Programação de Computadores I

Struct (estrutura)

struct

- Em linguagem C, uma struct (ou estrutura) é um tipo de estrutura de dados composto por múltiplos dados que não precisam ser do mesmo tipo.

struct

- struct em C implementa o que chamamos de registro em algoritmo.
- Em Algoritmos um registro é composto por vários campos, onde cada campo é um tipo.

Declaração

- Declaramos um struct da seguinte forma:

  struct nome-do-registro { declaração dos campos };
- Isto declara um tipo e para usarmos é necessário declarar instâncias desse tipo:

  struct nome-do-registro nome-da-variável;

Exemplo

```
//definindo a estrutura
struct Cliente{
	long int cpf;
	char nome[50];
	float salario;
};

//declarando variável
struct Cliente cliente1;
```

Acessando os Dados

- Para acessar os campos, itens de dados, de um registro usa-se:

  nome-da-variável.nome-do-campo

- Exemplo:

  printf(“Informe o Cpf do Cliente:”);

  scanf(“%ld”, &cliente1.cpf);

No programa

```c
#include <stdio.h>
#include <string.h>

// 1. Definindo a estrutura
struct Cliente {
    long int cpf;
    char nome[50];
    float salario;
};

int main() {
    // 2. Declarando uma variável do tipo struct Cliente
    struct Cliente cliente1;

    // 3. Atribuindo valores usando o operador ponto (.)
    strcpy(cliente1.nome, "Carlos Silva");
    cliente1.cpf = 123456678900;
    cliente1.salario = 2500.00;

    // 4. Acessando os dados
    printf("Nome: %s\n", cliente1.nome);
    printf("CPF:  %ld\n", cliente1.cpf);
    printf("Salário: R$ %.2f\n", cliente1.salario);

    return 0;
}
```

typedef-Definição de Tipo

- Em linguagem C, o typedef (abreviação de *type definition*) é uma palavra-chave utilizada para criar um pseudônimo (apelido) para um tipo de dado já existente.
- Ele não introduz um tipo de dado verdadeiramente novo; apenas define um nome alternativo mais conveniente, legível ou curto para um tipo que já foi declarado.

Sintaxe Básica

A estrutura do comando segue a mesma lógica da declaração de uma variável comum, mas adiciona o typedef no início:

```
typedef tipo_original novo_nome;
```

Exemplos Práticos de Uso

- Cuidado! Uso 1.

```
struct Ponto {
    int x;
    int y;
};
typedef struct Ponto p1;
```

- Aqui o nome do tipo é p1.

Exemplos Práticos de Uso

- Uso 2.

```
typedef struct {
    int x;
    int y;
} Ponto;
Ponto p1;
```

- Aqui o nome do tipo é Ponto e p1 é a variável.

Exemplos Práticos de Uso

- Uso 3: Melhorando a Legibilidade do Código

  Você pode usar o typedef para dar um significado semântico a tipos primitivos, tornando o código auto documentado.

  ```
  typedef float Preco;

  typedef int Ano;


  Preco valorProduto = 29.90;

  Ano fabricacao = 2026;
  ```

Exemplos Práticos de Uso

- Podemos criar um vetor de registro, por exemplo:

```
struct Cliente clientes[100];
```

- Aqui reservamos espaço para armazenar dados de até 100 clientes. Acessamos:

```
clientes[1].cpf=12345678900;
```

Exercício

- Construa um programa que leia rgm, o nome e 4 notas de até 10 acadêmicos, um a um e perguntando se quero continuar informando. Os dados de cada acadêmico têm os tipos :
  - rgm é um inteiro longo,
  - Nome é uma string para 50 caracteres; e
  - As 4 notas são float.
- Depois de ler pergunte se quero imprimir todos.