Estruturas de Dados

Prof. Rodrigo Martins
rodrigo.martins@francomontoro.com.br

Cronograma da Aula

- Structs
- Typedef
- Exemplos
- Exercícios

O que são Registros (Structs) em C++?

- O conceito de orientação a objeto tem uma base muito sólida no conceito de estrutura de dados.
- Os structs em C++ são uma forma de definir tipos de dados personalizados que contêm um conjunto de campos (variáveis) que podem ter diferentes tipos de dados.
- As estruturas de dados consistem em criar apenas um dado que contém vários membros, que nada mais são do que outras variáveis.
- De uma forma mais simples, é como se uma variável tivesse outras variáveis dentro dela.

O que são Registros (Structs) em C++?

- A vantagem em se usar estruturas de dados é que podemos agrupar de forma organizada vários tipos de dados diferentes, por exemplo, dentro de uma estrutura de dados podemos ter juntos tanto um tipo float, um inteiro, um char ou um double.

- As variáveis que ficam dentro da estrutura de dados são chamadas de membros.

Como são Definidos?

- Para criar uma estrutura de dados usamos a palavra reservada struct.

- Toda estrutura deve ser criada antes de qualquer função ou mesmo da função principal main. Toda estrutura tem nome e seus membros são declarados dentro de um bloco de dados.

- Sintaxe:

```
struct nome_do_struct {
        tipo_de_dado campo1;
        tipo_de_dado campo2;
        //...
        tipo_de_dado campoN;
};
```

Como são Definidos?

- Onde:
  - **nome_do_struct** é o nome que você dá ao struct.
  - **tipo_de_dado** é o tipo de dado que será armazenado em cada campo do struct.
  - **campo1**, **campo2**, ..., **campoN** são os nomes dos campos do struct.

Como são Definidos?

- Por exemplo, um struct que define um ponto no plano cartesiano pode ser definido da seguinte forma:

```
struct Pessoa {
        string nome;
        int idade;
        float altura;
};
```

- Neste exemplo, Pessoa é um registro que contém três variáveis de tipos diferentes: nome (uma string), idade (um inteiro) e altura (um float).

Como são Definidos?

```
struct Carro {
        string marca;
        string modelo;
        int ano;
        float quilometragem;
        bool estaFuncionando;
};
```

- Neste exemplo, Carro é o nome do registro. Ele tem cinco membros:
  - marca: uma string representando a marca do carro.
  - modelo: outra string para o modelo do carro.
  - ano: um inteiro para o ano de fabricação do carro.
  - quilometragem: um float para a quilometragem atual do carro.
  - estaFuncionando: um booleano para indicar se o carro está funcionando ou não.

Uso do Registro

- Após declarar o registro, você pode utilizá-lo para criar variáveis desse tipo.

```
Carro meuCarro;
meuCarro.marca = "Toyota";
meuCarro.modelo = "Corolla";
meuCarro.ano = 2021;
meuCarro.quilometragem = 15000.5;
meuCarro.estaFuncionando = true;
```

- Neste exemplo, meuCarro é uma variável do tipo Carro, e cada um dos seus membros é acessado e modificado individualmente.

Para que Servem?

- São extremamente úteis para modelar dados mais complexos, como um objeto do mundo real (por exemplo, um carro, um livro, um empregado).

- Registros podem ser passados para funções, permitindo o envio de um grupo de variáveis de uma só vez.

- Da mesma forma, registros podem ser usados para retornar múltiplos valores de uma função.

Para que Servem?

- Registros podem ser usados como elementos de listas, árvores, e outras estruturas de dados, facilitando a organização e manipulação de conjuntos complexos de informações.

- Em C++, structs podem ter funções e podem ser usados de maneira similar às classes, servindo como uma base para a programação orientada a objetos.

struct1.cpp

```cpp
#include <iostream>

using namespace std;

struct Data
{
    int dia;
    int mes;
    int ano;
};

int main (void)
{
    setlocale(LC_ALL, "Portuguese");
    Data hoje;

    cout << "Que dia é hoje? ";
    cin >> hoje.dia;
    cout << "Qual mes? ";
    cin >> hoje.mes;
    cout << "Qual ano? ";
    cin >> hoje.ano;
    cout << "Hoje é " << hoje.dia << "/" << hoje.mes << "/" << hoje.ano << endl;
    system("pause");
    return 0;
}
```

Entendendo o Código

- A variável hoje é declarada como sendo um tipo de dado data. Data é uma estrutura de dados que tem três características (ou três membros) inteiros: dia, mes e ano.

- Como hoje é um tipo de dado data, ele obtém os mesmos três membros.

- Para acessar cada membro, usamos a variável e depois o nome do membro que queremos acessar separados por ponto (.).

struct2.cpp

```cpp
#include <iostream>

using namespace std;

struct Pessoa
{
    string nome;
    int idade;
};

int main()
{
    setlocale(LC_ALL, "Portuguese");
    Pessoa p;

    cout << "Qual seu nome? ";
    getline(cin, p.nome); //Lê uma linha inteira de caracteres de um fluxo de entrada
    //cin >> p.nome;

    cout << "Quantos anos voce tem? ";
    cin >> p.idade;

    cout << "Nome: " << p.nome << endl;
    cout << "Idade: " << p.idade << " anos" << endl;
    return 0;
}
```

struct3.cpp

```cpp
#include <iostream>

using namespace std;

struct Carro {
    string marca;
    string modelo;
    int ano;
};

int main() {
    setlocale(LC_ALL, "Portuguese");
    Carro carro;
    cout << "Digite a marca do carro: ";
    getline(cin, carro.marca);
    cout << "Digite o modelo do carro: ";
    getline(cin, carro.modelo);
    cout << "Digite o ano do carro: ";
    cin >> carro.ano;

    cout << "Carro: " << carro.marca << " " << carro.modelo << ", Ano: " << carro.ano << endl;
    return 0;
}
```

struct4.cpp

```cpp
#include <iostream>

using namespace std;

struct Livro {
    string titulo;
    string autor;
    int anoPublicacao;
    string disponivel;
};

int main() {
    setlocale(LC_ALL, "Portuguese");
    Livro livro;
    cout << "Digite o título do livro: ";
    getline(cin, livro.titulo);
    cout << "Digite o autor do livro: ";
    getline(cin, livro.autor);
    cout << "Digite o ano de publicação: ";
    cin >> livro.anoPublicacao;

```

struct4.cpp

```
22      char resposta;
23      cout << "O livro está disponível? (s/n): ";
24      cin >> resposta;
25      if (resposta == 's' || resposta == 'S'){
26         livro.disponivel = "Sim";
27      }
28      else{
29          livro.disponivel = "Não";
30      }
31
32      cout << "Livro: " << livro.titulo << "\nAutor: " << livro.autor << "\nAno de Publicação: " << livro.anoPublicacao;
33      cout << "\nDisponibilidade: " << livro.disponivel << endl;
34      return 0;
35  }
```

struct5.cpp

```cpp
#include <iostream>

using namespace std;

struct Veiculo {
    string marca;
    string modelo;
    int ano;
    string tipoCombustivel;
};

int main() {

    setlocale(LC_ALL, "Portuguese");
    Veiculo veiculo;
    cout << "Digite a marca do veículo: ";
    getline(cin, veiculo.marca);
    cout << "Digite o modelo do veículo: ";
    getline(cin, veiculo.modelo);
    cout << "Digite o ano do veículo: ";
    cin >> veiculo.ano;

    cout << "O veículo é a gasolina (G), diesel(D) ou elétrico (E)? ";
    char tipo;
    cin >> tipo;
```

struct5.cpp

```
26
27      switch(tipo) {
28      case 'G':
29      case 'g':
30          veiculo.tipoCombustivel = "Gasolina";
31          break;
32      case 'D':
33      case 'd':
34          veiculo.tipoCombustivel = "Diesel";
35          break;
36      case 'E':
37      case 'e':
38          veiculo.tipoCombustivel = "Elétrico";
39          break;
40      default:
41          veiculo.tipoCombustivel = "Indefinido";
42      }
43
44      cout << "Veículo: " << veiculo.marca << " " << veiculo.modelo << "\nAno: " << veiculo.ano;
45      cout << "\nTipo de Combustível: " << veiculo.tipoCombustivel << endl;
46
47      return 0;
48  }
```

struct6.cpp

```
25          cout << "Deseja adicionar outro aluno? (s/n): ";
26          cin >> continuar;
27          cin.ignore();  // Limpa o buffer após a entrada do caractere
28
29      } while (continuar == 's' || continuar == 'S');
30
31      return 0;
32  }
```

struct7.cpp

```cpp
#include <iostream>

using namespace std;

struct TemperaturaDia {
    int dia;
    float temperatura;
};

int main() {

    setlocale(LC_ALL, "Portuguese");
    TemperaturaDia tempDia;
    int contador = 0;

    cout << "Registro de Temperaturas Diárias. Digite -999 para encerrar." << endl;

```

struct7.cpp

```
18      do {
19          contador++;
20          cout << "Dia " << contador << " - Digite a temperatura: ";
21          cin >> tempDia.temperatura;
22
23          if (tempDia.temperatura != -999) {
24              tempDia.dia = contador;
25              cout << "Dia: " << tempDia.dia << " - Temperatura: " << tempDia.temperatura << "°C" << endl;
26          }
27
28      } while (tempDia.temperatura != -999);
29
30      cout << "Registro de temperaturas encerrado." << endl;
31
32      return 0;
33  }
```

struct8.cpp

*exemplo6.cpp

```cpp
#include <iostream>
#include <cstdlib>

using namespace std;

//Ponteiro de Struct

typedef struct data {
    short dia;
    short mes;
    int ano;
} Data;

int main (void){
    Data data; //variável data do tipo struct data
    Data *hoje; //ponteiro hoje para um tipo struct data
    hoje = &data; //hoje aponta para o endereço de data

    //dados sendo inseridos na variável data
    (*hoje).dia = 20;
    (*hoje).mes = 1;
    (*hoje).ano = 2009;

    //hoje->dia = 20;
    //hoje->mes = 1;
    //hoje->ano = 2009;

    //mostrando o que está gravado no endereço contido em hoje
    cout << "Data registrada:" << endl;
    cout << hoje->dia <<"/"<< hoje->mes <<"/"<< hoje->ano << endl;
    system ("pause");
}
```

Exercícios

1- Criar um programa que registre informações de vários pacientes, incluindo nome, idade e um sintoma específico. O programa deve permitir a entrada de dados para múltiplos pacientes até que o usuário decida parar.

2- Desenvolver um programa que registre detalhes de livros em uma biblioteca, como título, autor e ano de publicação. O programa deve continuar solicitando informações até que um título de livro específico seja inserido.

Exercícios

3 - Construir um programa para avaliar filmes, pedindo ao usuário para avaliar diferentes aspectos como enredo, atuação e efeitos especiais. Calcule a média das notas avaliadas em enredo, atuação e efeitos especiais. O programa deve permitir a avaliação de vários filmes. O programa deve parar quando usuário não desejar mais continuar.

4 - Criar um programa para registrar o desempenho de estudantes em uma disciplina, incluindo nome do estudante, nota final e se ele foi aprovado ou reprovado. O programa deve permitir a inserção de dados para vários estudantes até que o usuário escolha encerrar. Utilize vetor.

Exercícios

5 - Implementar um programa para registrar pedidos em um restaurante, coletando informações como o nome do prato, quantidade e preço total. O programa deve continuar recebendo novos pedidos até que um valor específico seja inserido como preço.

Typedef

- Em C++ podemos redefinir um tipo de dado dando-lhe um novo nome.
- Essa forma de programação ajuda em dois sentidos:
  - 1º. Fica mais simples entender para que serve tal tipo de dado;
  - 2º. É a única forma de conseguirmos referenciar uma estrutura de dados dentro de outra (struct dentro de struct).

Typedef

- Typedef deve sempre vir antes de qualquer programação que envolva procedimentos (protótipo de funções, funções, função main, structs, etc.).

- Sua sintase base é:

```
typedef tipo_de_dado novo_nome;
```

- Onde:

  - **tipo_de_dado** é o tipo de dado que você deseja criar um novo nome.
  - **novo_nome** é o novo nome que você deseja criar para o tipo de dado.

Typedef

- Por exemplo, você pode criar um novo nome MeuInt para o tipo de dado int da seguinte forma:

```
typedef int MeuInt;
```

- Agora, sempre que você utilizar o nome MeuInt no código, ele será interpretado como o tipo de dado int. Por exemplo:

```
MeuInt x = 42;
cout << "x = " << x << endl;
```

Typedef

- O typedef pode ser especialmente útil quando você está lidando com tipos de dados complexos e longos, como ponteiros para funções ou estruturas com muitos campos.

- Com o typedef, você pode criar nomes mais curtos e legíveis para esses tipos de dados complexos, tornando o código mais fácil de ler e entender.

Definindo nomes para estruturas de dados

- Uma vantagem muito grande que typedef nos proporciona é definir um nome para nossa estrutura de dados (struct).

- Graças a isso, somos capazes de auto-referenciar a estrutura, ou seja, colocar um tipo de dado struct dentro de outro struct.

- Podemos definir o nome de uma estrutura de dados (struct) de duas maneiras:

  - Definindo o nome da estrutura e só depois definir a estrutura; ou
  - definir a estrutura ao mesmo tempo que define o nome.

Definindo nomes para estruturas de dados

- Da primeira forma:

```
typedef struct estrutura1 MinhaEstrutura;

   struct estrutura1 {
        int var1;
       float var2;
};
```

- Da segunda forma:

```
typedef struct estrutura1 {
     int var1;
     float var2;
} MinhaEstrutura;
```

typedef1.cpp

```cpp
#include <iostream>
#include <string>
using namespace std;

typedef struct {
    string nome;
    int idade;
} Pessoa;  // Apelido para uma estrutura que representa uma pessoa

int main() {
    setlocale(LC_ALL, "Portuguese");

    Pessoa pessoal;

    cout << "Digite o nome da pessoa: ";
    getline(cin, pessoal.nome);

    cout << "Digite a idade da pessoa: ";
    cin >> pessoal.idade;

    cout << "Nome: " << pessoal.nome << ", Idade: " << pessoal.idade << " anos" << endl;

    return 0;
}
```

typedef2.cpp

```cpp
#include <iostream>
#include <vector>
using namespace std;

typedef vector<int> VetorInteiros;  // Apelido para um vetor de inteiros

int main() {
    setlocale(LC_ALL, "Portuguese");
    int tamanho;

    cout << "Digite o tamanho do vetor: ";
    cin >> tamanho;

    VetorInteiros numeros(tamanho);

    cout << "Digite os números do vetor:" << endl;
    for (int i = 0; i < tamanho; i++) {
        cin >> numeros[i];
    }

    cout << "Números: ";
    for (int num : numeros) {
        cout << num << " ";
    }
    cout << endl;
    return 0;
}
```

typedef3.cpp

```cpp
#include <iostream>
using namespace std;

typedef enum {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO} DiaDaSemana;  // Apelido para uma enumeração
//enum começa em 0

int main() {
    setlocale(LC_ALL, "Portuguese");

    int dia;

    cout << "Digite o número do dia da semana (1 a 7): ";
    cin >> dia;

    DiaDaSemana hoje;
    if (dia >= 1 && dia <= 7) {
        hoje = static_cast<DiaDaSemana>(dia - 1);
        //static_cast é usado para converter esse valor para o tipo DiaDaSemana.
        //A conversão é necessária porque DiaDaSemana é uma enumeração que começa com 0
    } else {
        cout << "Número de dia inválido!" << endl;
        return 1;
    }
```

typedef3.cpp

```
24
25      cout << "Hoje é ";
26      switch (hoje) {
27          case SEGUNDA: cout << "segunda-feira."; break;
28          case TERCA: cout << "terça-feira."; break;
29          case QUARTA: cout << "quarta-feira."; break;
30          case QUINTA: cout << "quinta-feira."; break;
31          case SEXTA: cout << "sexta-feira."; break;
32          case SABADO: cout << "sábado."; break;
33          case DOMINGO: cout << "domingo."; break;
34      }
35      cout << endl;
36
37      return 0;
38  }
```

typedef4.cpp

```cpp
#include <iostream>

using namespace std;

typedef struct data
{
    int dia;
    int mes;
    int ano;
} Data;

typedef struct aniversario
{
    string nome;
    Data nascimento;
} Aniversario;

```

typedef4.cpp

```
18  int main ()
19  {
20      setlocale(LC_ALL, "Portuguese");
21      Aniversario alguem;
22      cout << "Digite o nome de alguem" << endl;
23      getline(cin,alguem.nome);
24      cout << "Digite o dia que esta pessoa nasceu" << endl;
25      cin >> alguem.nascimento.dia;
26      cout << "Digite o mes que esta pessoa nasceu" << endl;
27      cin >> alguem.nascimento.mes;
28      cout << "Digite o ano que esta pessoa nasceu" << endl;
29      cin >> alguem.nascimento.ano;
30      system ("cls");
31
32      cout << alguem.nome << endl;
33      cout << "nasceu em ";
34      cout << alguem.nascimento.dia << "/";
35      cout << alguem.nascimento.mes << "/";
36      cout << alguem.nascimento.ano << endl;
37
38      system ("pause");
39      return 0;
40  }
```

Exercícios

1. Crie uma estrutura chamada pessoa que seja capaz de armazenar o nome, o endereço, o CPF e a idade de 5 pessoas.

Exercícios

2. Suponha que você está desenvolvendo um programa para armazenar informações sobre animais em um zoológico. Crie uma estrutura chamada Animal que deve armazenar as seguintes informações sobre cada animal:

- Nome do animal (string)
- Espécie do animal (string)
- Idade do animal (int)
- Peso do animal em kg (float)

Em seguida, crie um typedef para uma lista de animais chamada ListaAnimais, que deve ser um vetor de 10 elementos do tipo Animal.

Por fim, crie um programa que pede ao usuário para digitar os dados de 10 animais e armazena esses dados em uma variável do tipo ListaAnimais.

Ao final, o programa deve exibir os dados de cada um dos 10 animais na tela.

Exercícios

3. Crie uma estrutura chamada Endereco que armazene os campos rua, cidade e CEP. Em seguida, crie um typedef para Endereco chamado TipoEndereco. Crie um programa que solicite ao usuário que insira informações de endereço e as armazene em uma variável do tipo TipoEndereco.

Exercícios

4. Crie um typedef para um vetor de inteiros chamado VetorInteiros. Em seguida, crie um programa que declare e inicialize um vetor de 5 inteiros usando o typedef e exiba os valores na tela.

Exercícios

5. Crie uma enumeração chamada Cores para representar cores básicas (por exemplo, vermelho, verde, azul). Em seguida, crie um typedef para Cores chamado TipoCor. Crie um programa que permita ao usuário escolher uma cor usando o typedef e exiba a cor escolhida na tela.

Referência desta aula

- Notas de Aula do Prof. Prof. Armando Luiz N. Delgado, baseado em revisão sobre material de Prof.a Carmem Hara e Prof. Wagner Zola.

- VELOSO, P. et Alli. Estruturas de Dados. Ed. Campus, 1986.

- http://www.cplusplus.com/reference/

Obrigado

Rodrigo