Tipo Abstrato de Dados (TAD)

Profª. Elisa de Cássia Silva Rodrigues

Prof. Pedro Henrique Del Bianco Hokama

Profª. Vanessa Cristina Oliveira de Souza

Introdução

- Tipo de dado:
  - Conjunto de valores que uma variável pode assumir.
  - Exemplo:
    - `char`, `int`, `float` etc (representando dados independentes).
- Estrutura de dados:
  - Conjunto de tipos de dados com algum relacionamento lógico.
  - Exemplo:
    - Tipos `int` e `char` que representam CPF e nome de uma pessoa.
    - Estrutura de dados Pessoa (definido através de um `struct` em C.)

Tipo Abstrato de Dados (TAD)

- Definição:
  - Técnica de programação usada para encapsular um conjunto de dados estruturados (*struct*) que podem ser manipulados através das operações definidas como funções.
- Características:
  - O usuário utiliza o TAD por meio de sua interface.
  - Os dados da estrutura não podem ser manipulados diretamente.
  - A interação com os dados deve ser feita apenas através de funções.
  - A implementação do TAD não está vinculada a sua utilização.

Vantagens de usar um TAD

- Encapsulamento:
  - A implementação do TAD é ocultada do usuário.
  - Apenas um conjunto de operações é fornecido para manipular o TAD.
  - Dispensa conhecer a implementação, tornando seu uso mais fácil.
- Segurança:
  - O programa usuário do TAD não tem acesso direto aos dados.
  - Evita manipulação imprópria dos dados.

Vantagens de usar um TAD

- Flexibilidade:
  - Modificações no TAD não implicam em atualização das aplicações.
  - É possível ter diversas implementações de um TAD, desde que a interface seja a mesma.
- Reutilização:
  - A implementação do TAD é independente do programa do usuário.
  - Um TAD pode ser utilizado por diversos programas.

Exemplo de TAD na Linguagem C

- Estrutura usada para a manipulação de arquivos: `FILE`.
  - É definida na biblioteca *stdio.h*.
  - Para utilizar essa estrutura deve-se declarar um ponteiro desse tipo:
    - `FILE *f;`
- Operações:
  - `fopen()`: função para criar um arquivo.
  - `fclose()`: função para fechar um arquivo.
  - `fputc()`: função para inserir um caractere.
  - `fgetc()`: função para ler um caractere.
  - `feof()`: verifica se está no fim do arquivo.

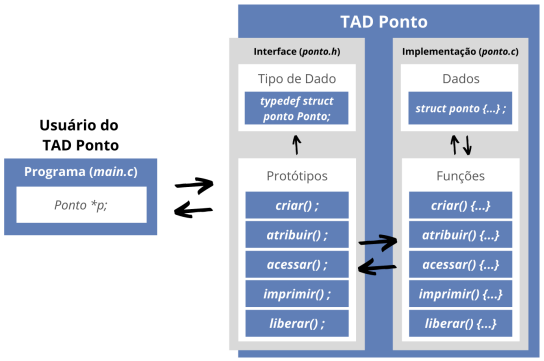
Modularização

- Definição:
  - Técnica de programação que utiliza o conceito de módulos.
  - Módulo é uma unidade independente que tem uma função específica.
  - Pode ser facilmente reutilizado.
  - Pode ser modificado independente do programa do usuário.
- O uso de módulos evita problemas como:
  - Dificuldade de manutenção do código.
  - Reutilização de código pelo processo de copiar e colar.
  - Recompilação de todo o código a cada modificação.

Modularização

- O conceito de TAD envolve modularizar o código a fim de separar a interface da implementação.
- Para isso, define-se dois arquivos:
  - Arquivo ".h" (interface visível ao usuário):
    - Definição dos tipos de dados (ponteiros) e dos dados visíveis.
    - Declaração dos protótipos das funções visíveis ao usuário.
  - Arquivo ".c" (implementação oculta do usuário):
    - Definição dos tipos de dados ocultos (`struct`).
    - Implementação das funções.

Exemplo de Modularização



Implementação: https://repl.it/@elisa_rodrigues/Modulo4-TAD-Ponto

Referências Bibliográficas

1. BACKES, A. ***Estrutura de Dados Descomplicada em Linguagem C***. 2016.

   Vídeo aulas (1ª a 2ª):

   https://programacaodescomplicada.wordpress.com/indice/estrutura-de-dados/.