CENTRO DE CIÊNCIA E TECNOLOGIA
LABORATÓRIO DE CIÊNCIAS MATEMÁTICAS
UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE

Conceitos Básicos sobre Dados

Disciplina: Estruturas de Dados I

Prof. Fermín Alfredo Tang Montané

Curso: Ciência da Computação

Conceitos

- Dados Atómicos e Compostos;
- Tipo de dado;
- Estrutura de Dados;
- Tipos Abstratos de Dados (TADs).

Dados Atómicos e Compostos

Definição

- **Dados atómicos** são dados que consistem de uma peça única de informação. Elas não podem ser divididas em outras peças significativas de dados.
    - Por exemplo, o número inteiro 4562, pode ser considerado um valor inteiro único. Não pode ser dividido, embora seja possível extrair os dígitos.
- **Dados compostos** são dados que podem ser divididos em subcampos que possuem algum significado.
    - Por exemplo, uma data ou um número telefônico.

Tipo de Dado
Definição

- Um tipo de dado consiste de duas partes:
  - Um conjunto de dados;
  - As operações que podem ser realizadas sobre os dados.
- Por exemplo, o tipo inteiro:
  - consiste de valores inteiros definidos em um determinado intervalo;
  - além de operações (adição, substração, multiplicação, divisão) entre outras.

Tipo de dado

1. Um conjunto de valores;
2. Um conjunto de operações sobre os valores.

Tipo de Dado
Exemplos

- Mostra-se, três exemplos de tipos de dados encontrados em todos os sistemas.

Type	Values	Operations
integer	-∞, ... , -2, -1, 0, 1, 2,...,∞	*, +, -, %, /, ++, --, ...
floating point	-∞, ... , 0.0, ..., ∞	*, +, -, /, ...
character	\0, ..., 'A', 'B', ..., 'a', 'b', ..., ~	<, >, ...

Three Data Types

Estrutura de Dados
Definição

- Uma estrutura de dados é uma agregação de **dados atómicos** e/ou **dados compostos** em um conjunto com relações definidas.
- Nesta definição, **estrutura** significa um conjunto de regras que permitem que os dados permaneçam unidos ou que definem a maneira em que os dados se relacionam.
- Estruturas de dados podem ser **aninhadas**. Isto é, podemos ter estruturas da dados que consistem de outras estruturas de dados.

Estrutura de Dados

1. Uma combinação de elementos em que cada um deles é de um tipo de dado ou outra estrutura;
2. Um conjunto de associações ou relações que envolvem aos elementos.

Estrutura de Dados
Exemplos

- Temos como exemplos, as estruturas de vetor e registro.

Array	Record
Homogeneous sequence of data or data types known as elements	Heterogeneous combination of data into a single structure with an identified key
Position association among the elements	No association

Data Structure Examples

Estrutura de Dados

- A maioria das linguagens de programação suporta diversas estruturas de dados.
- Linguagens de programação modernas permitem que os programadores criem novas estruturas de dados para as suas aplicações.

Tipo Abstrato de Dados (TAD)

Definição

- Um tipo abstrato de dado TAD é uma declaração de dados empacotada junto com as operações que são significativas para os dados declarados.
- Consiste no **encapsulamento** de dados e das operações sobre esses dados assim como na ocultação de ambos aspectos do usuário.

Tipo abstrato de dado

1. Declaração de dados;
2. Declaração de operações;
3. Encapsulamento de dados e operações.

Tipo Abstrato de Dados (TAD)

Exemplo

- Um analista de sistema precisa simular a fila de espera de um banco para determinar quantos caixas são necessários para servir os clientes de maneira eficiente.
- O analista requer a simulação de uma fila (*queue*). Em geral, estrutura de filas não se encontram disponíveis nas linguagens de programação. Além disso, ele precisa definir operações básicas de filas, tais como: inserção na fila (*enqueuing*) e remoção da fila (de*queuing*)
- Existem duas soluções potenciais:

1. Criar um programa que simula a fila que o analista precisa. Esta aplicação será útil somente para o problema corrente.
2. Criar uma, ADT fila, que poderá ser utilizada para resolver qualquer problema envolvendo filas. Neste caso, o analista ainda precisará escrever o programa para simular a aplicação bancária. Isto será mais simples e rápido porque podemos concentrar na aplicação e não na fila.

Tipo Abstrato de Dados (TAD)

Conceito de Abstração

- Um usuário de TAD não está preocupado em **como** uma tarefa é realizada, mas preocupasse com, **o que** poder ser feito.
- Os TADs consistem de um conjunto de definições que permitem aos programadores usar funções cuja implementação encontra-se oculta.
- Esta generalização de operações com implementações não especificadas é conhecido como **abstração**.
- Abstrai-se a essência do processo e deixamos os detalhes de implementação ocultos.

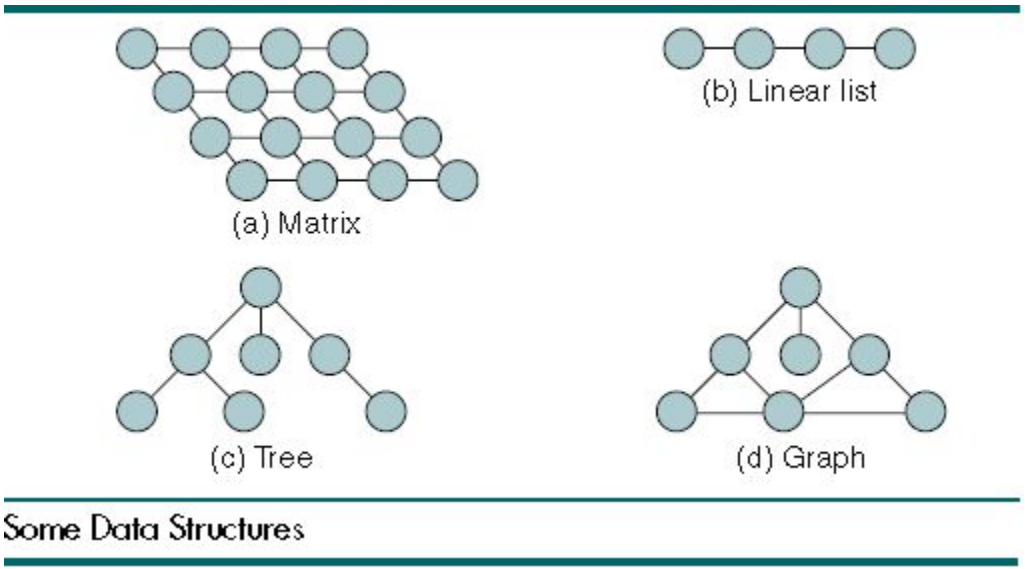
Conceito de abstração

1. Sabemos o que um tipo de dado pode fazer;
2. Como é feito não é especificado ou fica oculto.

Tipo Abstrato de Dados (TAD)

Conceito de Abstração - Exemplo

- Considere o conceito de lista. Pelo menos quatro estruturas de dados podem suportar uma lista: Uma matriz, uma lista linear, uma árvore ou um grafo.

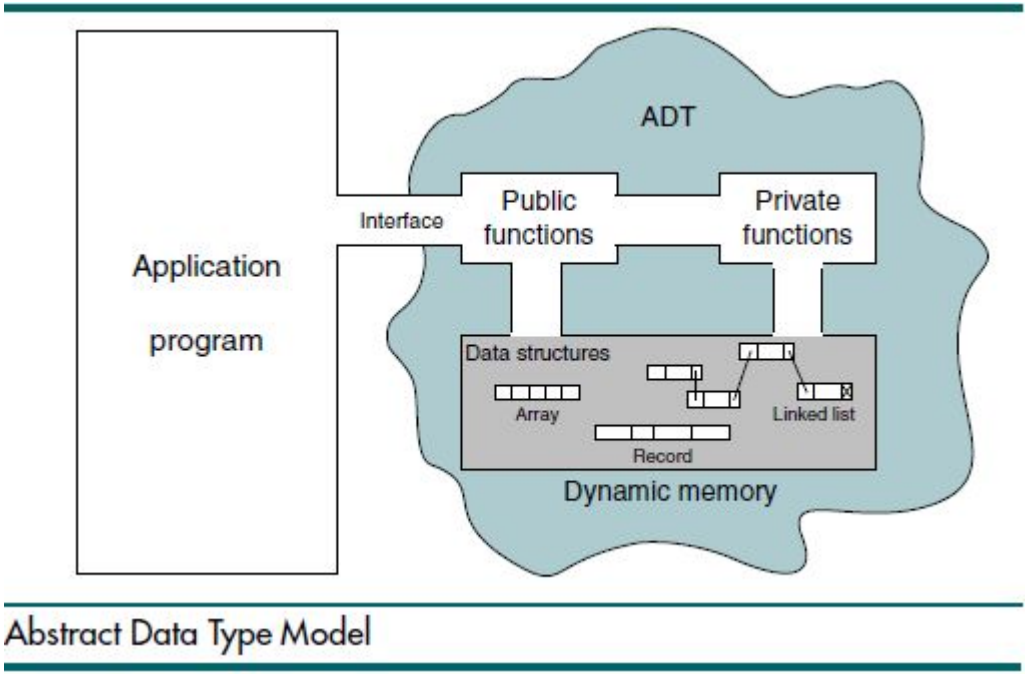


- Se definimos a nossa lista como um TAD, os usuários podem não estar cientes da estrutura usada. Desde que seja possível realizar operações de inserção e remoção de dados, não fara diferença como esses dados são armazenados.

Tipo Abstrato de Dados (TAD)

Modelo

- O modelo de TAD é mostrado na Figura.



Abstract Data Type Model

Tipo Abstrato de Dados (TAD)

Modelo

- Na figura, a área irregular representa o TAD.
- Dentro do TAD temos dois componentes diferentes:
    - Estruturas de Dados;
    - Funções (Pública e Privadas).

- Observe que ambos componentes se encontram completamente dentro do modelo TAD e que não se encontram no escopo do programa de aplicação.
- Por outro lado, as estruturas de dados são acessíveis para todas as funções TADs, assim como também qualquer função pode chamar uma outra para completar as suas tarefas.

Tipo Abstrato de Dados (TAD)

Operações do TAD

- Dados são ingressados, acessados, modificados e eliminados através de uma **interface externa** desenhada como caminho de acesso que esta parcialmente dentro e parcialmente fora do TAD.
- Somente as funções públicas são acessíveis através desta interface.
- Para cada operação TAD existe um algoritmo que realiza uma tarefa específica.
- Somente o nome da operação e seus parâmetros estão disponíveis para a aplicação e elas fornecem a única interface para a TAD.

Tipo Abstrato de Dados (TAD)

Estrutura de Dado do TAD

- Como um Tipo Abstrato de Dados deve ocultar a implementação de seu usuário, todos os dados referentes a estrutura devem ser mantidos dentro da TAD.
- Somente encapsular a estrutura não é suficiente.
- Também é necessário que diversas versões da estrutura possam co-existir.

Tipo Abstrato de Dados (TAD)

Estruturas

- Estudar as seguintes estrutura de dados:
  - Pilhas, Filas, Listas, Arvores Binárias;
- Desenvolver tais estruturas como TADs.

Tipo Abstrato de Dados (TAD)

Implementações TAD

- Existem duas estruturas básicas que podemos utilizar para implementar uma lista TAD:
  - Vetores e Listas Encadeadas.

Tipo Abstrato de Dados (TAD)

Implementações de Vetor

- Em um vetor a sequencialidade da lista é garantida pela estrutura de ordenação dos elementos do vetor (índices);
- A busca de um elemento individual pode ser bastante eficiente;
- Já a adição e remoção de elementos são processos complexos e ineficientes.
- Por este motivo as implementações de vetores são raramente usadas, especialmente quando a lista precisa mudar com frequência.
- Além disso, implementações de vetor para listas não-lineares podem se tornar excessivamente grandes, quando há muitos sucessores para cada elemento.

Tipo Abstrato de Dados (TAD)
Implementações de Lista Encadeada

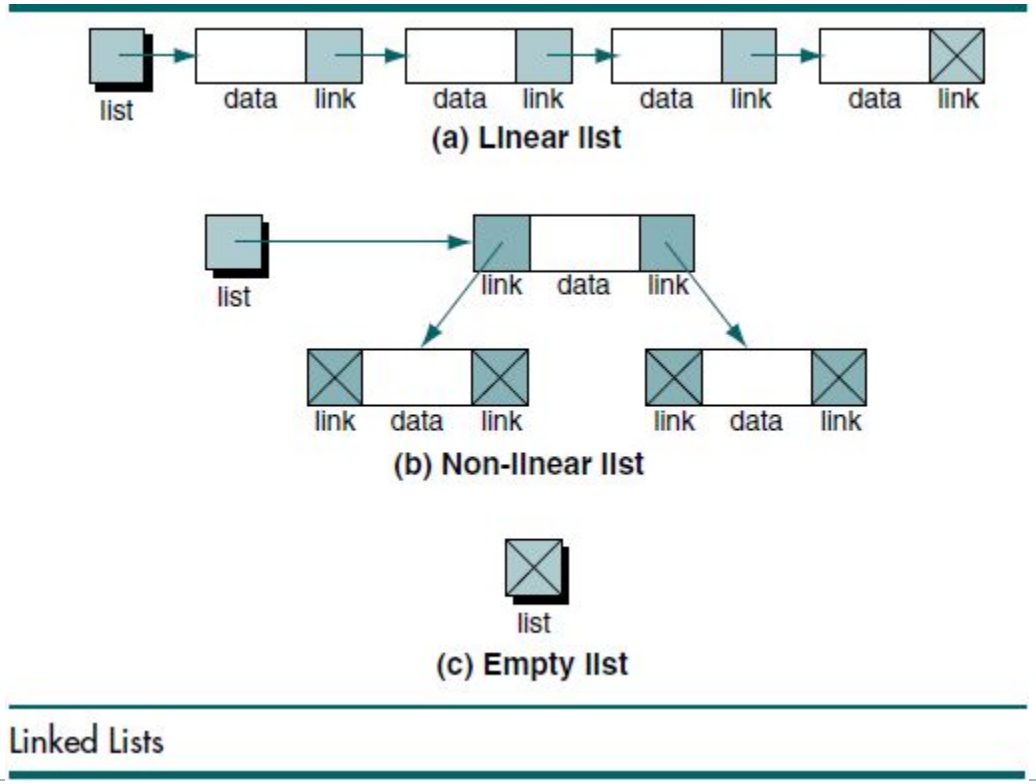
- Uma lista encadeada é uma coleção ordenada de dados em que cada elemento contêm a localização do seguinte elemento ou elementos.
- Cada elemento contêm duas partes:
    - Dados e uma ou mais conexões (*links*).
- A parte dos dados armazena os dados da aplicação. Dados que serão processados;
- Os links são usados para encadear os dados entre si. Contêm ponteiros que identificam o próximo elemento ou elementos na lista.
- Listas encadeadas podem servir para criar listas lineares e não-lineares. Nas listas lineares, cada elemento tem de zero a um sucessor. Já nas listas não-lineares, cada elemento tem zero, um ou mais sucessores.

Tipo Abstrato de Dados (TAD)
Implementações de Lista Encadeada

- A principal vantagem das listas encadeadas frente aos vetores é que os dados podem ser inseridos e removidos com facilidade.
- Não é necessário, mover elementos para fazer espaço na estrutura ao inserir nem para compactar os dados ao eliminar.
- Não entanto, a busca de um elemento individual fica limitada a ser sequencial uma vez que os elementos não se encontram fisicamente contíguos. Busca binária não é possível.

Tipo Abstrato de Dados (TAD)
Implementações de Lista Encadeada

- A figura ilustra: a) o conceito de lista linear implementado mediante lista encadeada; b) uma lista não-linear implementado mediante lista encadeada; c) uma lista vazia, linear ou não-linear. Definido como um ponteiro nulo.



Tipo Abstrato de Dados (TAD)

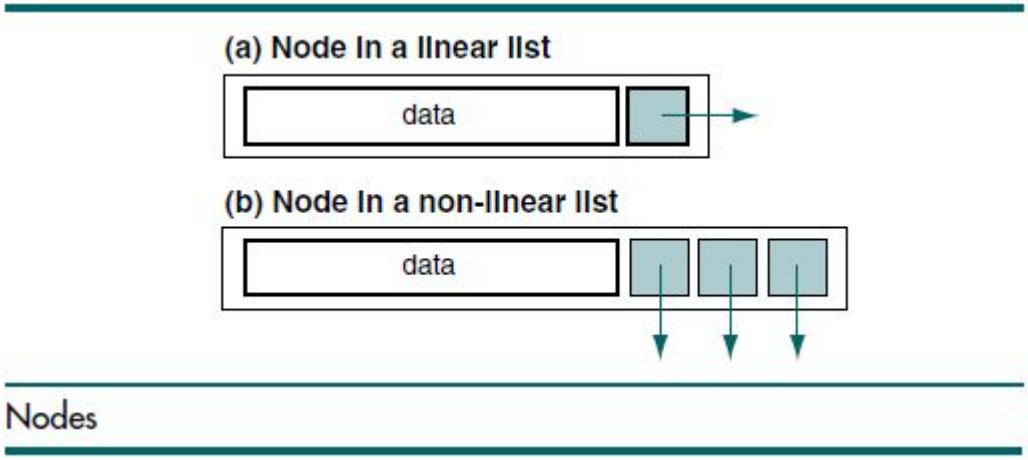
Nós em Listas Encadeadas

- Na implementação de listas encadeadas, os elementos da lista são chamados de nós.
- Um nó é uma estrutura que possui duas partes:
    - Dados e uma ou mais conexões (*links*).

Tipo Abstrato de Dados (TAD)

Nós em Listas Encadeadas

- A figura mostra dois tipos de nós diferentes: Um para lista linear e outro para lista não-linear.

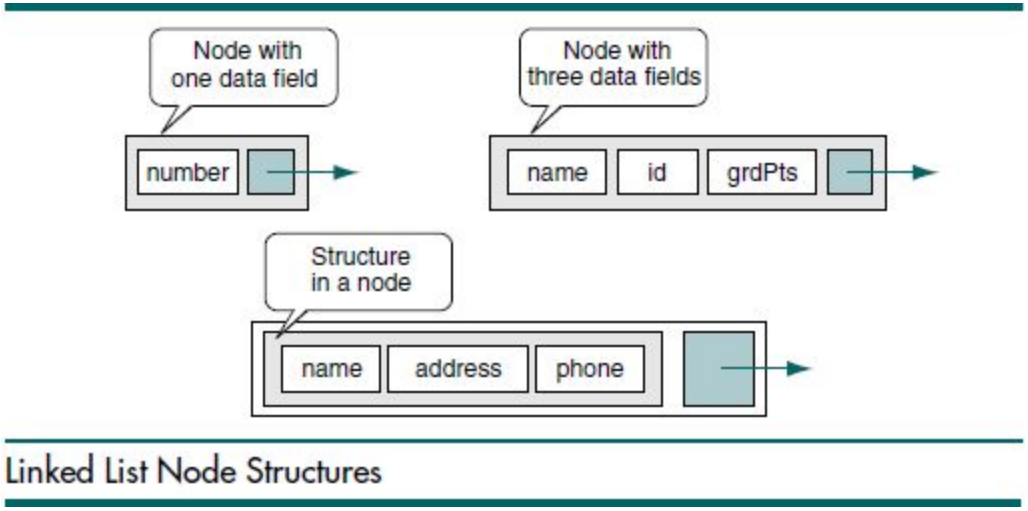


- Os nós em uma lista encadeada são chamados de estruturas auto-referenciais. Cada instância da estrutura contêm um ou mais ponteiros a outras instâncias do mesmo tipo estrutural.
- Na figura, as caixas coloridas com setas, representam os ponteiros que fazem que os nós sejam estruturas auto-referenciais.

Tipo Abstrato de Dados (TAD)

Nós em Listas Encadeadas

- A parte de dados em um nó pode ser:
  - um campo simples, vários campos, ou uma estrutura que contêm vários campos.
- A figura mostra três exemplos destes casos para um nó de uma lista linear.



Linked List Node Structures

Tipo Abstrato de Dados (TAD)

Ponteiros a Listas Encadeadas

- Uma lista encadeada deve sempre ter um ponteiro cabeçalho.
- Dependendo em como utilizaremos a lista, podemos ter outros ponteiros adicionais.
    - p.e. se vamos realizar uma busca em uma lista encadeada, podemos querer utilizar um ponteiro adicional para indicar a localização do dado que estávamos buscando.
    - Em outros casos, o processamento da lista pode ser mais eficiente se existe um ponteiro ao último nó da lista além do ponteiro cabeçalho.

Código Genérico para TADs

- Em estrutura de dados podemos criar código genérico para tipos abstratos de dados TADs.
- O código genérico nos permite escrever um conjunto de código e aplicá-lo a qualquer tipo de dado.
    - Por exemplo, podemos escrever funções genéricas para implementar uma estrutura de pilha. Podemos usar as funções genéricas para implementar pilhas de inteiros, float, double, e assim por diante.
- Linguagens como C++ e java proporcionam ferramentas especiais para manipular código genérico.
- A linguagem C tem a sua capacidade limitada através de dois recursos:
    - Ponteiro a void;
    - Ponteiro a função

Referências

- Gilberg, R.F. e Forouzan, B. A. Data Structures_A Pseudocode Approach with C. Capítulo 1. Basic Concepts. Segunda Edição. Editora Cengage, Thomson Learning, 2005.