UNIVERSIDADE FEDERAL DE ITAJUBÁ

Algoritmos e Estrutura de Dados I

COM 111

Tipo Abstrato de Dados – TAD

Vanessa Cristina Oliveira de Souza

Introdução

- Durante a implementação de uma solução computacional normalmente é necessário manipular dados obtidos como entrada.

- Exemplo :
  - Ordenar um conjunto de números inteiros
    - *Ler um conjunto de números inteiros*
    - *Armazená-los em alguma estrutura*
    - *Aplicar um algoritmo de ordenação*
    - *Mostrar a saída com os números ordenados*

Introdução

- O QUE VOCÊ PRECISA GUARDAR?
- COMO VOCÊ PRECISA GUARDAR?
- O QUE VOCÊ PRECISA FAZER COM ESSES DADOS?

Introdução

- O QUE VOCÊ PRECISA GUARDAR?
  - Dado
- COMO VOCÊ PRECISA GUARDAR?
  - Estrutura de Dados
- O QUE VOCÊ PRECISA FAZER COM ESSES DADOS?
  - Operações

Introdução

- O QUE VOCÊ PRECISA GUARDAR?
  - Dado
- C
  -
- O QUE VOCÊ PRECISA FAZER COM ESSES DADOS?
  - Operações

QUANDO OS DADOS SÃO DISPOSTOS E MANIPULADOS DE FORMA HOMOGÊNEA, CONSTITUEM UM *TIPO ABSTRATO DE DADOS*

Tipo Abstrato de Dados (TAD)

- Um tipo abstrato de dados é formado por um **conjunto de valores** e por uma **série de operações** que podem ser aplicadas sobre esses valores.

- Um TAD estabelece o conceito de tipo de dado divorciado da sua representação (modelo matemático).

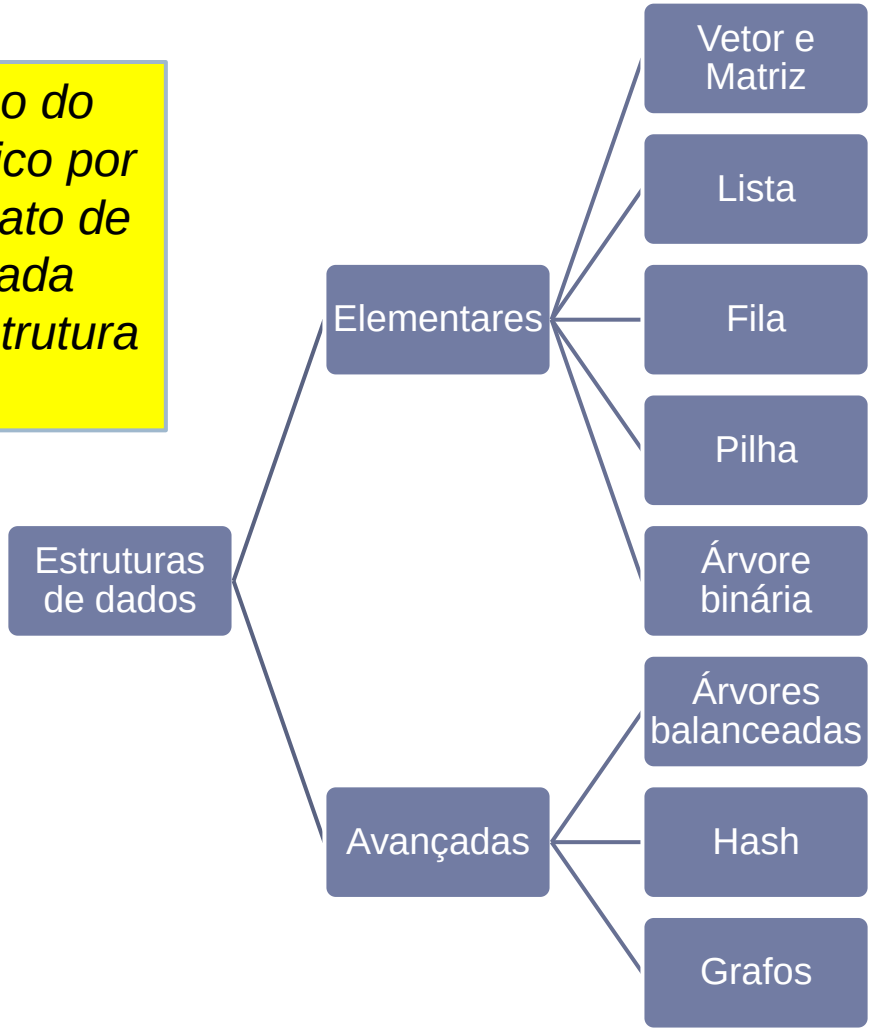
- Na representação de um TAD, emprega-se uma estrutura de dados.

Tipo Abstrato de Dados (TAD)

- Exemplo:
  - O conjunto dos inteiros acompanhado das operações de adição, subtração, multiplicação e divisão forma um exemplo de um tipo abstrato de dados.

Estruturas de Dados Dinâmicas

A representação do modelo matemático por trás do tipo abstrato de dados é realizada mediante uma estrutura de dados.



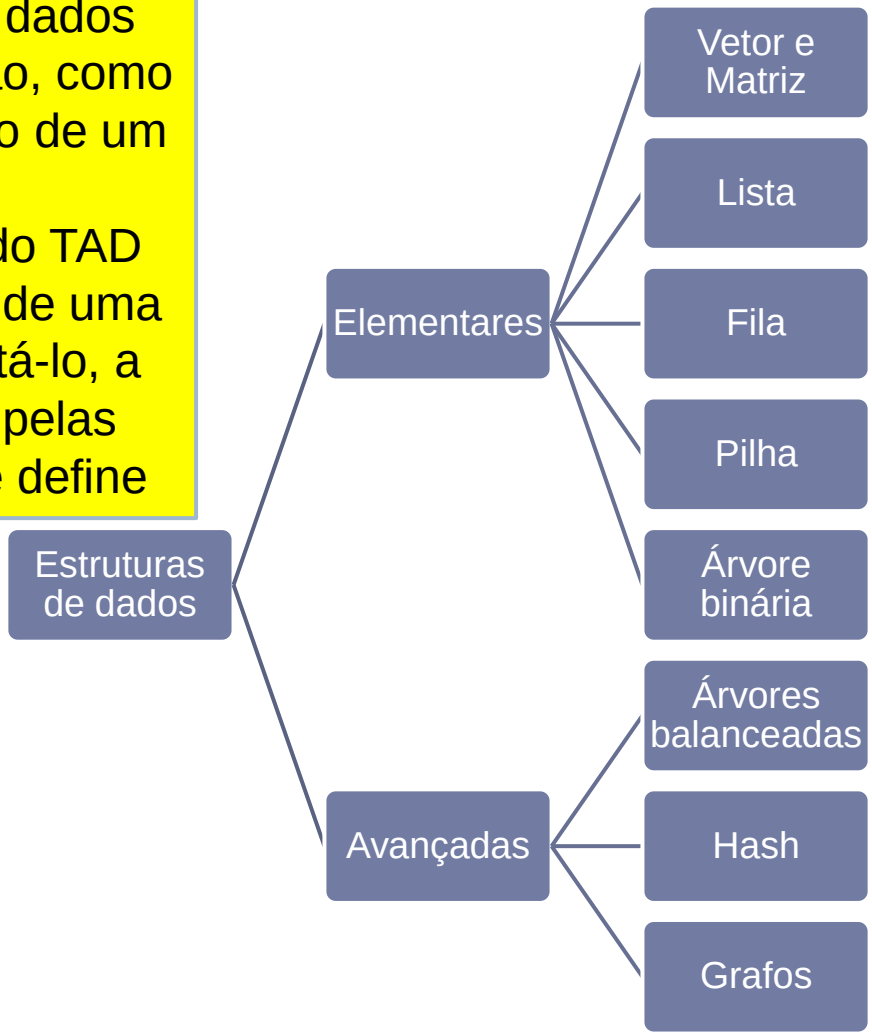
Principais TADs

Cada estrutura tem seu próprio conjunto de operações que lhe são pertinentes.

Principais TADs

Uma estrutura de dados pode ser vista, então, como uma implementação de um TAD

• implementação do TAD implica na escolha de uma ED para representá-lo, a qual é acessada pelas operações que ele define



Tipo Abstrato de Dados (TAD)

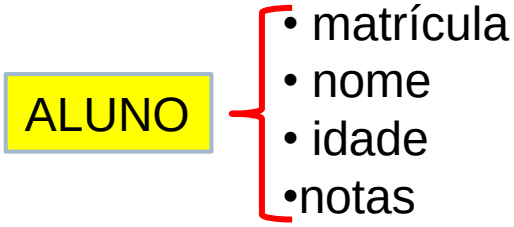
- Com o TAD podemos **encapsular** (esconder) de quem usa um determinado tipo de dado a forma concreta como este tipo foi implementado.
- Especifica o tipo de dado (domínio e operações) sem referência a detalhes da implementação
- Minimiza código do programa que usa detalhes de implementação
  - Mudanças na implementação acarretam menor impacto nos programas
  - Minimiza custos
- Os programas que usam o TAD não “conhecem” as implementações dos TADs
  - Fazem uso do TAD através de operações

Tipo Abstrato de Dados (TAD)

- Exemplo : considere uma aplicação que utilize uma lista de alunos.
  - Conjunto de valores : LISTA de alunos
  - Operações:
    - Crie a lista vazia
    - Insira um aluno na lista
    - Exclua um aluno na lista
    - Calcule a média do aluno
  - Para quem utilizará esse TAD, 'não importa' como ele foi implementado :
    - Lista como vetor
    - Lista encadeada
    - Lista duplamente encadeada
    - ...

Tipo Abstrato de Dados (TAD)

- O QUE VOCÊ PRECISA GUARDAR?
  - Dado



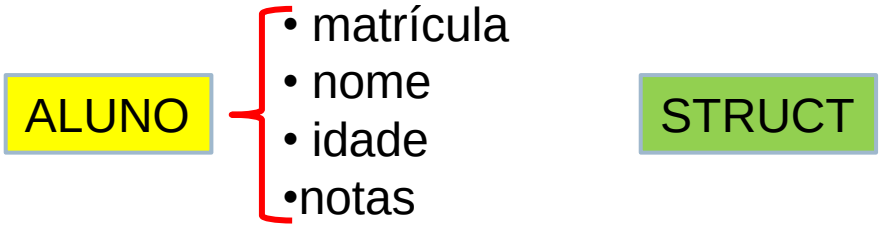
- COMO VOCÊ PRECISA GUARDAR?
  - Estrutura de Dados VETOR

- O QUE VOCÊ PRECISA FAZER COM ESSES DADOS?
  - Operações

•Criar lista de alunos
•Incluir aluno
•Excluir aluno
•Calcular média

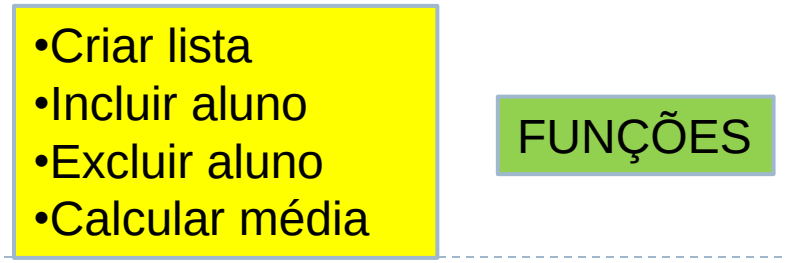
Tipo Abstrato de Dados (TAD)

- O QUE VOCÊ PRECISA GUARDAR?
  - Dado



- COMO VOCÊ PRECISA GUARDAR?
  - Estrutura de Dados

- O QUE VOCÊ PRECISA FAZER COM ESSES DADOS?
  - Operações

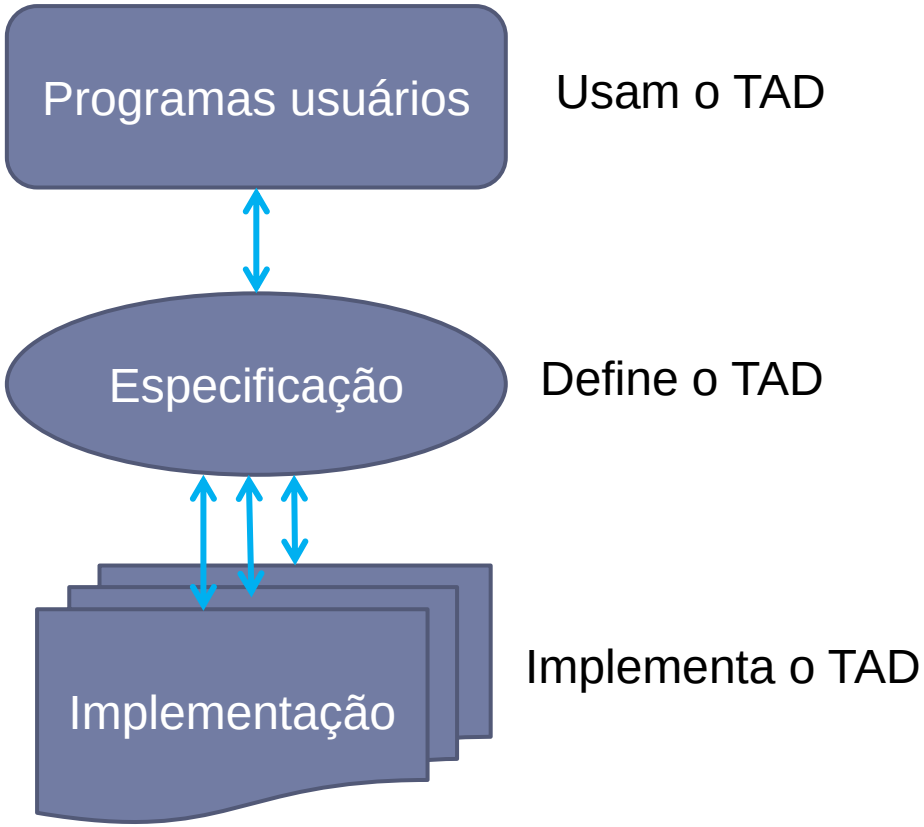


Tipo Abstrato de Dados (TAD)

- Quando usamos TAD's, nossos sistemas ficam divididos em:
  - Programas usuários:
    - A parte que usa o TAD
  - Implementação:
    - A parte que implementa o TAD

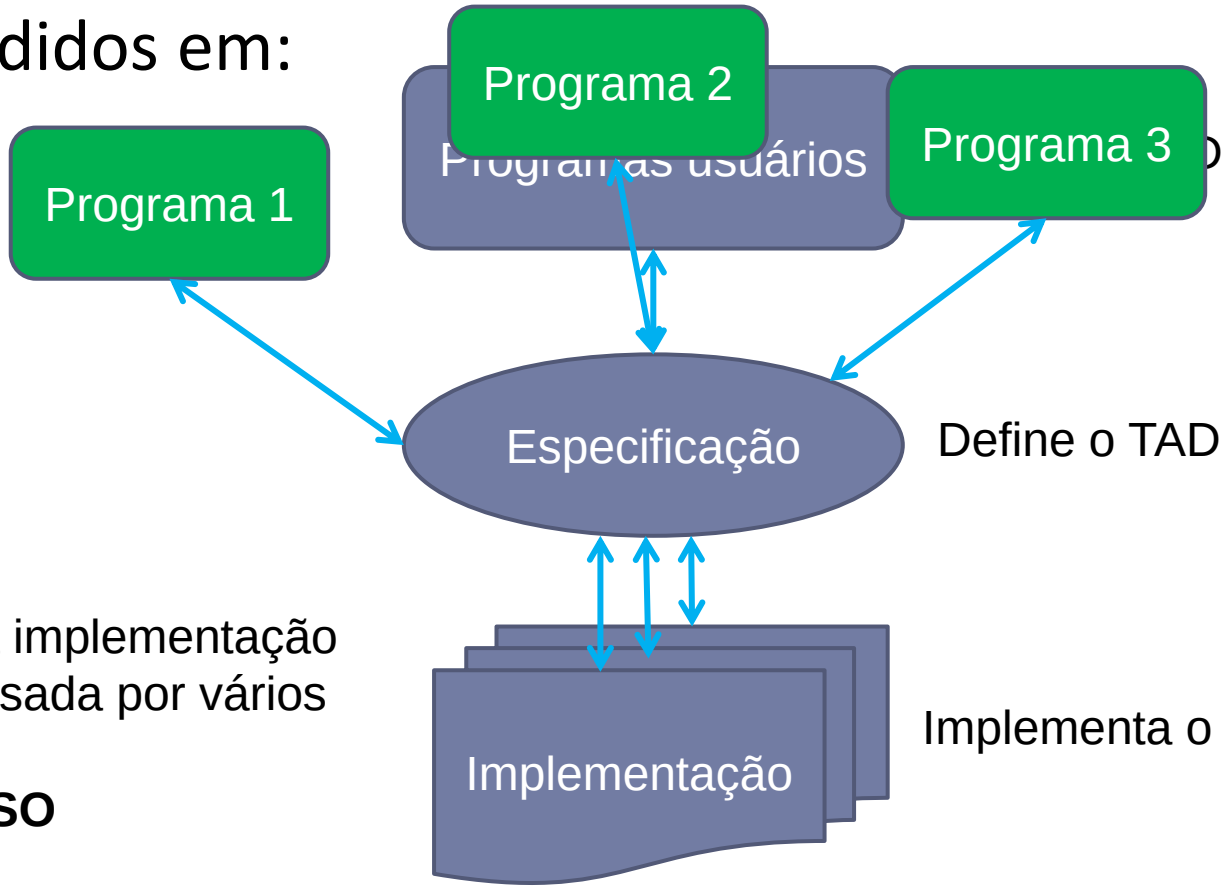
Tipo Abstrato de Dados (TAD)

- Quando usamos TAD's, nossos sistemas ficam divididos em:



Tipo Abstrato de Dados (TAD)

- Quando usamos TAD's, nossos sistemas ficam divididos em:



• A mesma implementação pode ser usada por vários programas

•**REUSO**

Vantagens do uso de TADs

- Reuso: uma vez definido, implementado e testado, o TAD pode ser acessado por diferentes programas
- Manutenção: mudanças na implementação do TAD não afetam o código fonte dos programas que o utilizam (decorrência do ocultamento de informação)
  - módulos do TAD são compilados separadamente
  - uma alteração força somente a recompilação do arquivo envolvido e uma nova link-edição do programa que acessa o TAD
  - O programa mesmo não precisa ser recompilado!
- Correção: TAD foi testado e funciona corretamente

Tipo Abstrato de Dados em C

TAD em C

- O uso de TAD em C é possível pois C permite modularização de programas.
  - *Programação modular é um paradigma de programação no qual o desenvolvimento das rotinas de programação é feito através de módulos, que são interligados entre si através de uma interface comum.*

- Característica essencial de TAD é a separação entre a definição conceitual e a implementação (ED específica)
  - O programa só acessa o TAD por meio de suas operações, a ED nunca é acessada diretamente
  - "ocultamento de informação"

TAD em C

- O uso de TAD em C é possível pois C permite modularização de programas.

- Para definir um TAD o programador o descreve em dois módulos separados:
  - Um módulo contém a interface de acesso
    - define o nome do tipo e os nomes das funções exportadas
  - Um módulo contém a definição do TAD
    - define a composição da estrutura do tipo
    - Inclui a implementação das funções externas
  - Outros programadores podem, por meio da interface de acesso, usar o TAD sem conhecer os detalhes representacionais e sem acessar o módulo de definição
    - não poderão acessar diretamente os campos da estrutura definida
    - só terão acesso aos dados obtidos através das funções exportadas

TAD em C

- O uso de TAD em C é possível pois C permite modularização de programas.

- Para definir um TAD o programador o descreve em dois módulos separados:
  - Um módulo contém a interface de acesso
    - define o nome do tipo e os nomes das funções exportadas
  - Um módulo contém a definição do TAD
    - define a composição da estrutura do tipo
    - Inclui a implementação das funções externas
  - Outros programadores podem, por meio da interface de acesso, usar o TAD sem conhecer os detalhes representacionais e sem acessar o módulo de definição
    - não poderão acessar diretamente os campos da estrutura definida
    - só terão acesso aos dados obtidos através das funções exportadas

TAD em C

- O conjunto de funções de cada biblioteca é descrito em um arquivo-interface (*header-file*), que tem o mesmo nome da biblioteca e sufixo .h.
  - essa interface também é conhecida como API, ou *application programming interface*

- Tecnicamente, uma **biblioteca de funções** é diferente de um **arquivo de funções** compilado separadamente. Quando as rotinas em uma biblioteca são linkadas com o restante do seu programa, apenas as funções que seu programa realmente usa são carregadas e linkadas. Em um arquivo compilado separadamente, todas as funções são carregadas e linkadas com seu programa.

TAD em C

- **Exercício**:
  - Crie um TAD para uma estrutura matriz, que armazena, além da matriz, as informações de linha e coluna da mesma.
  - Sendo assim:
    - Linhas : quantidade de linhas da matriz
    - Colunas : quantidade de colunas da matriz
    - v : a matriz de float, com as dimensões Linhas x Colunas

```
struct matriz
{
        int linhas;
        int colunas;
        float** v;
};
```

TAD em C

- **Exercício**:
  - As dimensões da matriz (linhas e colunas) devem ser informadas pelo usuário em tempo de execução do programa.

TAD em C

- **Exercício**:
  - O TAD *matriz* deverá ter as seguintes funções/operações:
    - Função cria - Aloca e retorna matriz m por n
    - Função libera - Libera a memória de uma matriz
    - Função acessa - Retorna o valor do elemento [i][j]
    - Função atribui - Atribui valor ao elemento [i][j]
    - Função linhas - Retorna o no. de linhas da matriz
    - Função colunas - Retorna o no. de colunas da matriz
    - Função preencheAleatoriamente – Preenche a matriz aleatoriamente
    - Função imprime – Imprime a matriz com dimensões m por n

TAD em C

- **Exercício**:
  - Faça um programa usuário para o TAD *matriz*.
  - Utilizando as funções disponibilizadas pelo TAD, o programa usuário deverá fazer as seguintes operações:
    - Solicitar ao usuário as dimensões da matriz e alocar a estrutura
    - Preencher a matriz aleatoriamente
    - Imprimir a matriz
    - Imprimir uma posição específica da matriz
    - Trocar o valor dessa posição e imprimi-la novamente

TAD em C

- **Exercício**:

```
Digite a quantidade de linhas da matriz : 4
Digite a quantidade de colunas da matriz : 7
Estrutura alocada com sucesso
41.00    67.00    134.00   100.00   169.00   124.00   78.00
158.00   162.00   64.00    105.00   145.00   81.00    27.00
161.00   91.00    195.00   142.00   27.00    36.00    191.00
4.00     102.00   153.00   92.00    182.00   21.00    116.00

A posicao 2, 0 da matriz possui o valor 161.00

A posicao 2, 0 da matriz possui o valor 5.15
```

Ponteiros para Estruturas em C

- **Operador seta ->**
  - O operador seta é usado no lugar do ponto quando se está acessando um elemento de estrutura por meio de um ponteiro para a estrutura.
    - void imprimeDimensoesMatriz (struct matriz mat)
      - printf("%d x %d", mat.linhas, mat.colunas);
    - void imprimeDimensoesMatriz (struct matriz *mat)
      - printf("%d x %d", mat->linhas, mat->colunas);

```
struct matriz
{
        int linhas;
        int colunas;
        float** v;
};
```