



Algoritmos e Programação de Computadores

Introdução & Plano de Desenvolvimento

Profa. Sandra Avila

Instituto de Computação (IC/Unicamp)

MC102 Turma KLMN, 28 Fevereiro, 2018

EXPRESSO

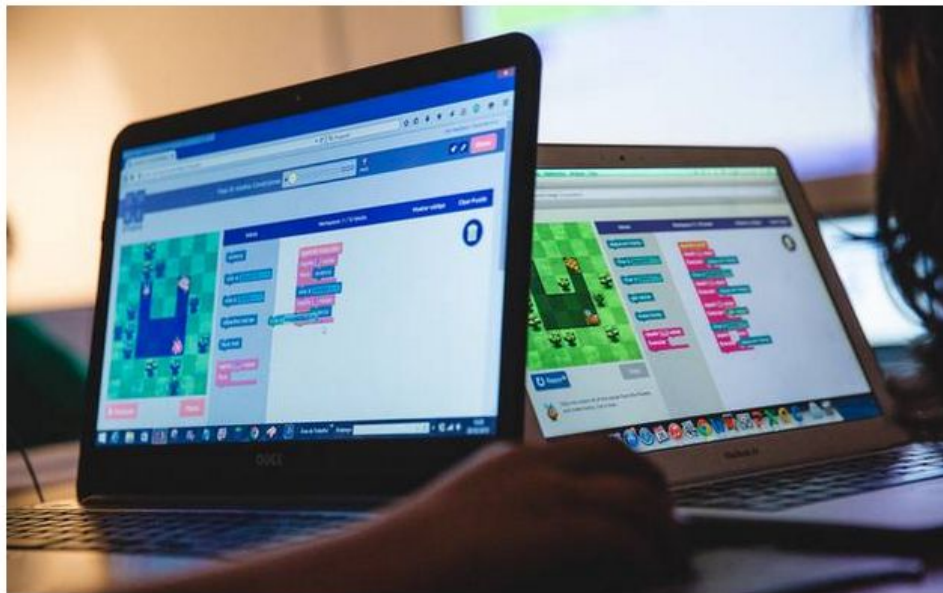
Por que programar é o novo 'aprender inglês'

Beatriz Montesanti 02 Abr 2017 (atualizado 04/Abr 18h17)

Escolas no Brasil e no exterior começam a adotar programação entre as disciplinas do currículo



FOTO: OLABI MAKERSPACE/FLICHR / CREATIVE COMMONS



📊 UMA A CADA QUATRO ESCOLAS AMERICANAS OFERECEM AULAS DE PROGRAMAÇÃO

GRÁFICOS

CARREIRA - VOCÊ S/A

Por que aprender programação é tão crucial quanto saber ler

Pesquisador do MIT Media Lab defende que aprender programação é importante para qualquer profissional

Por **Rafael Carvalho**
© 21 ago 2015, 11h00



Programação: caminho para tornar as pessoas fluentes em novas tecnologias, segundo especialista do MIT (Thinkstock/)



TECNOLOGIA

Por que é tão importante aprender programação?

Escrito por HostGator Brasil

25 de janeiro de 2018 | Comente

Diferente do que muito gente pensa, você não precisa ser um gênio para aprender a programar. Lembra de quando você não sabia ler? As letras eram como desenhos ou rabiscos e pra você não formavam palavras, muito menos frases. Mas, aos poucos você

O que é Programação
de Computadores?



Neste curso, vamos aprender a criar **algoritmos** e **programas** para resolver problemas.



Bolo de Chocolate Super Rápido

Ingredientes:

- 4 colheres de sopa de chocolate em pó
- 2 colheres de sopa de margarina
- 3 xícaras de chá de farinha
- 2 xícaras de chá de açúcar
- 1 xícara de chá de leite
- 4 ovos
- 2 colheres de sopa de fermento em pó (não muito cheias)



Modo de Fazer:

- Misture todos os ingredientes no liquidificador e bata por cerca de 4 minutos.
- Desligue o liquidificador, acrescente o fermento em pó, misture bem com a colher, e coloque a mistura em uma forma untada.
- Leve ao fogo (pré-aquecido) por cerca de 40 minutos. Cubra com brigadeiro e chocolate granulado.

Por que é importante
Aprender a Programar?

Por que é Importante?

- Resolução de problemas
- Visão de futuro
- Senso crítico e criatividade
- Mercado de trabalho

Resolução de Problemas

“Usar a programação como ferramenta para encontrar respostas e soluções para os desafios **acrescenta muito na capacidade de raciocínio lógico** — e essa característica é essencial em qualquer profissional.”

Visão de Futuro

“Saber programação significa entender como a tecnologia funciona. Esse conhecimento não só vai expandir a sua visão de futuro, como também **vai torná-lo capaz de trabalhar melhor em meio a tanta inovação.**”

Senso Crítico e Criatividade

“Ao aprender como as tecnologias funcionam, **ganhamos maior senso crítico, liberdade e criatividade**. Isso porque, em vez de se conformar com as aplicações prontas, podemos criá-las para atender às nossas necessidades.”

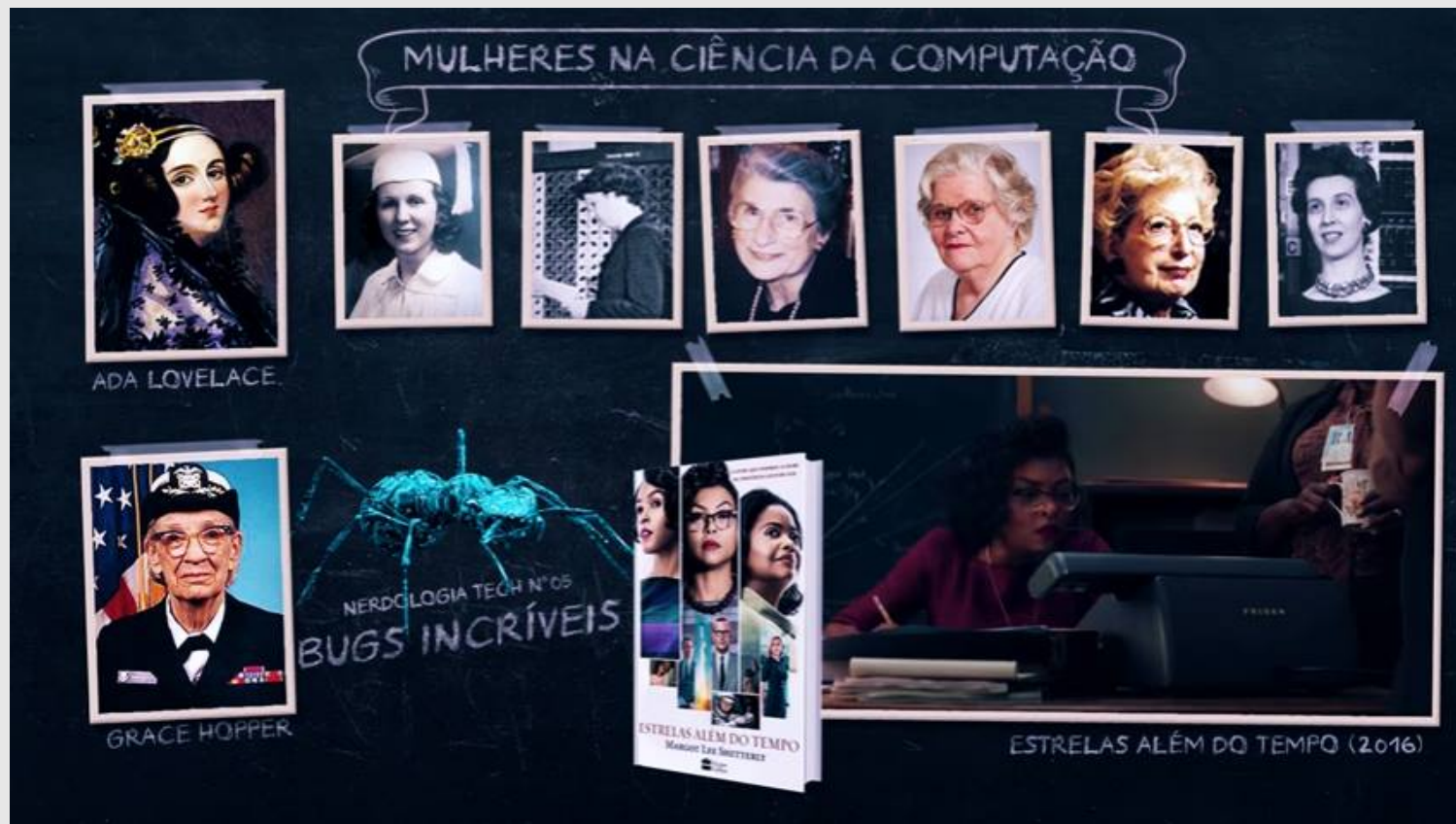
Mercado de Trabalho

“Todas as habilidades que estão embutidas no aprendizado de programação são exigidas pelas empresas na hora de contratar um profissional:

- Clareza, rapidez e fluidez nos pensamentos
- Raciocínio lógico
- Organização

Um pouco da história

“O primeiro computador”: <https://youtu.be/wyZPsCQd7Uo>



Plano de Desenvolvimento

Plano de Desenvolvimento

- Informações Básicas
- Critérios de Avaliação
- Atendimento & Referências

Informações Básicas

- Carga horária da disciplina
- Ementa
- Linguagem de programação
- Divulgação de informações



Carga Horária da Disciplina

- Duração: 90 horas
- Distribuição semanal
 - 4 horas de aulas teóricas
 - 2 horas de aulas de laboratório
- Frequência mínima: 75% (veja o Regimento Geral de Graduação)
<https://www.dac.unicamp.br/portal/graduacao/regimento-geral>

Ementa

- Conceitos básicos de organização de computadores
- Construção de **algoritmos** e sua representação em pseudocódigo e linguagens de alto nível
- Desenvolvimento sistemático e **implementação de programas**
- Estruturação, depuração, testes e documentação de programas
- **Resolução de problemas**

Linguagem de Programação

- Python (versão 3)
- Criada por Guido Van Rossum
- Primeiro *release* em 1991
- Comunidade dinâmica
- Muitas bibliotecas e recursos disponíveis



Divulgação de Informações

- Página web da disciplina
 - <http://www.ic.unicamp.br/~mc102>
- Páginas web específica para a turma KLMN
 - <http://www.ic.unicamp.br/~sandra/teaching/2018-1-mc102klmn>
- E-mails encaminhados ao endereço fornecido pela DAC
- Fique atenta(o)!

Critérios de Avaliação

- Atividades conceituais
- Tarefas de laboratórios
- Provas teóricas



Atividades Conceituais

- Visam indicar o **grau de compreensão** dos conceitos básicos
- **Questionários Online:**
 - Exercícios de múltipla escolha
 - Podem ser realizados em laboratório ou em casa
- **Tarefas Presenciais:**
 - Realizadas em laboratório, sob supervisão
 - Aguarde divulgação de datas e outras orientações

Atividades Conceituais

- Ferramenta de apoio:
 - **Moodle** (<http://www.ggte.unicamp.br/eam>)
 - Login: email da DAC
 - Curso: MC102 - 1S2018 - Algoritmos e Programação de Computadores

Atividades Conceituais — Avaliação

- Proposta de n atividades ao longo do semestre
- Nota AC_i , $1 \leq i \leq n$, será proporcional ao número de questões respondidas pela(o) aluna(o)
- Questionários Online: Peso 1
- Tarefas Presenciais: Peso 3
- Média M_{AC} : média ponderada das atividades conceituais

Tarefas de Laboratório

- Implementação de problemas em Python
- Enunciado disponível na página da disciplina, com orientações, restrições e prazos
- Auxílio ao desenvolvimento durante as aulas de laboratório
- Correção automática
 - Ferramenta de apoio: SuSy
 - **Testes abertos & Testes fechados**

Tarefas de Laboratório — Avaliação

- Proposta de m tarefas
- Nota L_i , $1 \leq i \leq m$, calculada de acordo com o estipulado no enunciado da tarefa i .
- Peso $LP_i \in \{1, 2, 3\}$
- Média M_L : média ponderada das tarefas de laboratório

Provas Teóricas

Prova	Peso	Data	Horário
P_1	2	20 de Abril	14h–16h
P_2	3	20 de Junho	14h–16h

- Individuais e sem consulta
- Média M_p : média ponderada das provas teóricas

Critérios de Avaliação

- Média ponderada dos elementos:

$$M_{\text{Elem}} = (0.6 \times M_P) + (0.3 \times M_L) + (0.1 \times M_{AC})$$

M_P : média ponderada das **provas teóricas**

M_L : média ponderada das **tarefas de laboratório**

M_{AC} : média ponderada das **atividades conceituais**

- Média antes do exame: $M = \min(M_{\text{Elem}}, M_P, M_L)$

Critérios de Avaliação — Média Final F

Frequência $\geq 75\%$:

- $M \geq 5$: **aprovação** por nota e frequência com $F = M$
- Se $2.5 \leq M < 5$:
 - Exame E no dia 13 de julho das 14h–16h, $F = (M+E)/2$
 - $F \geq 5.0$: **aprovação** por nota e frequência
 - $F < 5.0$: reprovação por nota
- Se $M < 2.5$: reprovação por nota com $F = M$

Frequência $< 75\%$: reprovação por frequência com $F = M$

Observações

- Nos **dias de prova** será necessária a apresentação de **documento oficial com foto**.
- Não há possibilidade de troca de horário de provas e/ou do exame final.
- Não haverá reposição de tarefas de laboratório.
- Todas as tarefas são individuais.
- Qualquer **tentativa de fraude** implicará em nota **0.0 (zero)** na disciplina para todas as pessoas envolvidas.
- O sistema SuSy possui detector de plágio.

Apoio

- Atendimento (Dúvidas)
- Material de apoio & Referências bibliográficas



Atendimento

- Equipe de monitoras(es)
 - PEDs: alunas(os) de pós-graduação
 - PADs: alunas(os) de graduação
- Atendimento durante as aulas de laboratório
- Atendimento extra: veja página web da disciplina
- Aulas extras de revisão: veja página web da disciplina

Atendimento

- Equipe
 - Celso Augusto R. L. Brennand (PED, Doutorado)
 - Alceu Emanuel Bissoto (PED, Mestrado)
 - Matheus Martins Susin (PED, Mestrado)
 - Renato Noronha Máximo (PAD, Graduação)
 - William Ryuji Massuda (PAD, Graduação)

Material de Apoio & Referências Bibliográficas

“How to Think Like a Computer Scientist: Interactive Edition”, de Brad Miller e David Ranum: <https://runestone.academy/runestone/static/thinkcspy/index.html>



How To Think Like a Computer Scientist

Chapters ▾



How to Think Like a Computer Scientist: Interactive Edition

About this Project

Table of Contents

- [Assignments](#)
- [1. General Introduction](#)
 - [1.1. The Way of the Program](#)
 - [1.2. Algorithms](#)
 - [1.3. The Python Programming Language](#)
 - [1.4. Executing Python in this Book](#)
 - [1.5. More About Programs](#)
 - [1.6. What is Debugging?](#)
 - [1.7. Syntax errors](#)
 - [1.8. Runtime Errors](#)
 - [1.9. Semantic Errors](#)
 - [1.10. Experimental Debugging](#)
 - [1.11. Formal and Natural Languages](#)
 - [1.12. A Typical First Program](#)
 - [1.13. Comments](#)

Material de Apoio & Referências Bibliográficas

“How to Think Like a Computer Scientist: Interactive Edition”, de Brad Miller e David Ranum: <https://panda.ime.usp.br/pensepy/static/pensepy>

[Como pensar como um Cientista da Computação](#) »

[next](#) | [index](#)

Como Pensar Como um Cientista da Computação



Aprendendo com Python: Edição interativa (usando Python 3.x)

Tradução do livro “How to Think Like a Computer Scientist: Interactive Version”, de Brad Miller e David Ranum

Preâmbulos

- [Aviso de direitos autorais](#)
- [Copyright Notice](#)
- [Prefácio](#)
- [Prefácio da primeira e segunda edições](#)
- [Prefácio à terceira edição](#)
- [A edição local de Rhodes \(RLF\)](#)
- [Prefácio à Edição Interativa](#)
- [Para que fazer o log in?](#)

Table Of Contents

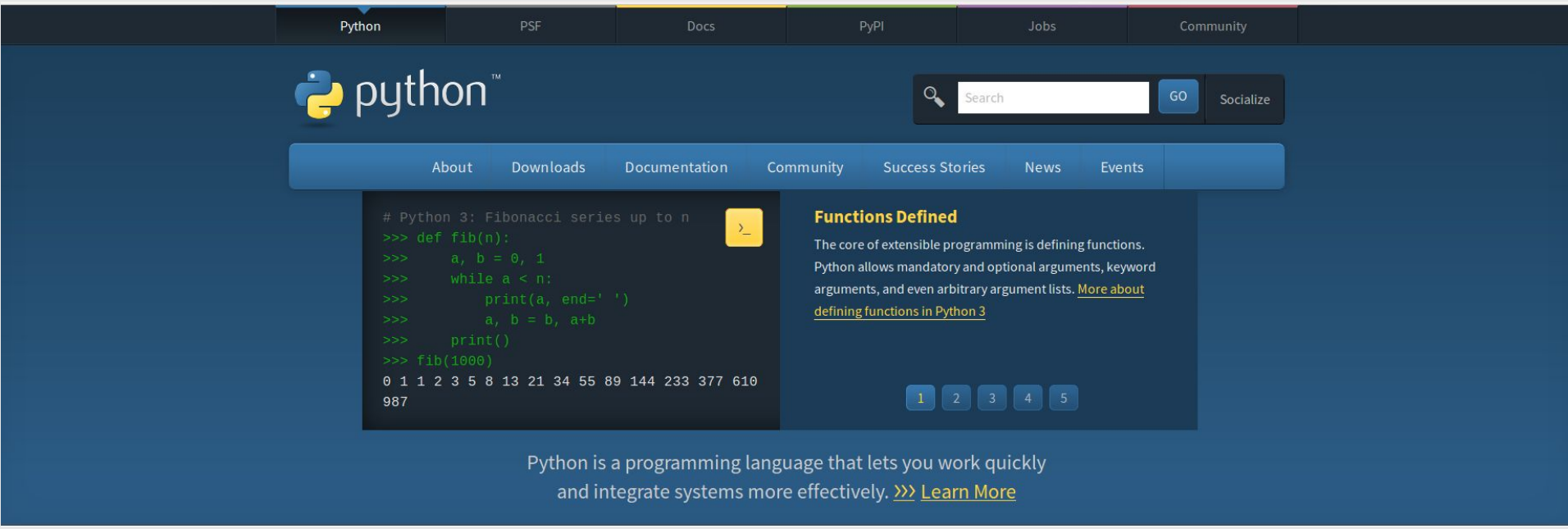
Como Pensar Como um Cientista da Computação

- **Aprendendo com Python: Edição interativa (usando Python 3.x)**
 - Preâmbulos
 - Introdução
 - Conceitos de Python
 - Programando uma Tartaruga em Python
 - Módulos do Python
 - Funções
 - Seleção
 - Mais Sobre Iteração
 - Strings
 - Listas
 - Arquivos
 - Dicionários
 - Recursão
 - Definindo Classes
 - Labs
 - Appendices
 - Índices e tabelas

[Next topic](#)

Material de Apoio & Referências Bibliográficas

Veja material nas páginas oficiais: www.python.org



The image shows a screenshot of the Python.org website. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is a search bar with a magnifying glass icon, a 'GO' button, and a 'Socialize' button. A secondary navigation bar contains links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a code editor with a Python 3 script for calculating the Fibonacci series up to n. The code is as follows:

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
987
```

To the right of the code editor is a section titled "Functions Defined" with the text: "The core of extensible programming is defining functions. Python allows mandatory and optional arguments, keyword arguments, and even arbitrary argument lists. [More about defining functions in Python 3](#)". Below this text are five numbered buttons (1-5). At the bottom of the page, a footer states: "Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)".

Material de Apoio & Referências Bibliográficas

Veja material nas páginas oficiais: www.python.org.br

Python Brasil Impressiona-se ▾ Inicie-se ▾ Aprenda mais ▾ Participe ▾

APyB ▾



python[™]brasil

A comunidade Python Brasil reúne grupos de usuários em todo o Brasil interessados em difundir e divulgar a linguagem de programação.

Fork me on GitHub

Impressiona-se »

Inicie-se »

Aprenda mais »

Participe »

APyB »

Programador(a)

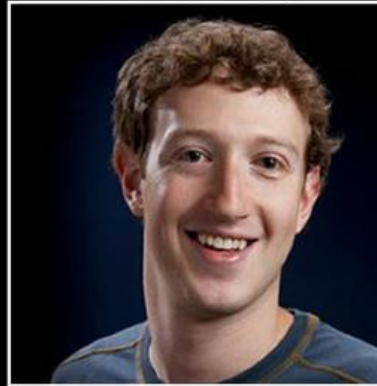
Como os outros me veem



Como meus pais me veem



Como eu me vejo



Como meu chefe me ve

