CES-11

Algoritmos e Estruturas de Dados

Carlos Alberto Alonso Sanches
Juliana de Melo Bezerra

Objetivos gerais

- Compreensão da necessidade de uma boa estruturação das informações processadas no computador
- Capacidade de escolher a estrutura de dados mais adequada para uma determinada aplicação
- Capacidade de programar métodos que criam e manipulam essas estruturas
- Famoso livro de Niklaus Wirth:
  Algorithms + Data Structures = Programs
- "Estar alfabetizado" *versus* "saber redigir"

Algoritmo

- Informalmente, um algoritmo é qualquer procedimento computacional bem definido que recebe um conjunto de valores como *entrada* e produz outro conjunto de valores como *saída*
- Algoritmo correto
  - Para cada instância de entrada, termina e produz uma saída adequada
- Algoritmo incorreto
  - Quando para com uma resposta indesejada
  - Quando não para (ou seja, permanece em *loop*)

Exemplos de algoritmos

- Projeto Genoma Humano
  - O DNA humano possui cerca de 100 mil genes e 3 bilhões de pares de bases químicas
    - Necessidade de armazenar esses dados e analisá-los
    - Inserção, eliminação e atualização dos dados
    - Ordenação, classificação e pesquisa de informações
    - Correlação entre os dados
    - Cálculos científicos envolvendo grandes matrizes multidimensionais
- Internet
  - Emprega algoritmos para gerenciar e manipular um enorme volume de dados
    - Localização de rotas por onde os dados trafegam
    - Mecanismos de pesquisa de páginas

Exemplos de algoritmos

- Comércio eletrônico
  - Necessidade de manter informações privativas (número de cartões, senhas, etc.)
    - Criptografia de chave pública e assinaturas digitais
    - Compactação e descompactação
- Indústria e comércio
  - Alocação adequada de recursos escassos
    - Onde localizar os poços de petróleo para maximizar o lucro?
    - Onde investir em publicidade? Público? Região?
    - Como designar tripulações de voo de um modo menos dispendioso?
    - Qual melhor investimento: ações, opções, títulos públicos?

Desenvolvimento de um algoritmo

- Características dos algoritmos
  - Considere um problema prático
  - Existem muitas resoluções candidatas, mas provavelmente não serão exatamente aquilo de que você precisa...

Convém conhecer os algoritmos existentes para então adaptá-los às suas necessidades

- Qual o melhor algoritmo?
  - Basta que forneça a resposta correta?
  - Também é importante a sua *eficiência*
    - Tempo de computação
    - Memória utilizada

Estruturas de dados

- O consumo de tempo e de memória torna-se extremamente crítico quando o universo de informações é muito grande...
- A eficiente utilização dos recursos computacionais e a redução do tempo de resposta dependem de dois fatores:
  - Boa estruturação das informações
  - Bons algoritmos que manipulem essas estruturas
- Principais modelos para visualizar, interpretar e armazenar dados:
  - Listas lineares
  - Árvores
  - Grafos

Listas lineares

- Seus elementos formam uma sequência linear:

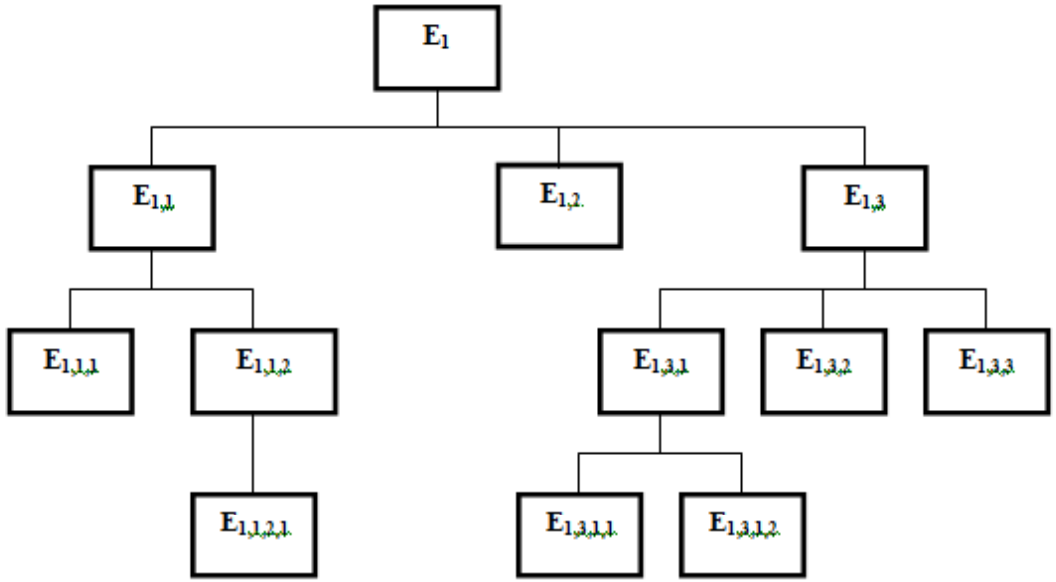
- Cada elemento possui um antecessor e um sucessor (exceto o primeiro e o último)
- Tabelas, em geral, se enquadram neste modelo:

Nome	Endereço	Outros

- Listas telefônicas
- Folhas de pagamento
- Livros de uma biblioteca
- Tabelas de um banco de dados relacional

Árvores

- Há uma hierarquia entre os elementos:

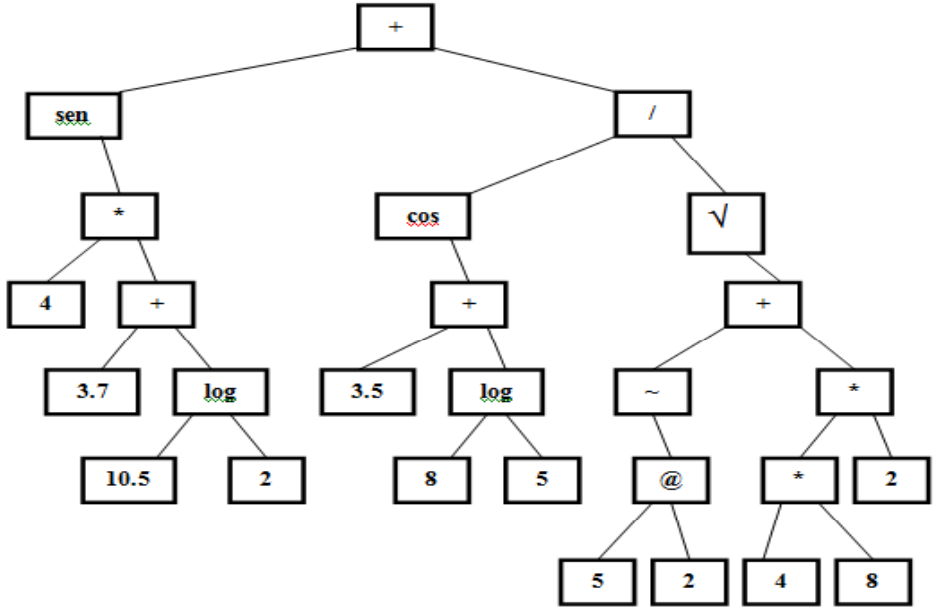


- Cada elemento:
  - Tem um único pai (exceto a raiz)
  - Pode ter vários filhos
  - Não pode ser pai de nenhum ancestral

Exemplos de árvores

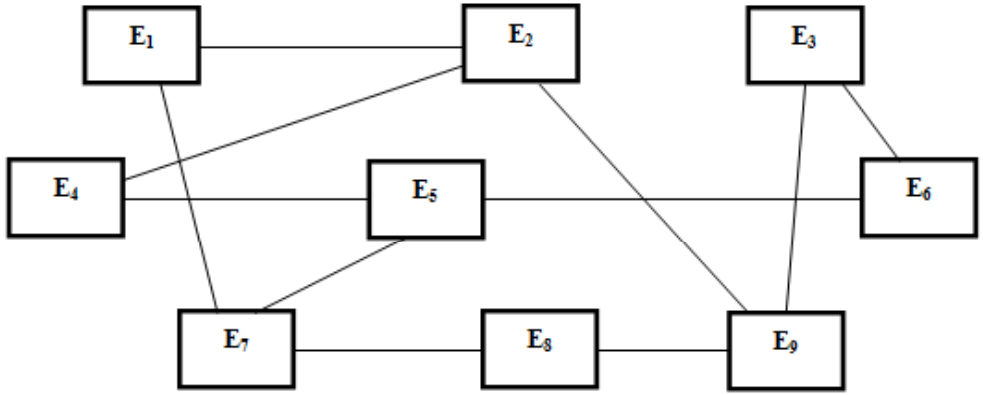
- Organograma de empresas
- Organização de livros e cursos
- Jogos eliminatórios de um campeonato
- Expressões aritméticas

$$\text{sen}(4*(3.7+\log_2 10.5))+\frac{\cos(3.5+\log_5 8)}{\sqrt{-(5)^2+4*8*2}}$$



Grafos

- Há uma interligação geral entre os elementos, sem formar sequências ou hierarquias:



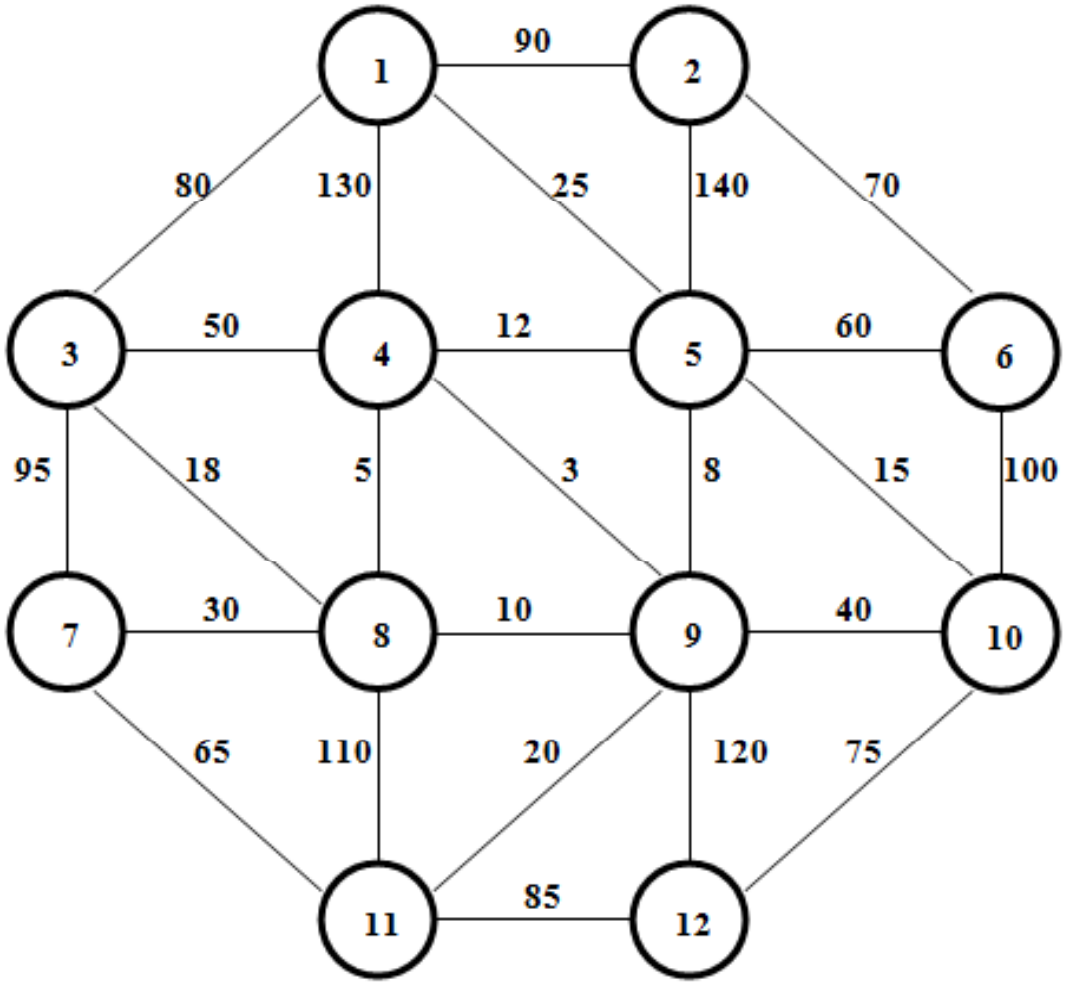
- Exemplos:
  - Tarefas de um projeto
  - Sistema rodoviário
  - Redes de interconexão
  - Fornecimento de produtos entre fábricas
  - Máquinas de estados finitos
  - etc.

Importância da disciplina

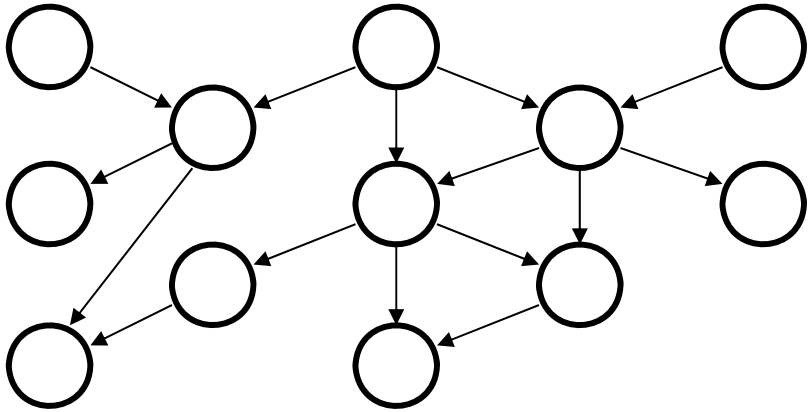
- Na Ciência da Computação:
  - Processo de compilação
  - Gerenciamento de programas e de memória
  - Construção de bancos de dados
  - etc.
- Mesmo sem ser um especialista na área, um engenheiro nos dias de hoje deve ser capaz de elaborar algoritmos não triviais.
- Alguns exemplos de problemas em grafos:
  - Caminhos mais curtos
  - Validação de grafos acíclicos
  - Componentes fortemente conexos
  - Pontos de articulação
  - Árvore geradora de custo mínimo

Caminhos mais curtos

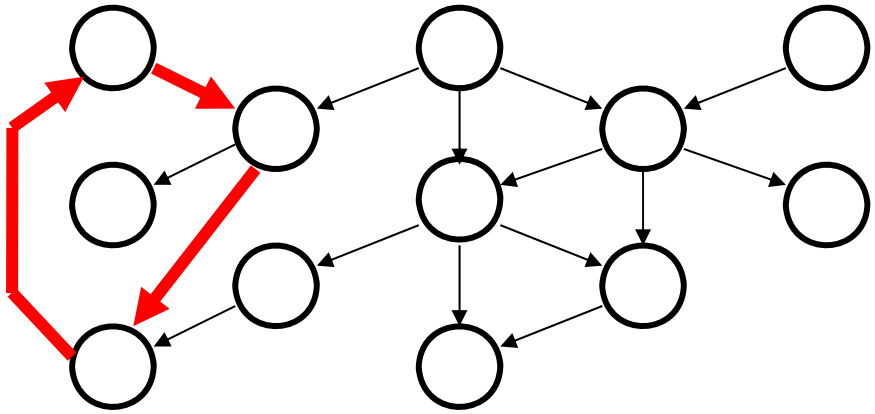
- Distâncias a partir de um dado vértice
- Aplicações:
  - Transporte terrestre ou aéreo
  - Robótica
  - Projetos de VLSI



Validação de grafos acíclicos



Grafo acíclico

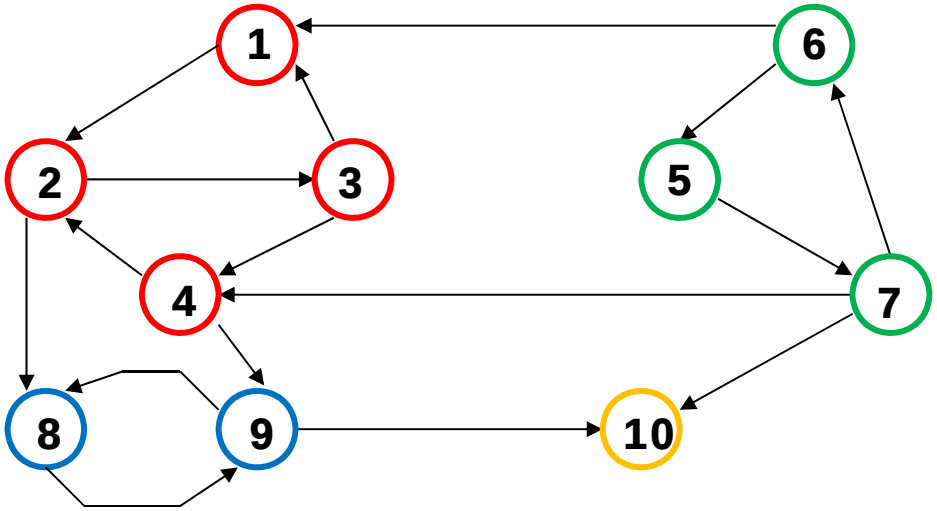


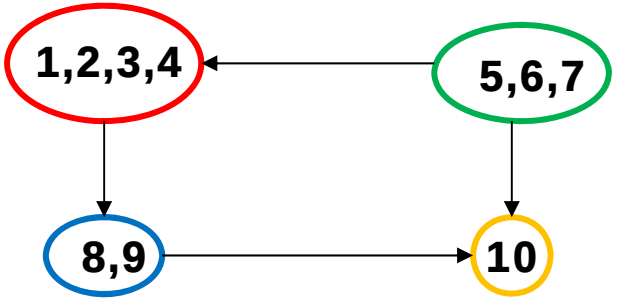
Grafo cíclico

- Aplicações:
  - Gerenciamento de projetos (redes PERT-CPM: *Program Evaluation and Review Technique, Critical Path Method*)
  - Simulação de circuitos combinacionais

Componentes fortemente conexos

- Subconjuntos maximais de vértices onde há caminhos de qualquer vértice a todos os demais.

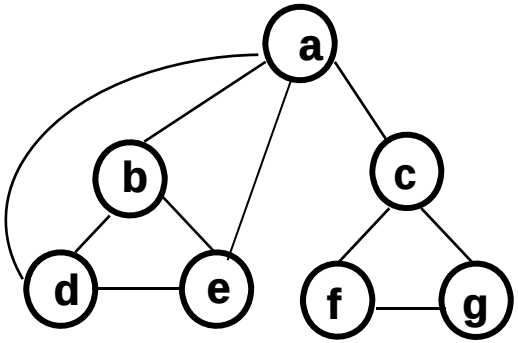


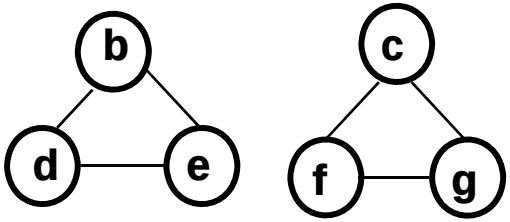


- Aplicações:
  - Privacidade em sistemas de comunicação
  - Classes de equivalência em circuitos digitais
  - Redução do tamanho de determinados problemas

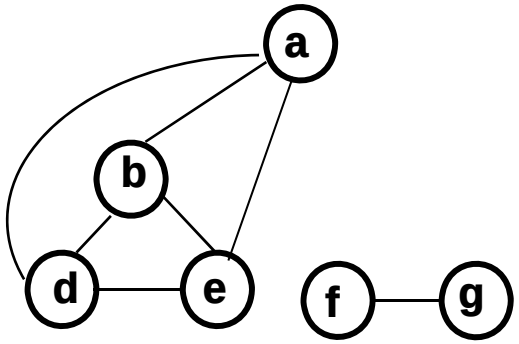
Pontos de articulação

- Um vértice é ponto de articulação se, ao ser removido, desconecta o grafo.





a é ponto de articulação

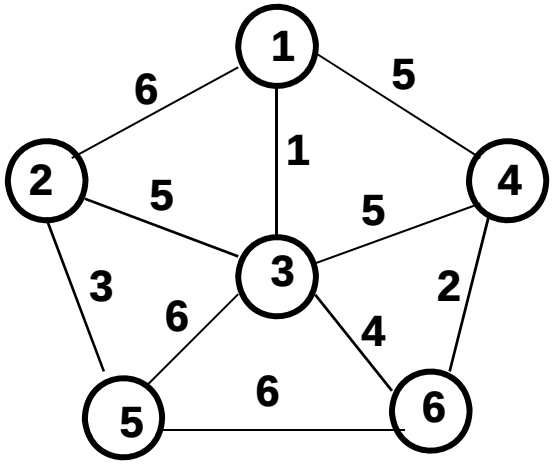


c é ponto de articulação

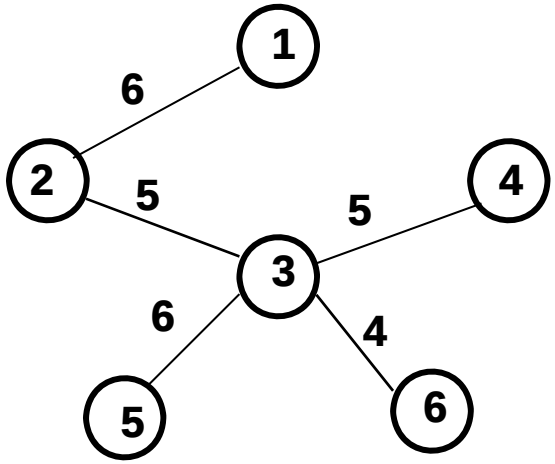
- Aplicações:
  - Sistemas de transmissão de energia elétrica
  - Sistemas de distribuição hidráulica
  - Redes de interconexão em geral

Árvore geradora de custo mínimo

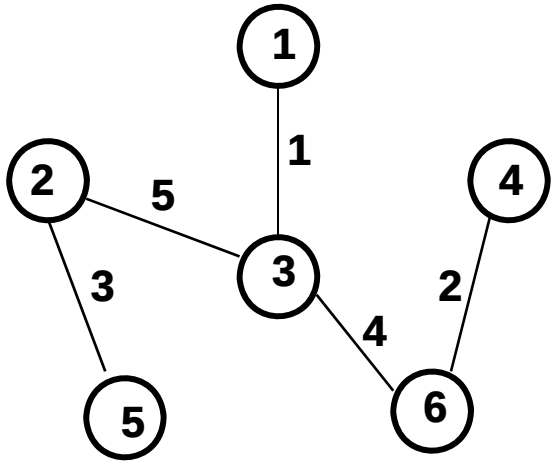
- Às vezes, é necessário encontrar um subgrafo gerador (todos os vértices e um subconjunto das arestas) que seja árvore e que tenha custo mínimo.



Grafo não orientado



Árvore de custo 26



Árvore de custo 15 (mínimo)

- Aplicações:
  - Redes de interconexão em geral

Plano do curso

- Primeiro Bimestre:
  - Breve revisão
  - Noções de complexidade de algoritmos
  - Listas lineares
  - Pilhas, filas e *deques*
  - Árvores
  - Árvores binárias
- Segundo Bimestre:
  - Algoritmos de ordenação
  - Paradigma da Divisão-e-Conquista
  - Grafos
    - Representações
    - Soluções de alguns problemas clássicos
  - Noções de Programação Orientada a Objetos

Avaliações individuais

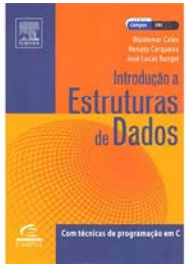
- Em cada bimestre:
  - 2 provas
  - 3 laboratórios
- Pesos:
  - Provas: 50%
  - Média dos laboratórios: 50%

Premissas éticas nos laboratórios

- É permitido:
  - Consultar material didático (*slides*, apostilas, códigos) de outros professores do ITA ou disponível na internet (neste último caso, se for código, sem fornecê-lo a outros colegas)
  - Pensar na solução junto com um colega, antes de programarem
  - Trocar ideias com outro colega, mas sem olhar o código que ele escreveu
  - Ajudar um colega a encontrar erros de codificação, desde que já tenha terminado o próprio laboratório
- Não é permitido:
  - Utilizar código pronto encontrado na internet
  - Olhar ou copiar soluções de outro aluno (da mesma turma ou de anteriores)
  - Fazer o exercício (mesmo parcialmente) de um colega com dificuldades
  - Escrever o código junto com outro colega

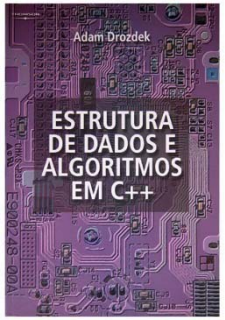
Bibliografia

- W. Celes, R. Cerqueira, J.L. Rangel
  Introdução a Estrutura de Dados



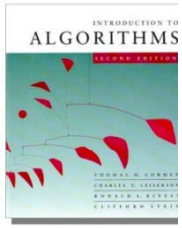
- P. Feofiloff
  Algoritmos em Linguagem C

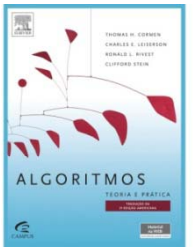
- A. Drozdek
  Estrutura de Dados e Algoritmos em C++



Bibliografia complementar

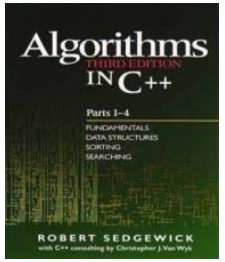
- T.H. Cormen, C.E. Leiserson, R.L. Rivest
  Introduction to Algorithms

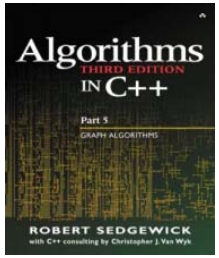


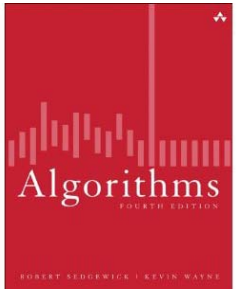


- A. V. Aho, J. E. Hopcroft, J. D. Ullman
  Data Structures and Algorithms

- R. Sedgewick
  Algorithms [in C, C++, Java]



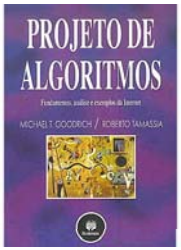




Bibliografia complementar

- A.M. Tanenbaum, Y. Langsam, M.J. Augenstein
  Estruturas de Dados usando C

- M.T. Goodrich, R. Tamassia
  Projeto de Algoritmos



- B.R. Preiss
  Estruturas de Dados e Algoritmos