. et. al. C++ How to Program. Prentice Hall, 2002.
- Nieto, T. R. Internet & World Wide Web. Como Programar. Bookman, 2003.
- Deitel, Harvey M. et. al. XML Como Programar. Bookman, 2003.
- Ziviani, Nivio. Projeto de Algoritmos. Editora Nova Fronteira, 2004.

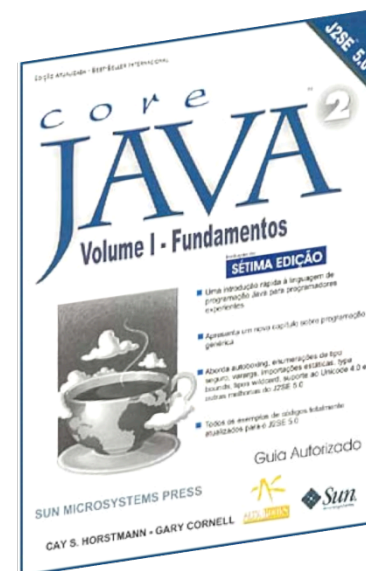
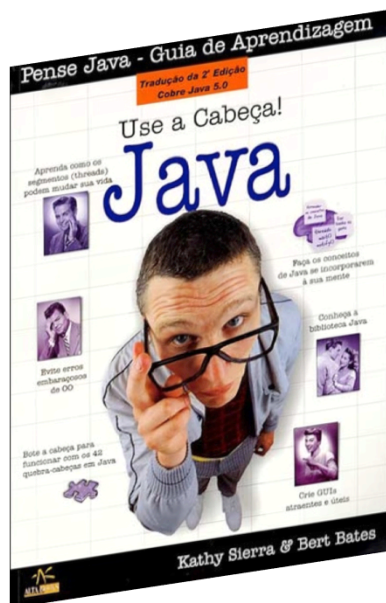


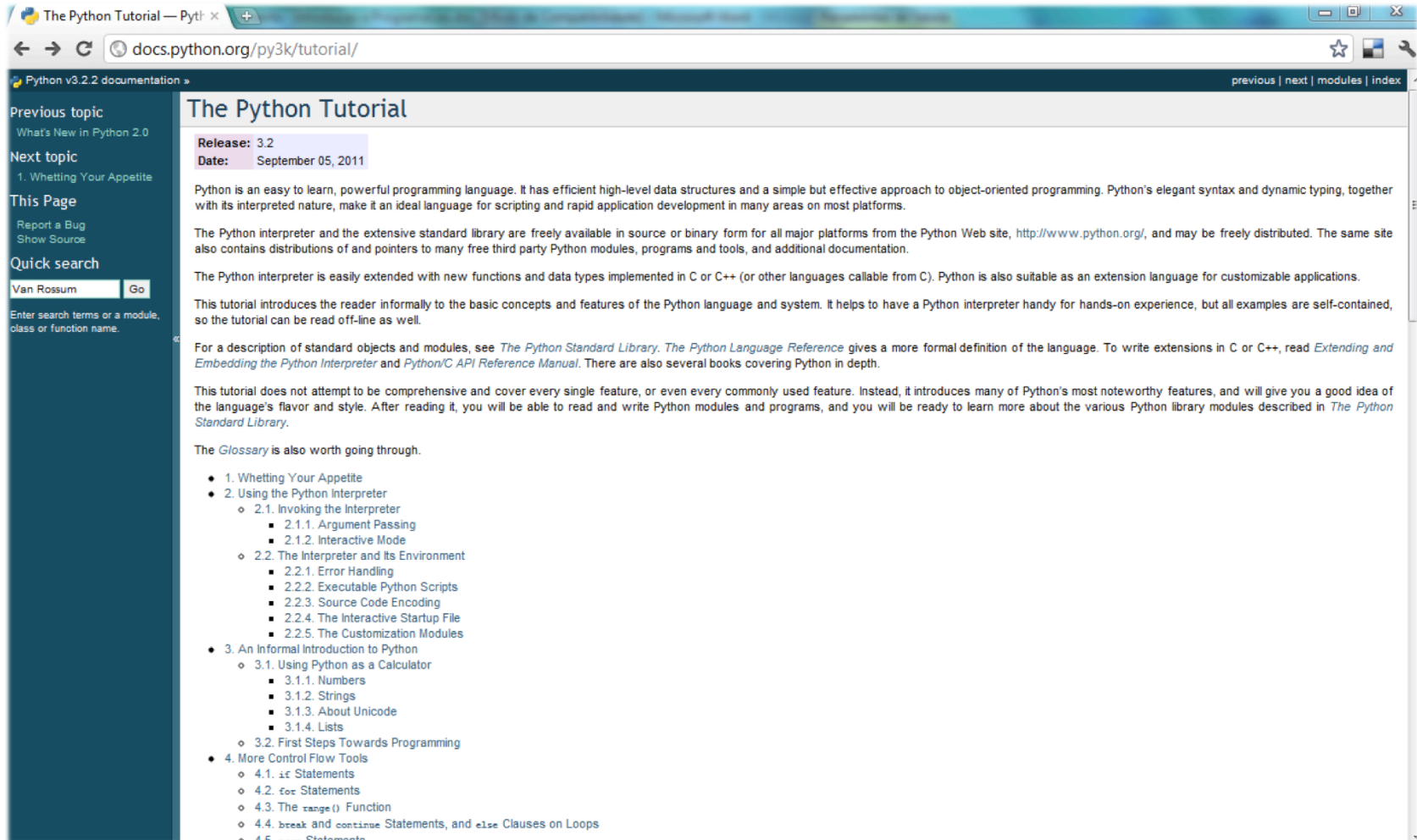
# Bibliografia





# Bibliografia





The screenshot shows a web browser window displaying the Python 3.2.2 documentation page titled "The Python Tutorial". The browser's address bar shows the URL `docs.python.org/py3k/tutorial/`. The page content includes:

- Release:** 3.2
- Date:** September 05, 2011
- Introduction:** Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.
- Availability:** The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, <http://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.
- Extensibility:** The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.
- Self-contained:** This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.
- References:** For a description of standard objects and modules, see *The Python Standard Library*. *The Python Language Reference* gives a more formal definition of the language. To write extensions in C or C++, read *Extending and Embedding the Python Interpreter* and *Python/C API Reference Manual*. There are also several books covering Python in depth.
- Scope:** This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in *The Python Standard Library*.
- Navigation:** The *Glossary* is also worth going through.
- Table of Contents:**
  - 1. Whetting Your Appetite
  - 2. Using the Python Interpreter
    - 2.1. Invoking the Interpreter
      - 2.1.1. Argument Passing
      - 2.1.2. Interactive Mode
    - 2.2. The Interpreter and its Environment
      - 2.2.1. Error Handling
      - 2.2.2. Executable Python Scripts
      - 2.2.3. Source Code Encoding
      - 2.2.4. The Interactive Startup File
      - 2.2.5. The Customization Modules
  - 3. An Informal Introduction to Python
    - 3.1. Using Python as a Calculator
      - 3.1.1. Numbers
      - 3.1.2. Strings
      - 3.1.3. About Unicode
      - 3.1.4. Lists
    - 3.2. First Steps Towards Programming
  - 4. More Control Flow Tools
    - 4.1. `if` Statements
    - 4.2. `for` Statements
    - 4.3. The `range()` Function
    - 4.4. `break` and `continue` Statements, and `else` Clauses on Loops
    - 4.5. `pass` Statements



**Metodologia**

**Teoria**

**+**

**Prática**



- 1ª Avaliação
  - Projeto e exercícios (0,0 – 5,0)
  - Prova (0,0 – 5,0)
- 2ª avaliação
  - Projeto (0,0 – 10,0)
- 3ª avaliação
  - Prova (0,0 – 10,0)
- Final
  - Prova (0,0 – 10,0)

**Re** **professor** **ponsabi** **alunos** **idade**

- Análise de erros;
- Leitura de artigos;
- Projeto Orientado a Objetos;





# Metodologia



- Repositório da disciplina: [www.bsi.ufrpe.br](http://www.bsi.ufrpe.br)
- Grupo:  
[ufrpe\\_modelagem\\_programacao\\_oo@googlegroups.com](mailto:ufrpe_modelagem_programacao_oo@googlegroups.com)  
[http://groups.google.com/group/ufrpe\\_modelagem\\_programacao\\_oo](http://groups.google.com/group/ufrpe_modelagem_programacao_oo)
- Faltas
- Prazos

