# Algoritmos e Estruturas de Dados II

2º Período Engenharia da Computação

Prof. Edwaldo Soares Rodrigues
Email: edwaldo.rodrigues@uemg.br

UNIVERSIDADE
DO ESTADO DE MINAS GERAIS | UEMG
UNIDADE DIVINÓPOLIS

# Pilha

# Pilha

- O que define uma pilha?
  - Para que um elemento entre na pilha ele será sempre colocado no topo da mesma;
  - Para que um elemento seja retirado da pilha, ele sempre sairá do topo da mesma;

# Pilha

- Estratégia LIFO:
  - Last in, first out;
  - Os elementos da pilha são retirados na ordem inversa à ordem em que foram introduzidos: o último a entrar é o primeiro a sair;
  - Não são permitidas operações sobre quaisquer nodos, somente sobre aqueles definidos pela organização da pilha;

# Pilha

- Aplicações:
  - Recurso de “voltar” entre os endereços mais recentemente visitados de um *browser*;

# Pilha

- Aplicações:
  - Controle de chamadas de subrotinas (procedimentos e funções);
    - Sempre que o programa encontrar uma chamada a uma subrotina ele irá empilhar o contexto atual da aplicação (valores de variáveis, endereços de retorno) em uma pilha;
    - Na medida em que as chamadas vão sendo finalizadas a pilha vai sendo desfeita;

# Pilha

- Implementação:
  - Uma pilha permite basicamente duas operações:
    - Empilhar (push);
    - Desempilhar (pop);
  - Antes de pensar os algoritmos, precisamos:
    - Pensar em uma estratégia de armazenamento da estrutura;
    - Pensar na interface das operações que irão manipular os dados da estrutura;

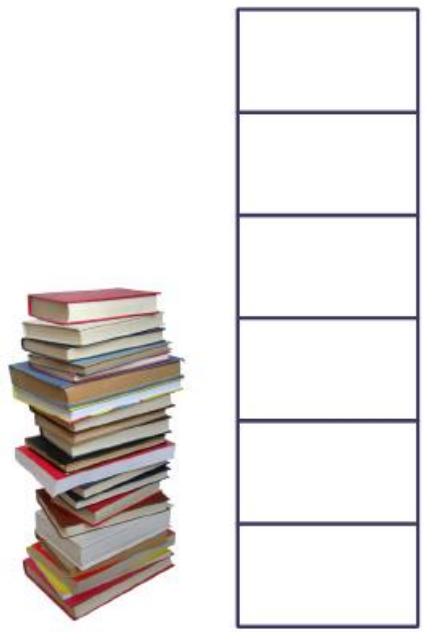
# Pilha

- Organização física:
  - Para representar fisicamente uma pilha podemos usar diferentes estratégias:
    - Contiguidade física – com o uso de vetores (pilha estática);
    - Alocação dinâmica ou encadeamento – com a utilização de ponteiros (pilha dinâmica);
  - A organização lógica da pilha independe da estratégia de armazenamento;

# Pilha

- Funcionamento:
  - Empilha 10
  - Empilha 20
  - Empilha 30
  - Desempilha
  - Empilha 50
  - Empilha 40
  - Empilha 12
  - Empilha 7
  - Empilha 9

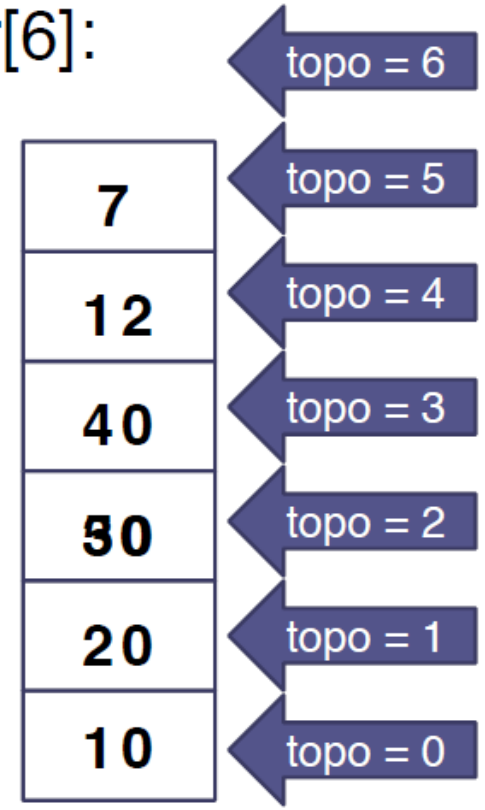
- Vetor[6]:



# Pilha

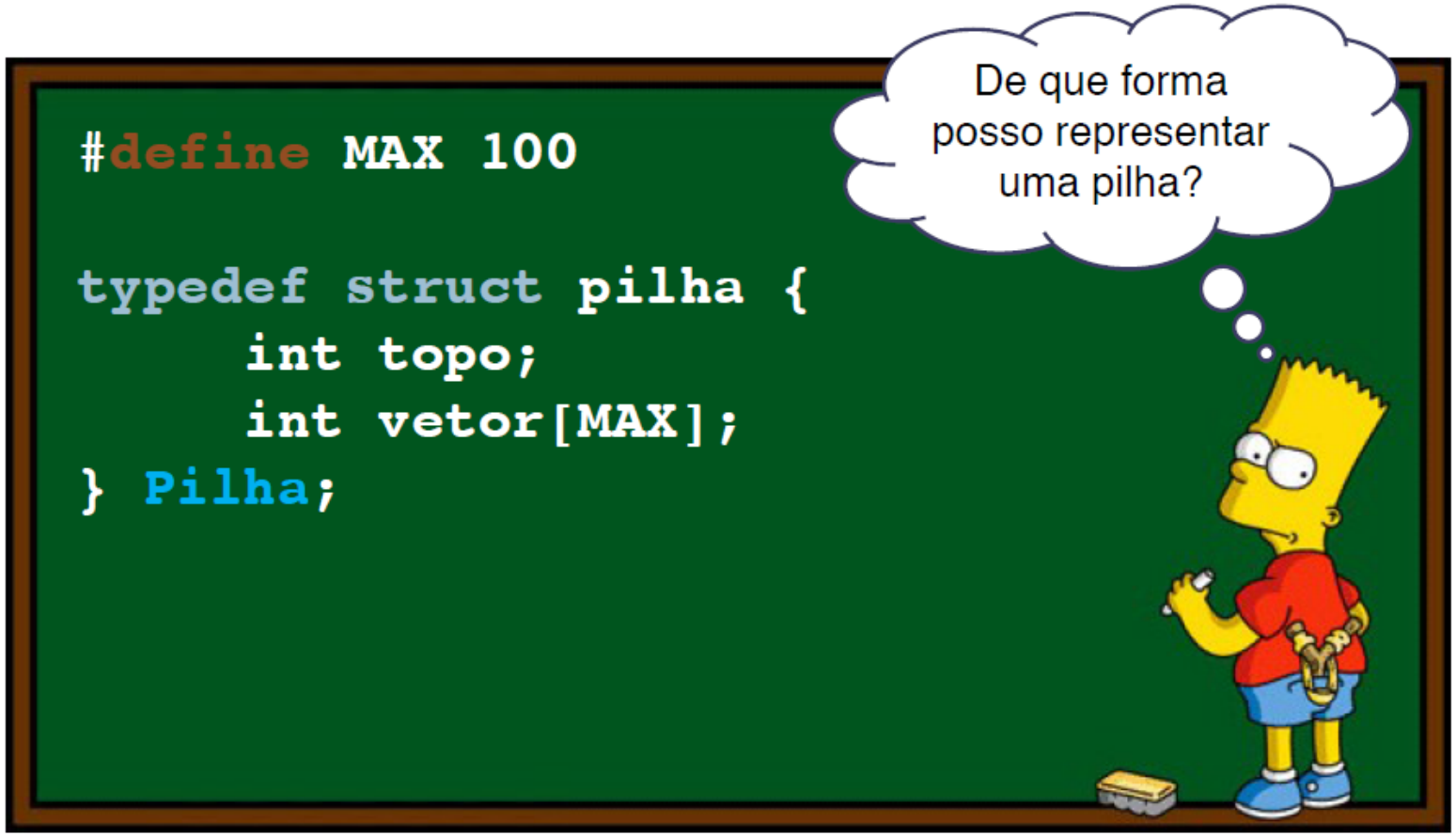
- Funcionamento
  - Empilha 10
  - Empilha 20
  - Empilha 30
  - Desempilha
  - Empilha 50
  - Empilha 40
  - Empilha 12
  - Empilha 7
  - Empilha 9
    - **Erro**! Pilha cheia

- Vetor[6]:



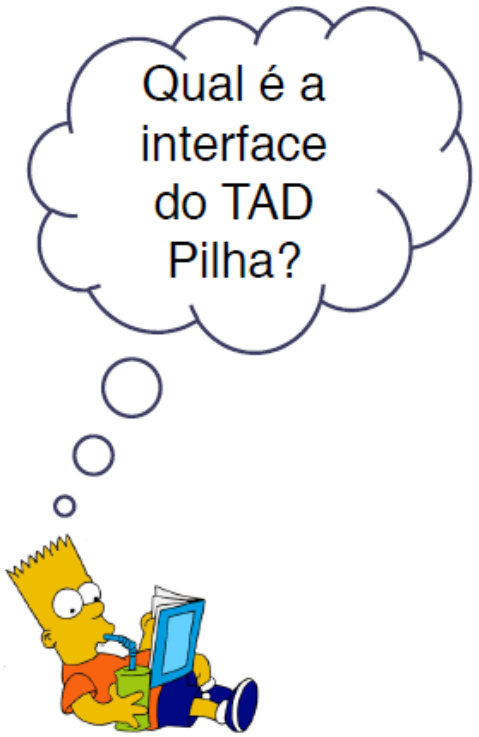
# Pilha

- Definição conceitual: TAD pilha;



# Pilha

- Quais operações podemos fazer com uma pilha?




- Inicialmente precisamos:
  - Criar uma pilha;
  - Empilhar um elemento;
  - Desempilhar um elemento;
  - Destruir uma pilha (liberar a memória ocupada pela mesma);
  - Saber se a pilha está vazia;
  - Saber se a pilha está cheia;
  - etc...

# Pilha

- Definição das operações da TAD pilha:

```
Pilha* criaPilha();

void liberaPilha(Pilha* p);

int empilha(Pilha* p, int v);

int desempilha(Pilha* p, int* v);

int estahVazia(Pilha* p);

int estahCheia(Pilha* p);
```

# Pilha

- Implementação e utilização das operações:

```
a = criaPilha();

empilha(a,10);
empilha(a,20);
empilha(a,30);

int x;
desempilha(a, &x);
printf("Elemento '%d' retirado",x);

liberaPilha(a);
```

# Fila

# Fila

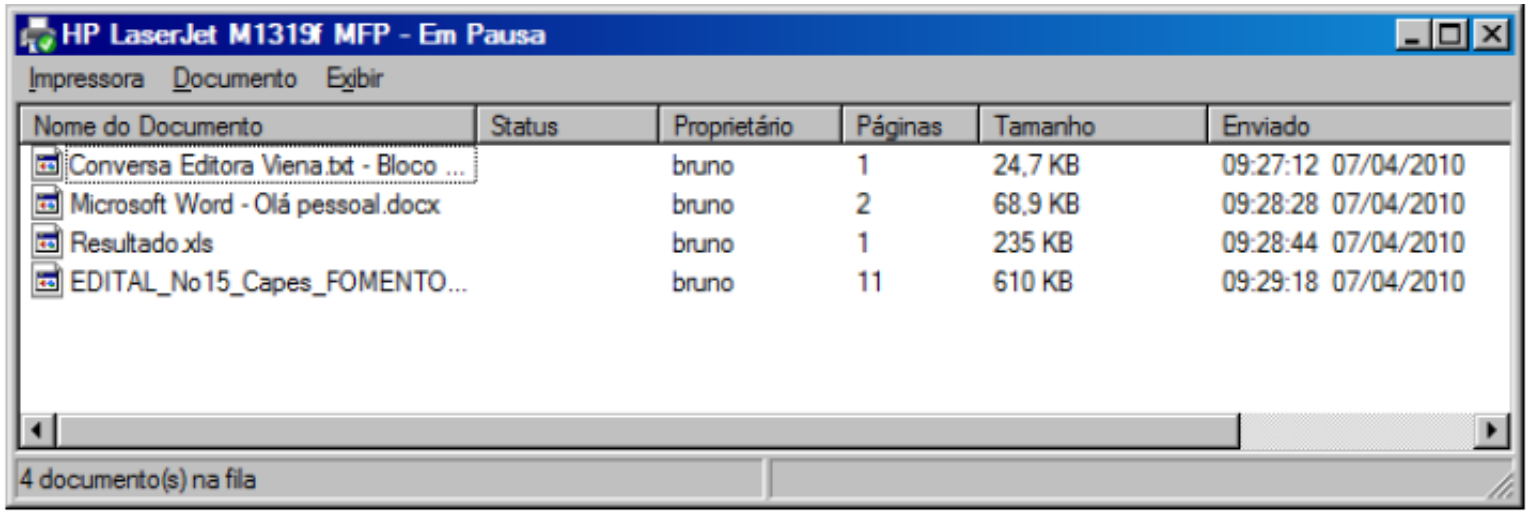
- O que define uma Fila:
  - Para que um elemento entre na fila ele será sempre colocado em uma extremidade denominada final (término, fim);
  - Para que um elemento seja retirado da fila, ele sairá de uma outra extremidade denominada começo (início, frente);

# Fila

- Estratégia FIFO:
  - First in, first out;
  - Os elementos da fila são retirados na mesma ordem em que foram introduzidos: o primeiro a entrar é o primeiro a sair;
  - Não são permitidas operações sobre quaisquer nodos, somente sobre aqueles definidos pela organização da fila;

# Fila

- Aplicações:
  - Controle de acesso a recursos compartilhados (fila de impressão, por exemplo);



# Fila

- Aplicações:
  - Definição das prioridades em uma lista de downloads;
  - Agendamento de tarefas;
  - Troca de mensagens em um sistema distribuído;
  - Leitura do buffer do teclado;
  - Enfileiramento de pacotes no equipamento de rede, antes de submetê-los ao enlace;
  - Entre outros;

# Fila

- Implementação:
  - Uma fila permite basicamente duas operações:
    - Enfileirar (inserir um elemento na fila);
    - Desenfileirar (retirar um elemento da fila);
  - Antes de pensar os algoritmos, precisamos:
    - Pensar em uma estratégia de armazenamento da estrutura;
    - Pensar na interface das operações que irão manipular os dados da estrutura;

# Fila

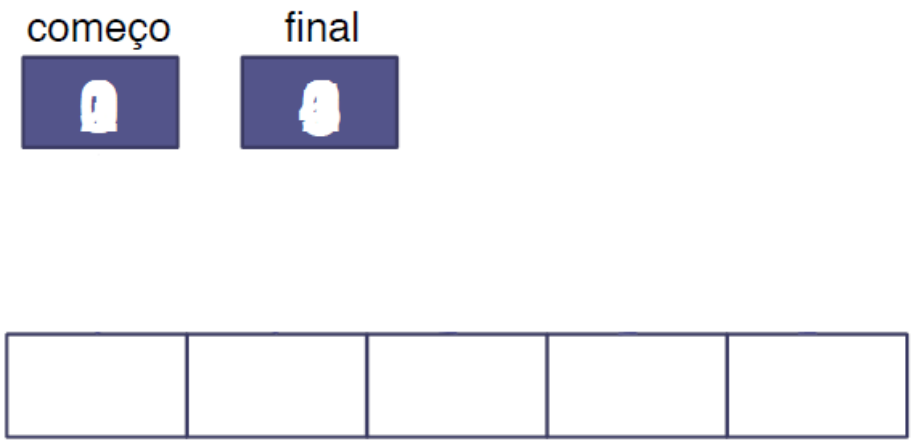
- Organização física:
  - Para representar fisicamente uma fila podemos usar diferentes estratégias:
    - Contiguidade física – com o uso de vetores (fila estática);
    - Alocação dinâmica ou encadeamento – com a utilização de ponteiros (fila dinâmica);
  - A organização lógica da fila independe da estratégia de armazenamento;

# Fila

- Funcionamento:

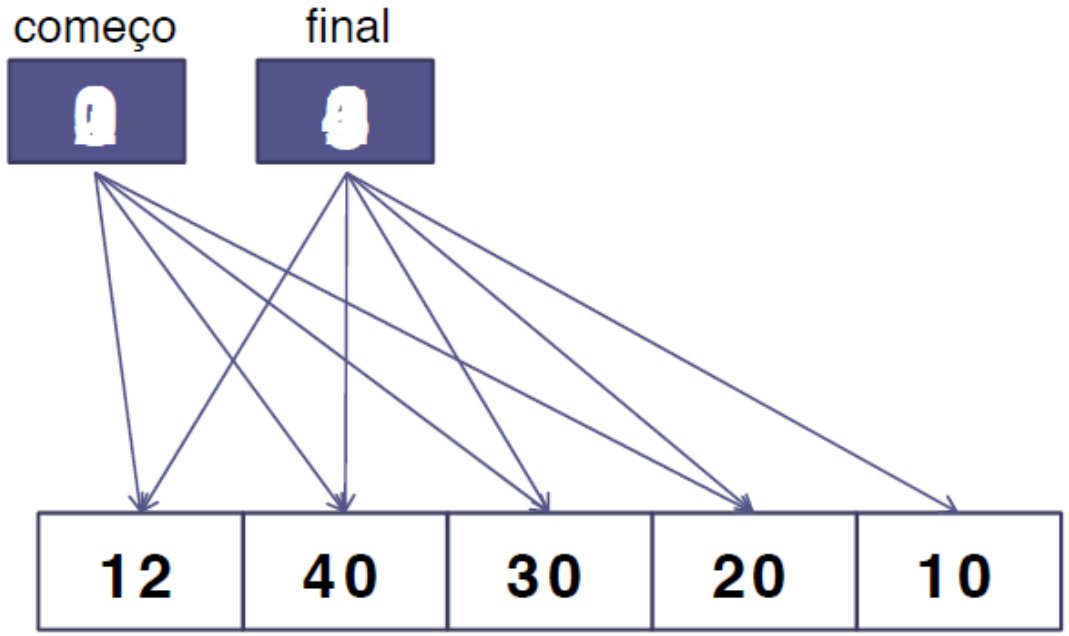
  - Insere 12
  - Insere 40
  - Insere 30
  - Retira
  - Insere 20
  - Retira
  - Retira
  - Insere 10
  - Insere 9

- Vetor[5]:



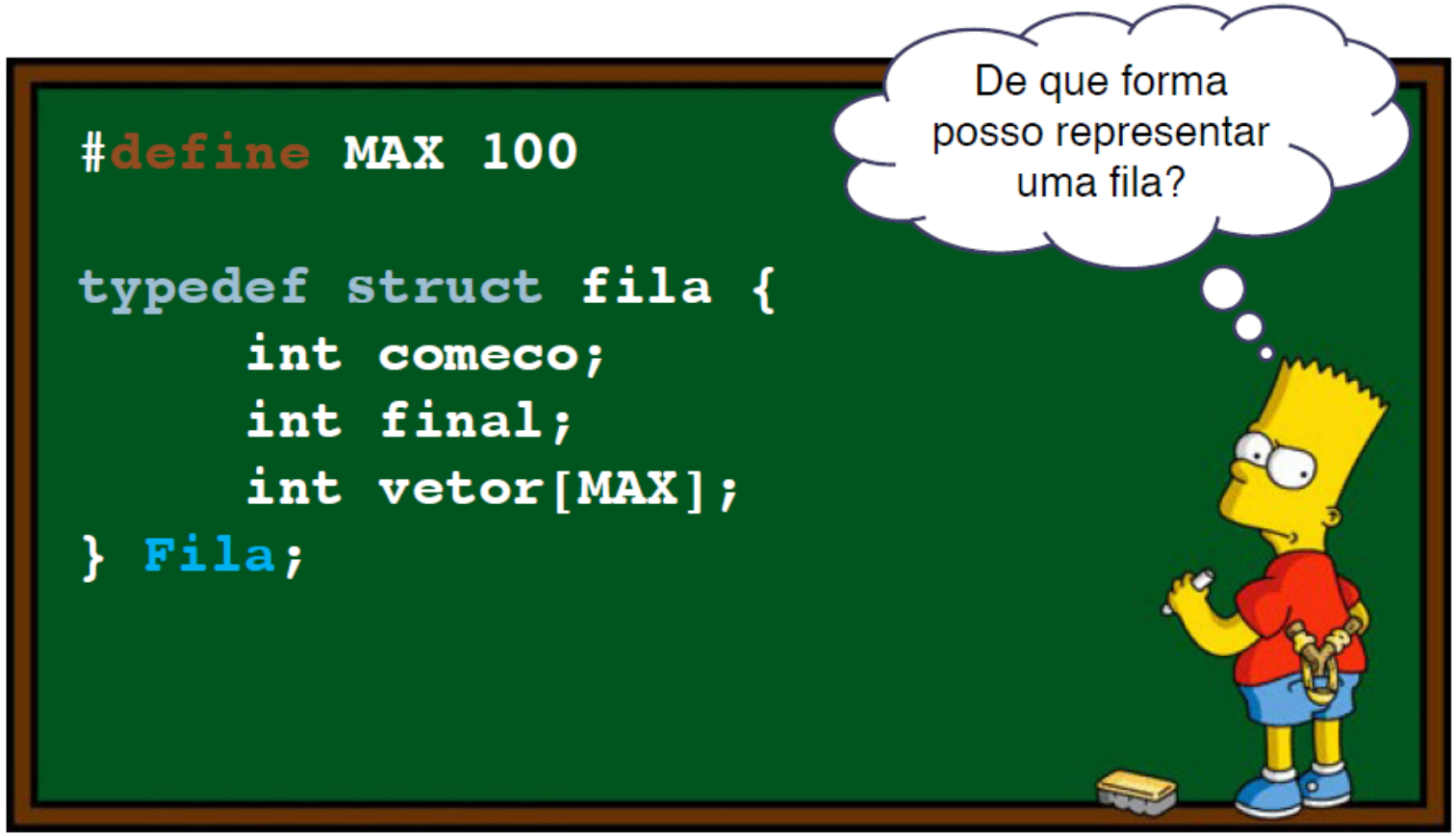
# Fila

- Funcionamento:
  - Insere 12
  - Insere 40
  - Insere 30
  - Retira
  - Insere 20
  - Retira
  - Retira
  - Insere 10
  - Insere 9
    - **Erro**! Fila cheia



# Fila

- Definição conceitual: TAD fila;



# Fila

- Quais operações podemos fazer com uma fila?


- Inicialmente precisamos:
  - Criar e inicializar uma fila;
  - Enfileirar (inserir) um elemento;
  - Desenfileirar (retirar) um elemento;
  - Destruir uma fila (liberar a memória ocupada pela mesma);
  - Saber se a fila está vazia;
  - Saber se a fila está cheia;
  - etc...

# Fila

- Definição das operações da TAD fila:

```
Fila* criaFila();

void liberaFila(Fila* p);

int inserir(Fila* p, int v);

int retirar(Fila* p, int* v);

int estahVazia(Fila* p);

int estahCheia(Fila* p);
```

# Fila

- Implementação e utilização das operações:

```
a = criaFila();

inserir(a,10);
inserir(a,20);
inserir(a,30);

int x;
retirar(a, &x);
printf("Elemento '%d' retirado",x);

liberaFila(a);
```

# Algoritmos e Estruturas de Dados II

- Bibliografia:
  - Básica:
    - CORMEN, Thomas, RIVEST, Ronald, STEIN, Clifford, LEISERSON, Charles. Algoritmos. Rio de Janeiro: Elsevier, 2002.
    - EDELWEISS, Nina, GALANTE, Renata. Estruturas de dados. Porto Alegre: Bookman. 2009. (Série livros didáticos informática UFRGS,18).
    - ZIVIANI, Nívio. Projeto de algoritmos com implementação em Pascal e C. São Paulo: Cengage Learning, 2010.
  - Complementar:
    - ASCENCIO, Ana C. G. Estrutura de dados. São Paulo: Pearson, 2011. ISBN: 9788576058816.
    - PINTO, W.S. Introdução ao desenvolvimento de algoritmos e estrutura de dados. São Paulo: Érica, 1990.
    - PREISS, Bruno. Estruturas de dados e algoritmos. Rio de Janeiro: Campus, 2000.
    - TENEMBAUM. Aaron M. Estruturas de dados usando C. São Paulo: Makron Books. 1995. 884 p. ISBN: 8534603480.
    - VELOSO, Paulo A. S. Complexidade de algoritmos: análise, projeto e métodos. Porto Alegre, RS: Sagra Luzzatto, 2001

# Algoritmos e Estruturas de Dados II