



25070 - Construção de Algoritmos e Programação

Aula 1 - 18/03/2019

Joice Otsuka

Apresentação da Disciplina e Fundamentos da Computação

Parte 1 - Apresentação da disciplina

**O que vamos aprender
nesta disciplina?**

**Algoritmos,
Programação ...**

Solucionar problemas!!

**Construir algoritmos e
programar são formas
de aprender a
solucionar problemas!!**



Solução de problemas [Wentworth et al, 2012]

- Entender o problema, formular questões
- Pensar criativamente sobre soluções possíveis
- **Expressar uma solução de forma clara e precisa**



Construção de algoritmos e **programação** são excelentes exercícios para **solução de problemas**



Vídeo no YouTube:

Por que todos deveriam aprender a programar? <https://youtu.be/mHW1Hsqlp6A>



Construção de algoritmos e programação

- **Entendimento do problema**
 - Identificação das condições, as restrições, os requisitos
 - Avaliação do estado inicial e do estado final
- **Construção de algoritmos para solucionar o problema**
 - Definição dos passos de uma solução
- **Programação**
 - Codificação da solução em uma linguagem de programação



Objetivos da disciplina

- **Entender problemas:** aprender a abordar um problema , identificar seus requisitos, condições
- **Propor soluções** para o problema **na forma de algoritmos**
 - Definição dos passos claros e precisos de uma solução
- **Codificação de um programa** que implemente a solução proposta
- **Primar pela qualidade e organização** dos algoritmos e programas



Conteúdo da disciplina

- Construção de algoritmos
 - Solução de problemas por refinamentos sucessivos
 - Soluções estruturadas
 - Estruturação de dados
- Programação
 - Codificação em linguagem C
 - Documentação do código



Planejamento da disciplina

Cronograma

<https://ava.ead.ufscar.br/mod/resource/view.php?id=704152>

Plano de ensino

<https://ava.ead.ufscar.br/mod/resource/view.php?id=704159>



Avaliação

- Componentes de avaliação
 - 4 Provas escritas (70%)
 - P1 e P3: 14%
 - P2 e P4: 21%
 - Trabalhos práticos (15%)
 - 1 Projeto final (15%)
- Todas avaliações receberão notas de 0,0 a 10,0.



Datas previstas

- Prova 1: 10 de abril;
- Prova 2: 29 de abril;
- Prova 3: 15 de maio;
- Prova 4: 24 de junho;
- Trabalhos ao longo da disciplina: AVA, URI, REMAR..
- Projeto final: 3 de julho.



Ambientes de apoio

- Ambiente Virtual de Aprendizagem (AVA) da UFSCar
 - <https://ava.ead.ufscar.br/>
- URI online Judge
 - <https://www.urionlinejudge.com.br/>



Reflexão e auto-avaliação contínua

- Estou fazendo a minha parte? O que posso fazer para melhorar?
- Estou aprendendo?
- Consigo aplicar o que estou aprendendo?
- Como posso ajudar meus colegas?
 - Criem grupos de estudos!!! 😊👤
 - Discutam soluções, tirem dúvidas, ensinem, compartilhem descobertas...
 - NÃO copiem, NÃO permitam cópias ... isso não ajuda!



Reflexão e auto-avaliação contínua

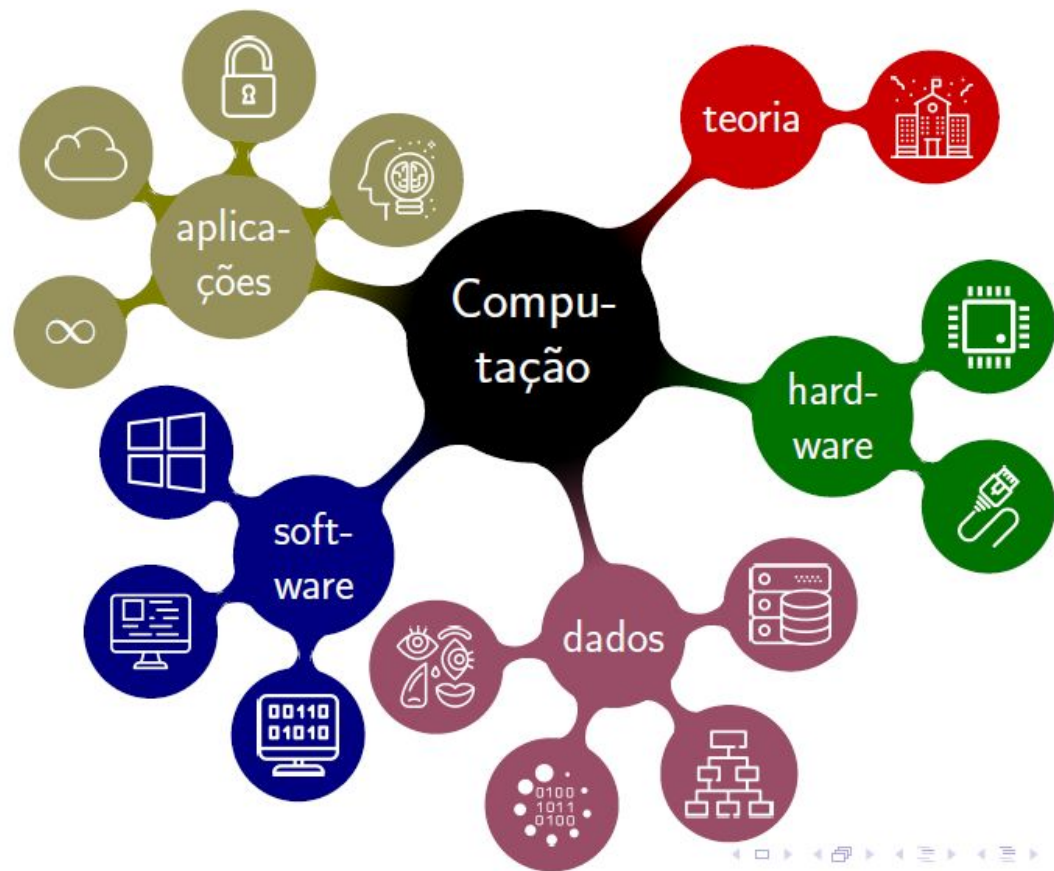
- APRENDER requer participação ATIVA, dedicação, organização, maturidade!
- Aproveitem bem o curso, aproveitem bem esse período de formação para APRENDER A APRENDER, para CRESCER!
- SUA FORMAÇÃO DEPENDE DE VOCÊ!!!
- Aproveite cada oportunidade de APRENDER!!!
- Algumas ferramentas:
 - Diário de bordo
 - Pesquisas de expectativas, experiência...

Parte 2 - Fundamentos da Computação



O que é computação?

- São computadores?
- São programas?
- São dados?
- Tudo isso e muito mais....



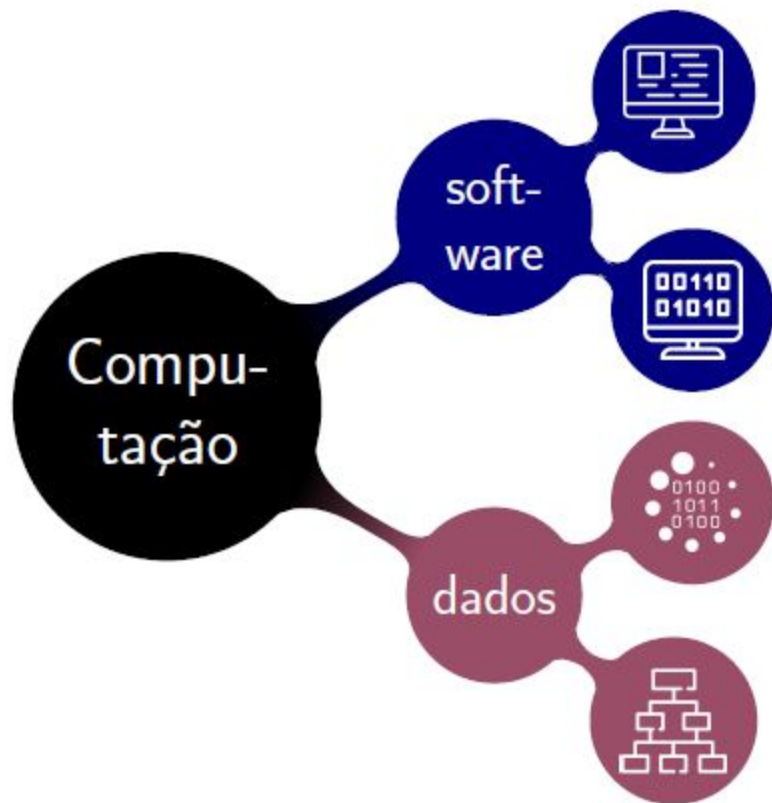
Teoria: computabilidade, complexidade, autômatos..

Hardware: organização, comunicação

Dados: abstração, representação, organização e banco de dados

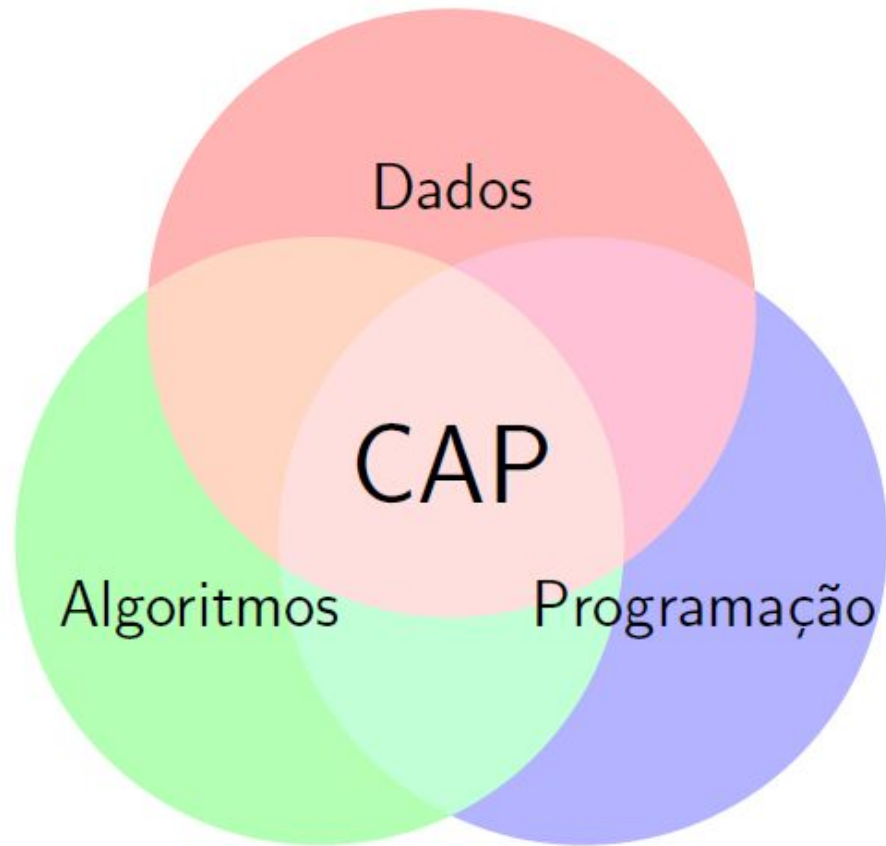
Software: sistemas operacionais, algoritmos, programação, engenharia de software, IHC..

Aplicações: Inteligência artificial, segurança, nuvem, mobilidade, big data...



Foco da disciplina CAP:

- Algoritmos
- Programação
- Representação de dados



Algoritmos



Algoritmo

- Um procedimento **passo a passo** para a **solução de um problema**
[Medina 2005]
- **Sequência de passos** que visam **atingir um objetivo** bem definido
[Forbellone 2005]
- É um acontecimento que, a partir de um **estado inicial**, após um período de **tempo finito**, produz um **estado final previsível e bem definido**
[Farrer, H. et. al., 1989]

Exemplo: Receita de bolo de chocolate

Ingredientes:

- 3 ovos inteiros
- 1/2 xícara de óleo
- 2 xícaras de leite
- 1 e 1/2 xícara de açúcar
- 1 xícara de nescau
- 3 xícaras de trigo
- 1 colher sopa fermento para bolo

Preparo:

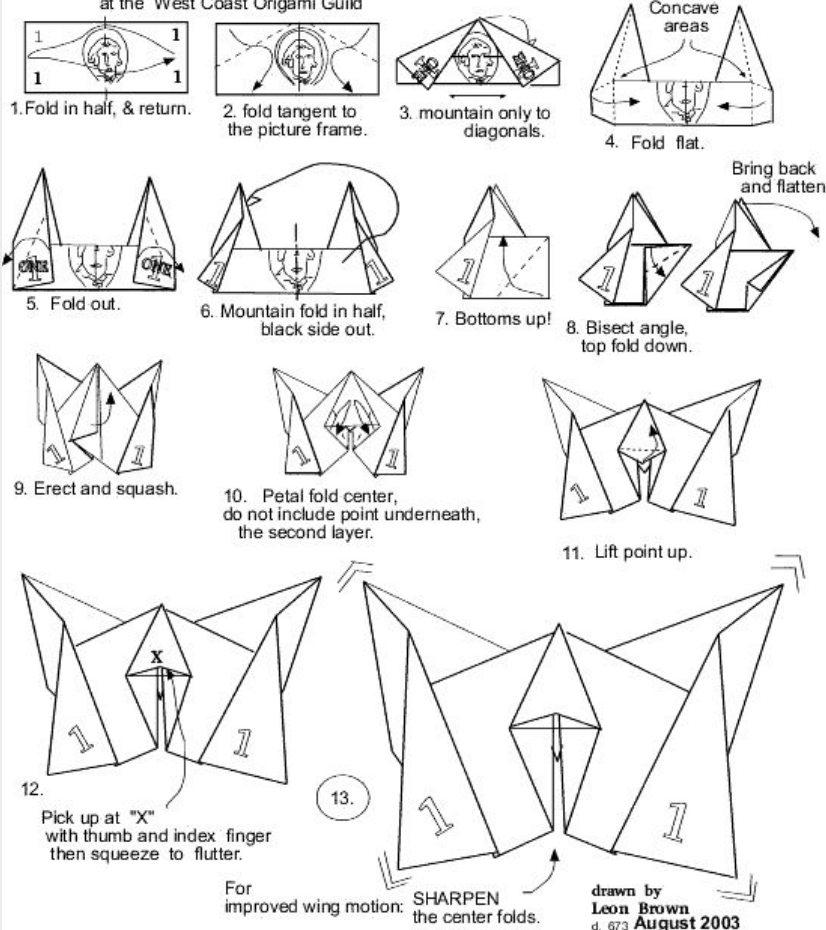
1. Coloque no liquidificador todos os ingredientes líquidos e bata-os bem.
2. Em seguida, acrescente o restante dos ingredientes e bata.
3. Coloque numa forma untada e enfarinhada.
4. Leve para assar em forno a 180°C, por 30 a 40 minutos, aproximadamente.



\$ BUTTERFLY , fluttering

modified by **John Vandrisan**
at the West Coast Origami Guild

Creator...**Robert Neale**



Exemplo: Origami

- Estado inicial: folha aberta
- Estado final: origami pronto
- Algoritmo: sequência de dobraduras



Figura por deviantart.com, de Won Park



Algoritmos informais

Como vir à aula? Escreva um algoritmo informal para orientar uma pessoa a vir à aula de CAP.



Algoritmos computacionais

- Quais as diferenças entre estes algoritmos?
- Todos atingem o objetivo?
- Qualquer um poderia executar qualquer dos algoritmos?



Algoritmos informais

- Escreva um algoritmo informal para orientar uma pessoa a ir do DC até o RU, fazendo uma rota a pé.

Quando terminar, mostre para o/a colega ao lado e verifique se ele/a compreende todos os passos e se os considera adequados



Algoritmos

- Podem haver vários algoritmos para resolver um mesmo problema
- Quem for executar um algoritmo precisa entender todos os passos
- O ideal são algoritmos que resolvam o problema com o menor custo e esforço



Algoritmos computacionais

- Seria possível elaborar algoritmos para computadores utilizando linguagem natural?
 - Pouca rigidez sintática e semântica
 - Problemas de ambiguidade
- E utilizando linguagem de programação?
 - Sim, mas pode ser complexo
- Há uma solução intermediária?
 - Fluxogramas
 - Pseudocódigo



Vídeo YouTube

Seu cérebro pode resolver algoritmos - David J. Malan

<https://youtu.be/6hfOvs8pY1k>

Programação



Programação

- Tradução de um algoritmo para um programa de computador
- Programa é uma **sequência de instruções** que especifica como **executar uma tarefa** [Wentworth et al, 2012] e **que pode ser executada por computador**



Programa de computador

- Quase todos os programas possuem o seguinte conjunto de instruções:
 - **ENTRAR:** Pegar dados do teclado, de um arquivo ou de algum outro dispositivo de entrada.
 - **SAIR:** Mostrar dados na tela ou enviar dados para um arquivo ou outro dispositivo de saída.
 - **CALCULAR:** Executar operações básicas.
 - **EXECUTAR CONDICIONALMENTE:** Checar certas condições e executar a sequência apropriada de instruções.
 - **REPETIR:** Executar alguma ação repetidamente, normalmente com alguma variação.



O que é um programa de computador?

- Um programa utiliza uma **linguagem de programação** para especificar sequências de instruções para um computador
- Linguagens de programação são **linguagens formais** que foram desenvolvidas para expressar como realizar comandos para computadores [Wentworth et al, 2012].



Linguagens formais

- Linguagens que foram projetadas para aplicações específicas
 - Ex: notação matemática, notação química, linguagens de programação
- Rigidez na sintaxe
 - Conjunto bem limitado de símbolos e estrutura
 - Exige construções muito bem definidas
- Rigidez semântica
 - Não permite ambiguidades na interpretação
 - Cada expressão deve ter uma única interpretação



Linguagens formais

- Regras de sintaxe
 - Relacionadas aos símbolos (quais símbolos são válidos)
 - Relacionadas à estrutura (como os símbolos são organizados)
- Regras semânticas
 - Relacionadas à interpretação dos símbolos de uma linguagem



Linguagens naturais x formais

- Exemplo 1 (linguagem natural):
 - Você vai ao churrasco?
 - Tu vais ao churrasco?
 - Vc vai ...?
 - Cê vai ...
- Exemplo 2 (linguagem natural):
 - Ouvi o barulho da porta
- Exemplo 2 (linguagem formal - Matemática):
 - $3 + 3 = 6$
 - $3 = +6\$$



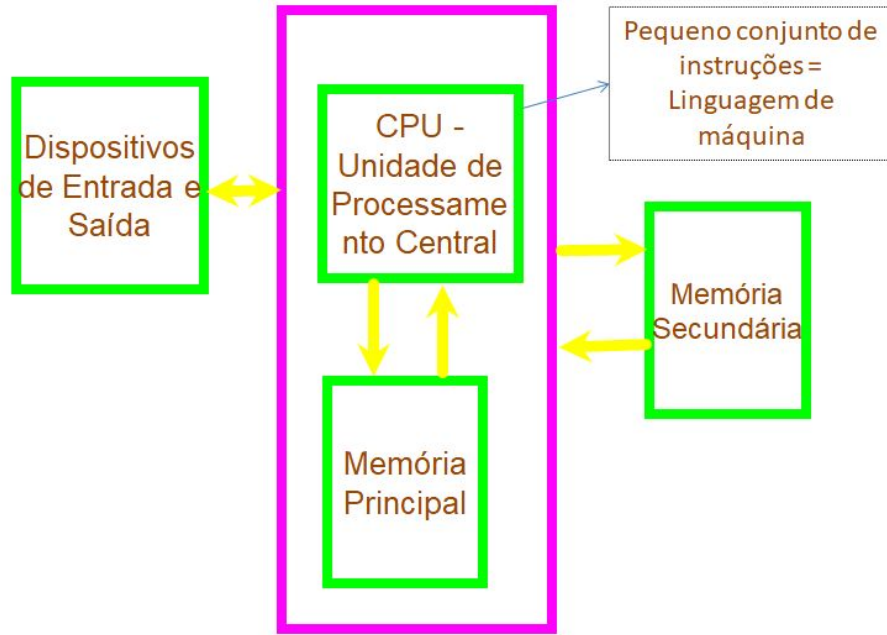
Resumindo

- Linguagens naturais são sintaticamente e semanticamente flexíveis
- Linguagens de programação são linguagens formais
 - Sintaticamente rígidas, ou seja, possuem símbolos e estruturas bem definidas
 - Semanticamente rígidas, ou seja, permitem apenas uma interpretação (não admitem ambiguidades)



**Mas como um programa é
compreendido e executado por um
computador?**

Como funciona um computador?



Arquitetura simplificada [SEVERANCE, 2016]

- CPU: processamento das instruções de um programa
- Dispositivos de entrada: Teclado, mouse, tela sensível ao toque.
- Dispositivos de saída: Monitor, alto-falante, impressora, gravador de DVD
- Memória principal: Rápido, pequena e volátil
- Memória secundária: Lenta, grande e permanente - dura até ser removido – Ex: HD, pen drive..



Como os computadores compreendem uma linguagem de programação?

- **Linguagem de máquina/linguagem de baixo nível**
 - Linguagem que o computador compreende
 - Pequeno conjunto de instruções da máquina
 - Cada tipo de processador contém um conjunto diferente de instruções
 - Baixa portabilidade



Como os computadores compreendem uma linguagem de programação?

- Linguagem de programação de alto nível
 - Maior portabilidade, independente de processador
 - Menos complexa de programar, mais produtiva
 - Traduzida para a linguagem de máquina por meio de um compilador ou interpretador



Linguagem de
alto nível

Lê, traduz e
executa cada
instrução



Linguagem de
alto nível

Traduz todo o
programa

Linguagem de
baixo nível
executável

Representação de dados



Dados

- Os programas manipulam **dados**
- Dados, assim como as instruções, são armazenados na **memória**
- Tipos de dados
 - **Básicos:** inteiros, reais, caracteres, lógicos
 - **Estruturados/compostos:** vetores e registros



Representação de dados

- Representação básica: bits
- 1 byte = agrupamento de 8 bits
- Agrupamentos de bytes:
 - 1 quilobyte = 1024 bytes = 2^{10} bytes
 - 1 megabyte = 1024 kilobytes = $2^{10}2^{10}$
 - 1 gigabyte = 1024 megabytes = $2^{10}2^{10}2^{10}$
 - 1 terabyte = 1024 gigabytes ...

Múltiplos do byte						x · d · e
Prefixo binário (IEC)			Prefixo do SI			
Nome	Símbolo	Múltiplo	Nome	Símbolo	Múltiplo	
byte	B	2^0	byte	B	10^0	
kibibyte	KiB	2^{10}	kilobyte	kB	10^3	
mebibyte	MiB	2^{20}	megabyte	MB	10^6	
gibibyte	GiB	2^{30}	gigabyte	GB	10^9	
tebibyte	TiB	2^{40}	terabyte	TB	10^{12}	
pebibyte	PiB	2^{50}	petabyte	PB	10^{15}	
exbibyte	EiB	2^{60}	exabyte	EB	10^{18}	
zebibyte	ZiB	2^{70}	zettabyte	ZB	10^{21}	
yobibyte	YiB	2^{80}	yottabyte	YB	10^{24}	
unbibyte	UiB	2^{110}	undecabyte	UB	10^{33}	



Representação de dados

- Todos os dados e instruções são representados em binário
- Exemplo: Conversão de caracteres para binário
 - Tabela ASCII:
<https://www.ime.usp.br/~pf/algoritmos/apend/ascii.html>
 - Ex: “CAP!” em ASCII : 01000011 01000001 01010000 00100001



Representação de dados

- Exemplo: Valor real -3,53125, formato IEEE-754:

11000000 01100010 00000000 00000000

- Instrução INC AL (processador 8086):

11111110 10100000



Vídeo YouTube

Dentro do seu computador - Bettina Bair

<https://www.youtube.com/watch?v=AkFi90lZmXA>



Quiz online!
Acesse: kahoot.it

- 
- Leitura recomendada:

<https://panda.ime.usp.br/pensepy/static/pensepy/01-Introducao/introducao.html>

- Vídeo: Discovery Channel: Entenda o seu Mundo {Volume 6} - Entendendo o Computador

<https://youtu.be/wTBOhNe8kIM>



Referências

FORBELLONE, André; EBERSPÄCHER, Henri. Lógica de Programação - A construção de algoritmos e estruturas de dados. 3ª Edição. Editora Pearson Prentice Hall, 2005 (disponível na biblioteca).

MEDINA, Marco; FERTIG, Cristina. Algoritmos e Programação - Teoria e Prática. 3ª Edição. Editora Novatec, 2005. (disponível na biblioteca).

ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. Fundamentos da Programação de Computadores. 2ª edição. Editora Pearson Prentice Hall, 2007 (disponível na biblioteca).

MILLER, B.; RANUM, D.; ELKNER, J.; WENTWORTH, P.; ELKNER, J.; DOWNEY, A.; MEYERS, C. Como pensar como um Cientista da Computação: Disponível em:
<https://panda.ime.usp.br/pensepy/static/pensepy/>