Estrutura de Dados

Linguagem C++

Estudando...

Estruturas

- **Estrutura**

Estrutura é uma coleção de variáveis de diferentes tipos de dados sob um único nome.

A palavra-chave **struct** define um tipo de estrutura seguido por um identificador (nome da estrutura).

Dentro da estrutura declara-se um ou mais membros (declarar variáveis dentro de chaves) dessa estrutura.

Definindo estruturas

```
#include <iostream>
#include <stdlib.h>
#include <string>

using namespace std;


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

Inicializando estruturas

Cada campo (variável) da estrutura pode ser acessada usando o operador "." (ponto).

Inicializando e exibindo estruturas

```
int main() {

  Pessoa aluno1;

  cout << "Entre com o nome do aluno: ";
  getline(cin, aluno1.nome);

  cout << "Entre com a idade: ";
  cin >> aluno1.idade;

  cout << "Entre com o salário: ";
  cin >> aluno1.salario;

  cout << endl << endl << "Exibindo informações." << endl;
  cout << "Nome: " << aluno1.nome << endl;
  cout << "Idade: " << aluno1.idade << endl;
  cout << "Salário: " << aluno1.salario << endl << endl;

  system("pause");
  return 0;
}
```

Inicializando e exibindo estruturas

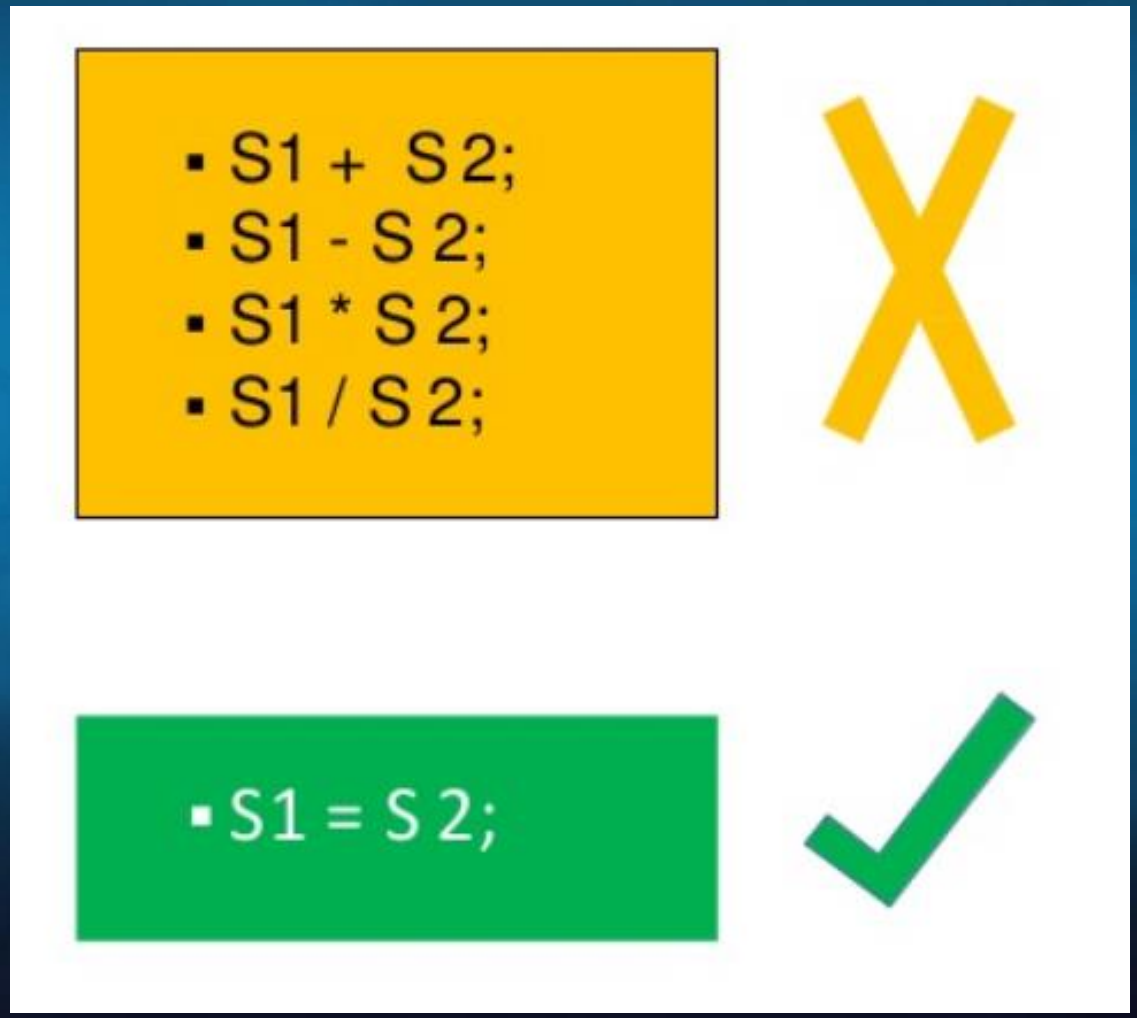
```
int main() {

  Pessoa aluno1 = {"João Pedro", 20, 1000.00};

  cout << endl << endl << "Exibindo informações." << endl;
  cout << "Nome: " << aluno1.nome << endl;
  cout << "Idade: " << aluno1.idade << endl;
  cout << "Salário: " << aluno1.salario << endl << endl;

  system("pause");
  return 0;
}
```

Inicializando e exibindo estruturas



Inicializando e exibindo estruturas

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

int main() {

  Pessoa aluno1 = {"João Pedro", 20, 1000.00};
  Pessoa aluno2;

  aluno2 = aluno1;

  cout << endl << endl << "Exibindo informações." << endl;
  cout << "Nome: " << aluno2.nome << endl;
  cout << "Idade: " << aluno2.idade << endl;
  cout << "Salário: " << aluno2.salario << endl << endl;

  system("pause");
  return 0;
}
```

Exercícios

- Crie um programa que cadastre em forma de estrutura dados de um veículo: marca, modelo, ano de fabricação, ano do modelo, número de portas, chassi e renavam. Em seguida exiba os dados cadastrados.

- Crie um programa que cadastre em forma de estrutura dados de um funcionário: código do empregado, nome, nome do departamento a que está vinculado, ano que foi admitido, ano que foi demitido e salário. Em seguida exiba os dados cadastrados.

Passando estrutura para função

```
void exibirDados(Pessoa paramAluno) {

   cout << endl << endl << "Exibindo informações." << endl;
   cout << "Nome: " << paramAluno.nome << endl;
   cout << "Idade: " << paramAluno.idade << endl;
   cout << "Salário: " << paramAluno.salario << endl << endl;

}
```

Passando estrutura para função

```
void exibirDados(Pessoa paramAluno);

int main() {

  Pessoa aluno1;

  cout << "Entre com o nome do aluno: ";
  getline(cin, aluno1.nome);

  cout << "Entre com a idade: ";
  cin >> aluno1.idade;

  cout << "Entre com o salário: ";
  cin >> aluno1.salario;

  exibirDados(aluno1);

  system("pause");
  return 0;
}
```

Retornando estrutura de função

```
Pessoa obterDados();
void exibirDados(Pessoa paramAluno);

int main() {

  Pessoa aluno1;

  aluno1 = obterDados();

  exibirDados(aluno1);

  system("pause");
  return 0;
}
```

Retornando estrutura de função

```
Pessoa obterDados() {
  Pessoa aluno;

  cout << "Entre com o nome do aluno: ";
  getline(cin, aluno.nome);

  cout << "Entre com a idade: ";
  cin >> aluno.idade;

  cout << "Entre com o salário: ";
  cin >> aluno.salario;

  return aluno;
}

void exibirDados(Pessoa paramAluno) {

  cout << endl << endl << "Exibindo informações." << endl;
  cout << "Nome: " << paramAluno.nome << endl;
  cout << "Idade: " << paramAluno.idade << endl;
  cout << "Salário: " << paramAluno.salario << endl << endl;
}
```

Exercícios

- Altere, os exercícios anteriores de estrutura, para que utilizem funções na obtenção e exibição dos dados da estrutura.

Exercícios

- Implemente um estrutura de ContaBancaria, com os seguintes dados do cliente: código do cliente, nome, idade, telefone, número da conta e saldo da conta. Em seguida crie as seguintes funções:
  - Iniciar uma conta com um número e saldo inicial
  - Depositar um valor
  - Sacar um valor
  - Imprimir o saldo

Exercícios

- Defina um registro empregado para armazenar os dados (nome, sobrenome, data de nascimento, CPF, data de admissão, salário) de um colaborador de sua empresa.

- Defina um vetor de empregados para armazenar 3 colaboradores de sua empresa.

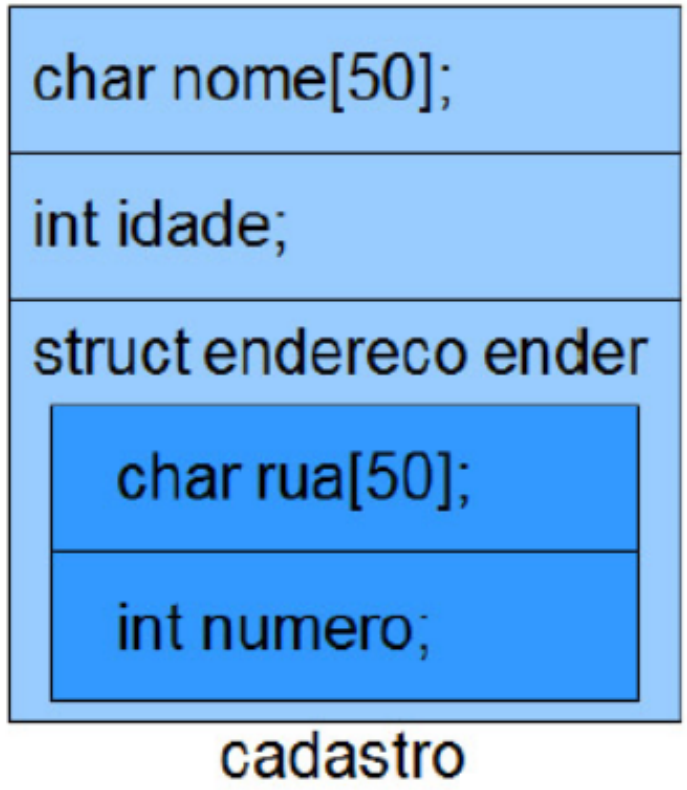
Estruturas Aninhadas

- Podemos declarar uma estrutura que possua uma variável do tipo de outra estrutura previamente definida. A uma estrutura que contenha outra estrutura dentro dela damos o nome de estruturas aninhadas.

Estruturas Aninhadas

Exemplo: struct aninhada.

```
1 struct endereco{
2    char rua[50]
3    int numero;
4 };
5 struct cadastro{
6    char nome[50];
7    int idade;
8    struct endereco
         ender;
9 };
```



Estruturas Aninhadas

```
struct Endereco {
  string rua;
  int nro;
};
struct Pessoa
{
  string nome;
  int idade;
  float salario;
  Endereco endereco;
};
int main() {

  Pessoa aluno1 = { "João Pedro", 20, 1000.00, {"Amapa", 35} };

  cout << endl << endl << "Exibindo informações." << endl;
  cout << "Nome: " << aluno1.nome << endl;
  cout << "Idade: " << aluno1.idade << endl;
  cout << "Salário: " << aluno1.salario << endl;
  cout << "Endereço: " << aluno1.endereco.rua << ", " <<  aluno1.endereco.nro << endl << endl;

  system("pause");
  return 0;
}
```

Exercícios

- **Crie a estrutura abaixo**

Universidade Estadual de Maringá - Centro de Tecnologia
Curso de : Análise de Sistemas Código do Curso: 0037
Aluno: Victor Alexandre Costa Matricula: 007043 Status: Regular

Histórico

Disciplina (codigo)	Professor - Código	Nota	Faltas	Situação
Análise de sistemas (AN001)	Roberto Carlos - 001	7,5	7	Aprovado
Matemática (MA002)	Jandira - 002	8.0	4	Aprovado
Inglês (IN101)	Junior Villas - 003	4.5	0	Reprovado

Vetor de Estrutura

```
#include <iostream>
#include <stdlib.h>
#include <string>

using namespace std;

struct Produto
{
  int CodProduto;
  string descricao;

};
```

Vetor de Estrutura

```
int main() {

   Produto Prod[2];

   cout << "Entre com código do produto 1: ";
   cin >> Prod[0].CodProduto;

   cout << "Entre com a descrição do produto 1: ";
   getline(cin, Prod[0].descricao);

   cout << "Entre com código do produto 2: ";
   cin >> Prod[1].CodProduto;

   cout << "Entre com a descrição do produto 2: ";
   getline(cin, Prod[1].descricao);
```

Vetor de Estrutura

```
  cout << endl << endl << "Exibindo informações." << endl;
  cout << "============ Produto 1 ============" << endl;
  cout << "Código: " << Prod[0].CodProduto << endl;
  cout << "Descrição: " << Prod[0].descricao << endl << endl;

  cout << "============ Produto 2 ============" << endl;
  cout << "Código: " << Prod[1].CodProduto << endl;
  cout << "Descrição: " << Prod[1].descricao << endl << endl;

  system("pause");
  return 0;
}
```

Vetor

IMPORTANTE:

- **VETOR** e **MATRIZ**, sempre são passados como **REFERÊNCIA** para uma função.

```
struct Produto
{
  int CodProduto;
  string descricao;

};

void exibirInformacoes(Produto Prod[]);
```

Vetor

IMPORTANTE:

- **VETOR** e **MATRIZ**, sempre são passados como **REFERÊNCIA** para uma função.

```
int main() {

  Produto Prod[2];

  cout << "Entre com código do produto 1: ";
  cin >> Prod[0].CodProduto;

  cout << "Entre com a descrição do produto 1: ";
  cin.ignore();
  getline(cin, Prod[0].descricao);

  cout << "Entre com código do produto 2: ";
  cin >> Prod[1].CodProduto;

  cout << "Entre com a descrição do produto 2: ";
  cin.ignore();
  getline(cin, Prod[1].descricao);

  exibirInformacoes(Prod);

  system("pause");
  return 0;
}
```

Vetor

IMPORTANTE:

- **VETOR** e **MATRIZ**, sempre são passados como **REFERÊNCIA** para uma função.
  - Formato **MATRIZ**.

```
void exibirInformacoes(Produto Prod[]) {

   cout << endl << endl << "Exibindo informações." << endl;
   cout << "========== Produto 1 ============" << endl;
   cout << "Código: " << Prod[0].CodProduto << endl;
   cout << "Descrição: " << Prod[0].descricao << endl << endl;

   cout << "========== Produto 2 ============" << endl;
   cout << "Código: " << Prod[1].CodProduto << endl;
   cout << "Descrição: " << Prod[1].descricao << endl << endl;
}
```

Vetor

IMPORTANTE:

- **VETOR** e **MATRIZ**, sempre são passados como **REFERÊNCIA** para uma função.
  - Formato **PONTEIRO**.

```
void exibirInformacoes(Produto *Prod) {

   cout << endl << endl << "Exibindo informações." << endl;
   cout << "========== Produto 1 ============" << endl;
   cout << "Código: " << (Prod + 0)->CodProduto << endl;
   cout << "Descrição: " << (Prod + 0)->descricao << endl << endl;

   cout << "========== Produto 2 ============" << endl;
   cout << "Código: " << (Prod + 1)->CodProduto << endl;
   cout << "Descrição: " << (Prod + 1)->descricao << endl << endl;
}
```

Vetor

IMPORTANTE:

- **VETOR** e **MATRIZ**, sempre são passados como **REFERÊNCIA** para uma função.
  - Formato **PONTEIRO**.

```
void exibirInformacoes(Produto *Prod) {

   cout << endl << endl << "Exibindo informações." << endl;
   cout << "========== Produto 1 ============" << endl;
   cout << "Código: " << (*(Prod + 0)).CodProduto << endl;
   cout << "Descrição: " << (*(Prod + 0)).descricao << endl << endl;

   cout << "========== Produto 2 ============" << endl;
   cout << "Código: " << (*(Prod + 1)).CodProduto << endl;
   cout << "Descrição: " << (*(Prod + 1)).descricao << endl << endl;
}
```

Exercícios

- **Crie a estrutura abaixo**

Livros & Cia. **Ped. Num**: _____

Nome cliente: _____
End. :_____
End. Entrega: _____

Livros solicitados

Cód.	Quant.	Preço unit.	Total
....			
		Tot. Ped: R$	

Vendedor: _____

Livros e Cia. **Ficha de livro**

Código: _____
Título: _____
Autores: _____
Preço R$: _____

Exercícios

3) Fazer um programa que cria uma estrutura livro, que contém os elementos título, ano de edição, número de páginas e preço. Criar uma variável desta estrutura que é um vetor de 5 elementos. Ler os valores para a estrutura e imprimir a média do número de páginas do livros.

Exercícios

1) Criar uma estrutura chamada DadosAluno, que armazena a média e idade de um aluno. Na função *main*: criar uma variável que é uma estrutura DadosAluno; ler a média e a idade de um aluno e armazenar na variável criada; exibir na tela a média e a idade do aluno.

2) Considerando o exercício 1, criar uma variável que é um **vetor** da estrutura DadosAluno. O programa deve obter a média e a idade de 10 alunos. Depois, estes dados devem ser exibidos.

Pensamento

"O pensamento lógico pode levar você de A a B, mas a imaginação te leva a qualquer parte do Universo.“

(Albert Einstein)

FIM

Prof. Me Ricardo Luis Balieiro