

UFMG  
UNIVERSIDADE FEDERAL  
DE MINAS GERAIS

## Programação e Desenvolvimento de Software 2

### Tipos Abstratos de Dados

Prof. Douglas G. Macharet  
douglas.macharet@dcc.ufmg.br

DCC  
DEPARTAMENTO DE  
CIÊNCIA DA COMPUTAÇÃO

## Introdução

- Algoritmo
  - Sequência de ações executáveis
  - Transformam uma entrada em uma saída
  - Trabalham sobre estruturas de dados
- Estruturas de dados
  - Informações que representam uma situação real
  - Abstração da realidade
  - Suportam as operações dos algoritmos

DCC UFMG PDS 2 - Tipos Abstratos de Dados 2

## Introdução

- Como representar os dados no programa?
- O que é um tipo de dado?
- O que os tipos “int” e “bool” representam?
  - Em quais operações podem ser utilizados?
- Programa possuirá apenas tipos como esses?
  - Primitivos
  - Como criar conceitos mais genéricos?

DCC UFMG PDS 2 - Tipos Abstratos de Dados 3

## Introdução

- Diferença entre Programa e Algoritmo?
  - Um programa é uma realização concreta de um algoritmo abstrato, baseado em representações de dados específicas
  - Programas precisam ser implementados numa linguagem que pode ser entendida e seguida pelo computador
    - C, C++, Java, Python, ...

DCC UFMG PDS 2 - Tipos Abstratos de Dados 4

## Introdução

- Quando um programa é considerado bom?
  1. Funciona (faz o que foi especificado)
  2. Fácil de entender e modificar
  3. Razoavelmente eficiente
- Bons programas fazem uso de Abstração
- Tipos Abstratos de Dados
  - Ajudam alcançar (2), que facilita (1)

DCC UFMG PDS 2 - Tipos Abstratos de Dados 5

## Tipos Abstratos de Dados (TADs)

- Generalização de tipos primitivos
- Encapsula uma estrutura de dados
  - Conjunto de valores
  - Conjunto de operações sobre esses valores
- Usuário do TAD x Programador do TAD
  - Acesso às operações disponibilizadas
  - Conhece a interface, não a implementação

DCC UFMG PDS 2 - Tipos Abstratos de Dados 6



## Structs

- Coleção de variáveis, possivelmente de tipos diferentes, juntas sob um único nome
- Ponto 3D
  - $x, y, z$
  - Ao invés de criar três variáveis, é possível criar uma única variável contendo três campos.
- Em C++ uma struct é similar à uma classe
  - Diferença está no nível de proteção

DCC

PDS 2 - Tipos Abstratos de Dados

13

## Structs

### Exemplo 1

```
#include <cstdlib>
#include <iostream>
#include <cmath>

using namespace std;

struct Ponto3D {
    float x;
    float y;
    float z;

    float calcularDistancia(Ponto3D* p2) {
        float dx = p2->x - x;
        float dy = p2->y - y;
        float dz = p2->z - z;
        return sqrt(dx*dx + dy*dy + dz*dz);
    }
};
```

DCC

PDS 2 - Tipos Abstratos de Dados

14

## Structs

### Exemplo 1

```
int main() {

    Ponto3D* p1 = new Ponto3D;
    p1->x = 0;
    p1->y = 0;
    p1->z = 0;

    Ponto3D* p2 = new Ponto3D;
    p2->x = 5;
    p2->y = 5;
    p2->z = 5;

    cout << p1->calcularDistancia(p2) << endl;

    delete p1;
    delete p2;

    return 0;
}
```

DCC

PDS 2 - Tipos Abstratos de Dados

15

## Structs

### Exemplo 2

- Como representar um Aluno?
  - Quais atributos/dados?
    - Nome, matrícula, curso, RSG...
  - Quais operações sobre esses dados?
    - Matricular, Calcular RSG, ...

DCC

PDS 2 - Tipos Abstratos de Dados

16

## Structs

### Exemplo 2

```
#include <iostream>
#include <string>

using namespace std;

struct Aluno {
    string nome;
    int matricula;

    float calcularRSG() {
        // Aqui a implementação necessaria
        return 0;
    }
};

int main() {
    Aluno aluno;
    aluno.nome = "Jose da Silva";
    aluno.matricula = 201812345;
    cout << aluno.nome << endl;
    return 0;
}
```

DCC

PDS 2 - Tipos Abstratos de Dados

17

## Structs

- Essas implementações seguem corretamente os conceitos de TADs que vimos?
- Qual o problema?
  - Especificação e Implementação estão juntas
- Como resolver isso?
  - Separar → Modularizar
  - Alguma ideia de como implementar?

DCC

PDS 2 - Tipos Abstratos de Dados

18

## Modularização

- Separação em arquivos
  - Especificação: NomeDoTAD.hpp
  - Implementação: NomeDoTAD.cpp
- Parte do programa e outros TADs que o utilizam devem dar `#include` no arquivo `.hpp`
- O arquivo `.hpp` define o contrato do TAD

DCC

PDS 2 - Tipos Abstratos de Dados

19

## Modularização

### Exemplo 3

- Como representar uma Circunferência?
  - Quais atributos/dados?
    - Coordenadas centro, raio, ...
  - Quais operações sobre esses dados?
    - Calcular Área, Perímetro, ...

DCC

PDS 2 - Tipos Abstratos de Dados

20

## Modularização

### Exemplo 3

```
#ifndef M_PI
#define M_PI 3.14159265358979323846
#endif

#ifndef CIRCUNFERENCIA_H
#define CIRCUNFERENCIA_H

#include <cmath>

struct Circunferencia {
    double _x, _y;
    double _raio;

    Circunferencia(double, double, double);
    double calcularArea();
};

#endif
```

Circunferencia.hpp

[https://en.cppreference.com/w/cpp/string/basic/basic\\_string\\_view](https://en.cppreference.com/w/cpp/string/basic/basic_string_view)

DCC

PDS 2 - Tipos Abstratos de Dados

21

## Modularização

### Exemplo 3

```
#include "Circunferencia.hpp"

Circunferencia::Circunferencia(double x, double y, double raio) {
    _x = x;
    _y = y;
    _raio = raio;
}

double Circunferencia::calcularArea() {
    return M_PI * pow(_raio, 2);
}
```

Circunferencia.cpp

Operação sendo implementada.

DCC

PDS 2 - Tipos Abstratos de Dados

22

## Modularização

### Exemplo 3

```
#include <iostream>
#include "Circunferencia.hpp"

using namespace std;

int main() {
    Circunferencia* circ = new Circunferencia(0, 0, 10);
    cout << circ->calcularArea() << endl;

    delete circ;

    return 0;
}
```

main.cpp

g++ -std=c++11 -Wall main.cpp Circunferencia.cpp -o main

DCC

PDS 2 - Tipos Abstratos de Dados

23

## Tipos Abstratos de Dados (TADs)

### Definição

- As descrições desses tipos devem ser
  - Precisas e sem ambiguidades
  - Completas
    - Tão completas quanto o necessário
  - Não excessivamente detalhadas
    - Tão abstrata quanto possível

DCC

PDS 2 - Tipos Abstratos de Dados

24

## Tipos Abstratos de Dados (TADs)

### Operações

- Quais operações devem estar em um TAD?
- Filosofia egoísta
  - Se algo não é útil, então não é necessário
  - Informe ao usuário apenas o necessário
    - Lembre-se do princípio anterior
    - Desejo x Necessidade
    - Liberdade para alterações futuras

## Desenvolvimento

- Níveis de detalhamento
  - Especificação/Requisitos
    - Mundo real
  - Projeto/Desenho/Modelagem
    - Tipos Abstratos de Dados
  - Implementação
    - Classes

## Desenvolvimento

- Desenvolvimento Orientado a Objetos
  - Construção de programas como uma coleção estruturada de implementações (talvez) parciais de tipos de dados abstratos (classes)
- Modularização
  - Programação Orientada a Objetos
  - Desenvolvimento por contrato

## Desenvolvimento

### Contrato

- O que é um contrato?
  - Acordo entre duas ou mais partes
- Quais contratos existem em um programa?
  - Requisitos de funcionalidade, desempenho, ...
  - Operações disponíveis em um TAD
    - Entradas e saídas

## Desenvolvimento

### Contrato

- Como garantir que o contrato estará sendo cumprido pelo Usuário do TAD?
  - Tipos
  - Asserções
  - Exceções
- Requisitos  $\leftrightarrow$  Implementação
  - Garantir a consistência entre esses artefatos

## Desenvolvimento

- Qual implementação é melhor?

```
struct TriangRetangulo {
    float base, altura;
}
```

```
struct TriangRetangulo {
    float base, hipot, ang;
}
```

- A implementação não é o mais importante
  - Pense no tipo como um conjunto de operações
    - Criar, SetarBase, SetarAltura, CalcularArea, ...
  - Usuário deve acessar os dados pelas operações!

## Desenvolvimento

- Esses TADs são iguais ou diferentes?

```
struct Ponto {
    float x;
    float y;
}
```

```
struct Ponto {
    float r;
    float theta;
}
```

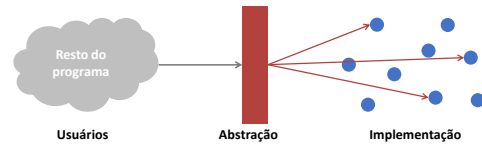
- Iguais (conceito) e Diferentes (dados)!
- Tipo/TAD
  - Elementos com propriedades semelhantes
  - Clientes dependem do conceito geral

DCC 

PDS 2 - Tipos Abstratos de Dados

31

## Desenvolvimento



- Implementação não deve ser conhecida
  - A abstração deve prover os métodos que serão utilizados para manipular os dados

DCC 

PDS 2 - Tipos Abstratos de Dados

32

## Exercícios

- Implemente TADs para representar
  - Curso
  - Disciplina
- Crie as associações necessárias
  - Estudante / Curso / Disciplina
- Implemente um programa simples que armazene e imprima a lista de alunos matriculados em uma determinada disciplina

DCC 

PDS 2 - Tipos Abstratos de Dados

33