Capítulo 2

Estrutura de dados

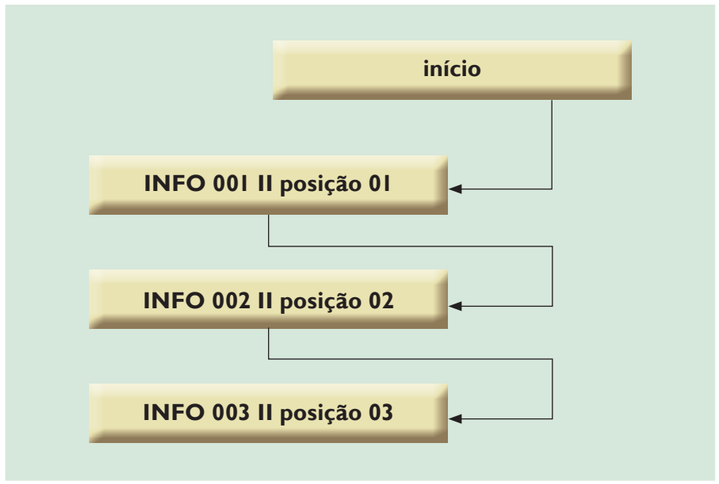
- Lista encadeada
- Lista circular
- Lista duplamente encadeada
- Pilhas
- Filas
- Árvores

A estrutura de dados define de que maneira os tipos primitivos serão organizados (FORBELLONE, 2005): por meio de lista, pilhas, filas ou árvores. Essas opções representam os conjuntos a serem manipulados por logarítmos, de diferentes formas. Esse tipo de estrutura pode sofrer alterações (MARCONDES, 2002). Na teoria, refere-se à identificação e ao desenvolvimento de modelos para a resolução de problemas. Na prática, significa criar representações concretas que podem atuar sobre modelos.

2.1. Lista encadeada

Um espaço de memória é reservado para os dados, quando são inseridos em uma lista encadeada. Então, o espaço ocupado por uma lista, na memória, é expresso pela quantidade de elementos contidos ali dentro. Para que se possa percorrer todos os elementos, deve-se armazenar – junto a cada um deles – um ponteiro que indique o elemento seguinte (figura 15).

Figura 15
Lista encadeada.



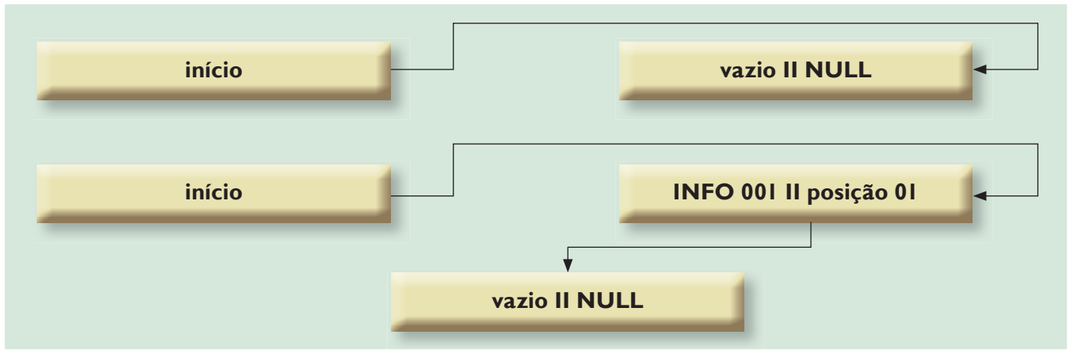


Figura 16
Inserção de elemento.

Cada elemento da lista é identificado como um nó. Utilizam-se os ponteiros para percorrer todos os nós da lista. O último elemento deverá apontar para NULL (em linguagem de programação de computador, é a palavra usada para se referir a um dispositivo nulo). Para a criação de uma lista, considera-se o primeiro ponteiro existente. Uma lista vazia deve ser criada com o ponteiro indicando NULL, permitindo inserções no final. Essa lista deve ser alocada em uma posição de memória, com encadeamento para o próximo elemento (figura 16).

Retirar um elemento da lista é um processo um pouco mais complexo, pois podemos encontrar diferentes situações, como eliminar o primeiro elemento da lista ou um elemento intermediário (figura 17).

Outras funções agregadas às listas são: busca de elementos e verificação para saber se ela está vazia ou não.

Figura 17
Eliminação de um elemento.

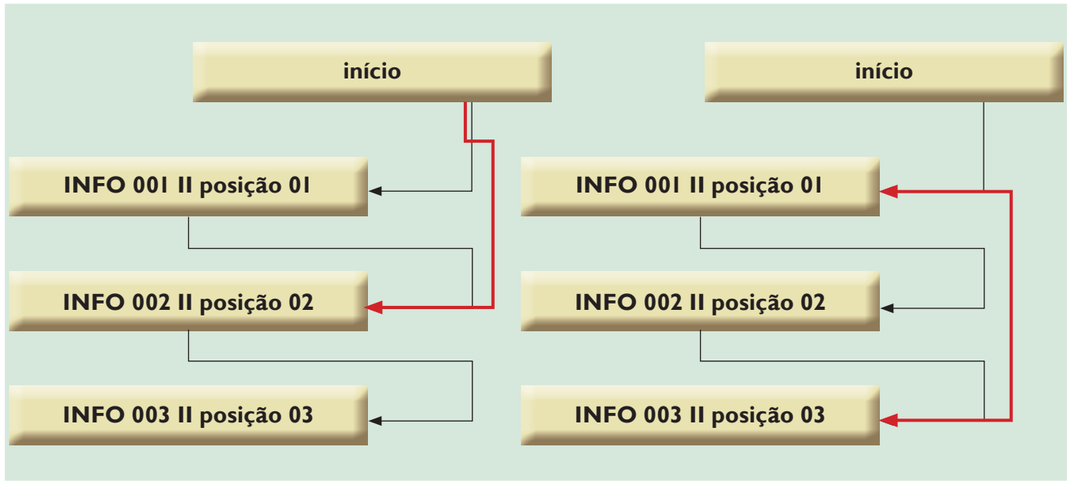
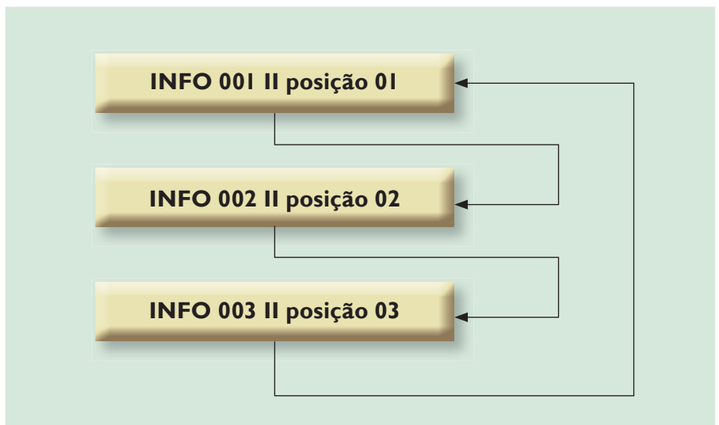


Figura 18
Lista circular.



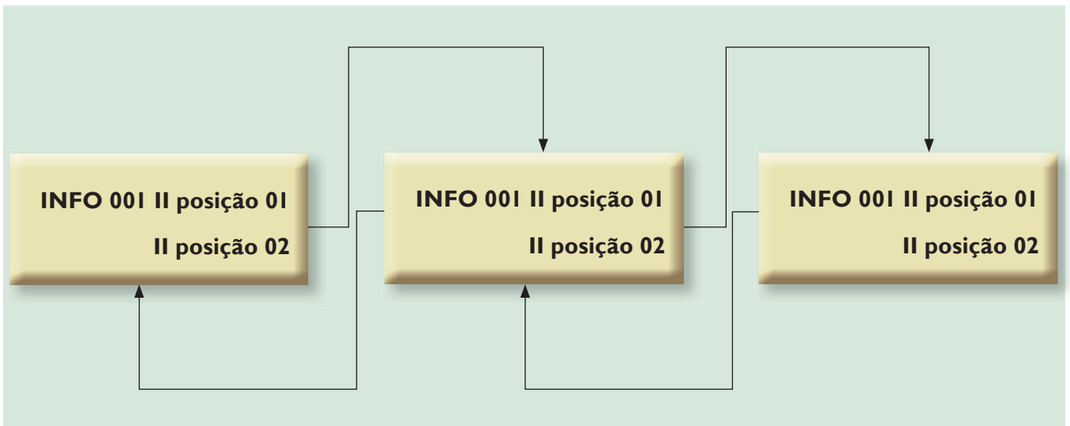
2.2. Lista circular

Em uma lista circular o último elemento deverá apontar para o primeiro (figura 18). Para percorrer a lista é preciso visitar todos os seus elementos, a partir de um ponteiro inicial, até chegar a ele novamente. Caso o ponteiro inicial seja NULL, a lista será considerada vazia.

2.3. Lista duplamente encadeada

Na lista encadeada há um ponteiro indicando o próximo elemento. Já na duplamente encadeada são dois ponteiros: um mostrando o elemento seguinte e outro, o anterior. A partir de um elemento podemos acessar ambos os lados (figura 19).

Figura 19
Lista duplamente encadeada.



Push	Push	Push	Pop	Push	Pop
		25		40	
	15	15	15	15	15
10	10	10	10	10	10

Tabela 8
Push e pop em uma pilha.

2.4. Pilhas

A pilha é uma estrutura bem simples, muitas vezes encontrada dentro do hardware. A ideia é acessar o primeiro elemento a partir do topo da pilha. Assim, um novo elemento inserido na pilha vai direto para o topo e, logicamente, é o único que pode ser retirado. Portanto, o primeiro que sai é o último que entrou. Temos, nesse caso, o LIFO (do inglês last in, first out, ou último dentro, primeiro fora). Há duas operações básicas, para trabalhar com as pilhas: push (empilhar) e pop (desempilhar) (tabela 8).

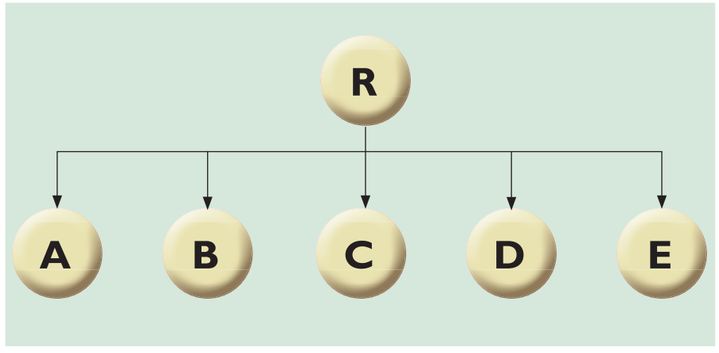
2.5. Filas

Outra estrutura é a fila, que apresenta a ordem de saída dos elementos de forma diferente do que acontece em uma pilha. Na fila, o primeiro elemento que entra é o primeiro a sair: FIFO (first in, first out, ou seja, primeiro dentro, primeiro fora). É possível inserir um novo elemento no final da fila e retirar o primeiro (tabela 9).

Tabela 9
Entrada e saída de uma fila.

Push	Pop	Push	Push	Pop	Pop
			10		
20		35	35	10	
15	20	20	20	35	10
10	15	15	15	20	35

Figura 20
Árvore e seus nós.



2.6. Árvores

Os tópicos anteriores representam a disposição de dados lineares. As árvores, por sua vez, permitem que os dados sejam dispostos de forma hierárquica, como uma estrutura de diretórios. Toda árvore possui um nó "r", denominado raiz. Os demais nós são identificados como internos ou "filhos" (figura 20).

• Árvores binárias

Um bom exemplo de árvore binária são as expressões matemáticas, as quais nos permitem distribuir os operadores e os operandos. Por exemplo, a expressão ((1+2)*(5*3))-2 (figura 21).

Figura 21
Árvore binária.

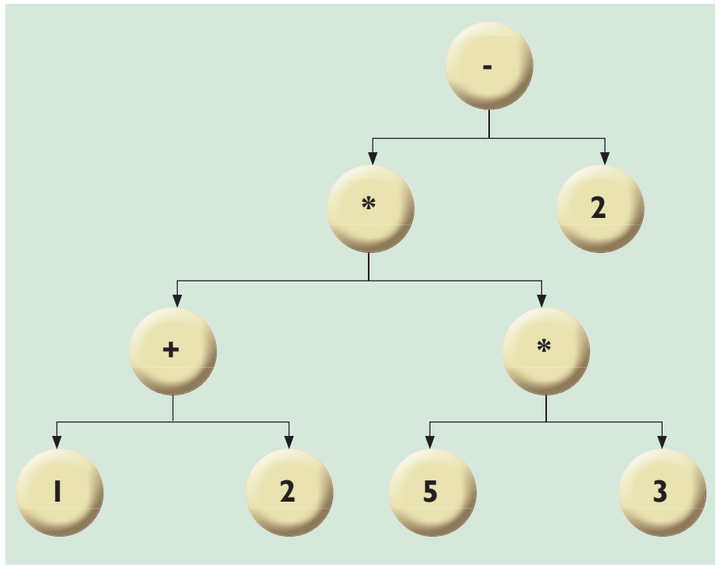
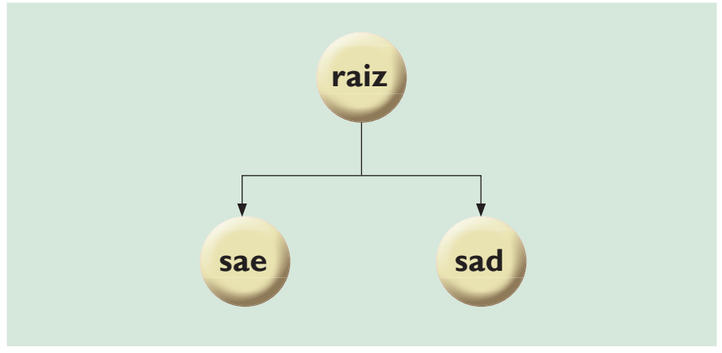


Figura 22
Esquema da Árvore Binária.



Na árvore binária cada nó tem zero, um ou, no máximo, dois "filhos". Então, é possível definir uma árvore binária como vazia ou um nó raiz com duas subárvores – uma na direita (sad) e outra na esquerda (sae) (figura 22).

• Árvore genérica

Na árvore binária há restrição quanto ao número de nós: no máximo dois. Esse limite não acontece nas árvores genéricas (figura 23).

Figura 23
Árvore genérica.