

# Struct (Estrutura)

## Agenda – Parte 1

1 - CONCEITO

2 - Acessando os elementos (campos) da estrutura

3 - Inicialização dos elementos (campos) da estrutura

4 - Estrutura em Estruturas

5 – Vetores de Estruturas

6 – Estruturas e Funções

7 – Referências

## Linguagem de Programação Estruturada – 2016.1

### 1. CONCEITO

Em linguagem **C**, uma estrutura é um conjunto de variáveis de tipos distintos ou não, agrupadas sob um único nome. As variáveis que compõem a estrutura são chamadas membros, campos ou elementos.

Imagine que você precisa gravar vários dados de um cliente (código, nome, endereço, telefone, CPF, e-mail, limite de credito).

```
struct TipoCliente {  
    int codigo;  
    char nome[50];  
    char endereco[100];  
    char telefone[12];  
    char CPF[11];  
    char eMail[40];  
    float limiteDeCredito;  
};
```

Observe que dentro da estrutura temos algumas variáveis de tipos primitivos (int, vetores de char e float).

## Linguagem de Programação Estruturada – 2016.1

### 1. CONCEITO

Para se declarar uma variável efetivamente, é preciso um enunciado:

```
struct TipoCliente cliente;
```

Assim, passamos a ter uma variável chamada **cliente** que é do tipo **TipoCliente**.

Podemos inclusive declarar mais de uma variável deste tipo:

```
struct TipoCliente cliente, clienteEspecial, clienteInativo;
```

## 1. CONCEITO

A declaração das variáveis pode ser feita junto com a criação do tipo. No caso acima teríamos:

Outro exemplo:

```
struct TipoCliente {
    int codigo;
    char nome[50];
    char endereco[100];
    char telefone[12];
    char CPF[11];
    char eMail[40];
    float limiteDeCredito;
} cliente, clienteEspecial, clienteInativo;
```

```
int main (){
    struct aluno
    {
        char nome[80];
        int matricula;
    }joana, marilena, marcelo;
}
```

## 1. CONCEITO

A declaração das variáveis pode ser feita junto com a criação do **tipo**. No caso acima teríamos:

Outro exemplo:

```
int main (){

    struct aluno
    {
        char nome[80];
        int matricula;
    }joana, marilena, marcelo;
}
```

NOTA. Observe que a linguagem permite ao programador a flexibilidade de declarar uma **struct** no cabeçalho do programa, ou seja, antes da função **main**. Como podem ser declaradas dentro da função **main**.

## Linguagem de Programação Estruturada – 2016.1

### 1. CONCEITO

Temos, então, como forma geral da definição de uma estrutura:

```
struct NomeDoNovoTipo {  
    tipo nomeDaVariável;  
    tipo nomeDaVariável;  
    ...  
    tipo nomeDaVariável;  
} variavelUm, variavelDois, ... , variavelN;
```

## 2. Acessando os elementos (campos) da estrutura

O acesso a um **campo da estrutura** se dá por meio do **nome da variável**, um **ponto** e o **nome do campo**.

Exemplo:

```
struct TipoCliente {
    int codigo;
    char nome[50];
    char endereco[100];
    char telefone[12];
    char CPF[11];
    char eMail[40];
    float limiteDeCredito;
} cliente, clienteEspecial, clienteInativo;
```

- cliente.codigo
- clienteEspecial.limiteDeCredito
- gets(clienteInativo.nome);

De um modo geral:

nomeDaVariavelDoTipoEstrutura.nomeDoCampo



## 2. Acessando os elementos (campos) da estrutura

O acesso a um **campo da estrutura** se dá por meio do **nome da variável**, um **ponto** e o **nome do campo**.

Outro Exemplo:

```
struct funcionario{
char nome[50];
double salario;
} x, y, z;
```

```
// estrutura x
strcpy(x.nome, "Amanda da Silva");
x.salario=370.00
```

```
//estrutura y
gets(y.nome);
scanf("%lf", &y.salario);
```

```
//estrutura z
z=x; //atribuição de todos os campos de x para z;
```

*OBS.* Só é possível atribuir o conteúdo de uma estrutura a outra estrutura, se ambas forem do mesmo tipo (int, float, char, ...) e com a ordem de declaração de campos iguais.

## 3. Inicialização dos elementos (campos) da estrutura

A inicialização dos campos de uma estruturas é semelhante à inicialização de um array. Os valores para cada um dos membros são escritos entre chaves e separados por vírgula, na ordem em que foram declarados.

### Exemplo 1

```
struct funcionario
{
    char nome[50];
    double salario;
} x, y, z;
```

```
struct funcionario x = {"Amanda da Silva", 370.00};
struct funcionario y = {"Marcelo de Souza Junior", 450.00};
struct funcionario z = {"Roberto dos Santos", 1530.00};
```

### 3. Inicialização dos elementos (campos) da estrutura

A inicialização dos campos de uma estruturas é semelhante à inicialização de um array. Os valores para cada um dos membros são escritos entre chaves e separados por vírgula, na ordem em que foram declarados.

#### Exemplo 2

```
struct data{  
    int dia;  
    int mes;  
    int ano;  
}hoje;
```

```
struct data hoje={13,11,2006};
```

## 4. Estrutura em Estruturas

**Considere o exemplo abaixo:**

```
struct data{  
    int dia;  
    int mes;  
    int ano;  
};nascimento
```

**Para atribuir valores aos campos da estrutura (dia, mes, ano), usamos:**

```
scanf("%d", &nascimento.dia); ou nascimento.dia=27;  
scanf("%d", &nascimento.mes); ou nascimento.mes=08;  
scanf("%d", &nascimento.ano); ou nascimento.ano=1979;
```

## 4. Estrutura em Estruturas

Agora considere que temos a **struct** aluno e que precise da informação da data de nascimento do aluno.

Como podemos fazer?

- Podemos criar os campos referentes ao dia, mês e ano dentro da estrutura.
- ou podemos declarar uma **struct** data como campo da **struct** aluno, tornando o código mais robusto de fácil manutenção.

## 4. Estrutura em Estruturas

### Exemplo:

```
struct data{  
    int dia;  
    int mes;  
    int ano;  
};
```

```
struct aluno{  
    char nome[80];  
    int matricula;  
    struct data nascimento;  
}fulano;
```

```
gets(fulano.nome);  
scanf("%d", &fulano.matricula);
```

Para atribuir valores a **struct data**, deve ser referenciado a estrutura que ela esta inserida seguida do nome do campo:

```
scanf("%d", &fulano.nascimento.dia);  
scanf("%d", &fulano.nascimento.mes);  
scanf("%d", &fulano.nascimento.ano);
```

## 5. Vetores de Estruturas

Vetores de estruturas são vetores onde os elementos (ou índices) apontam para dados do tipo ***struct***.

- O acesso a um vetor de estruturas é feito colocando-se o índice entre colchetes, logo após o nome do vetor seguido pelo operador ponto e o nome do campo da estrutura:

```
struct aluno
{
    char nome[80];
    int matricula;
}faculdade[10];
```

- O acesso pode ser individual, apenas referenciando o índice...

```
faculdade[0].nome="Cesar de Sousa";
```

- ... ou através de um controle de fluxo:

```
for(i=0;i<10;i++) faculdade[i].nome="cesar de sousa";
```

# Linguagem de Programação Estruturada – 2016.1

## 5. Vetores de Estruturas

```
#include <stdio.h>
#include <stdlib.h>
#define TAM_VET 4

// declaracao do modelo data
struct data{
    int dia, mes, ano;
};

// declaracao do modelo funcionario
struct aluno{
    char nome[50];
    double mediaFinal;
    struct data nascimento;
};

int main(){
    int i;
    // declaracao e inicialização do vetor de estruturas vet
    struct aluno faculdade[TAM_VET]={{"Adriana Martins", 7.5, {10,5,1982} },
                                        {"Luciano Araujo de Lima", 8, {23,8,1983} },
                                        {"Marcos Paulo da Silva", 9.2, {14,1,1981} },
                                        {"Jonathas Martins dos Santos", 10, {2,10,1983} }
    };
    puts("Imprimindo os dados.\n");
    for(i=0; i<4; i++){
        printf("%s", faculdade[i].nome);
        printf("Media Final: %.2f", faculdade[i].mediaFinal);
        printf("\n%d/%d/%d\n\n", faculdade[i].nascimento.dia,
                                faculdade[i].nascimento.mes,
                                faculdade[i].nascimento.ano );
    }
}
```



## 6. Estruturas e Funções

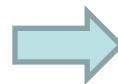
Suponha as seguintes declarações de estruturas:

```
struct data{
int dia, mes, ano;
};
struct aluno{
char nome[50];
double mediaFinal;
struct data nascimento;
}fulano;
```

- Passagem de alguns elementos da estrutura para funções:

**Sintaxe:** nome\_da\_função(tipo\_da\_estrutura.nome\_do\_campo);

```
double altera_mediaFinal (double media){
double novaMedia;
novaMedia = media + 1.0;
return novaMedia;
}
```



```
altera_mediaFinal(fulano.mediaFinal);
```

## 6. Estruturas e Funções

- Passagem de toda a estrutura para funções:

A passagem de toda uma estrutura para uma função pode ser feita por valor. Na definição da função o parâmetro declarado deve ser do mesmo tipo da estrutura que está sendo passada.

Exemplo:

```
void imprime(struct aluno s) // o parâmetro é do tipo struct aluno
{
    puts(s.nome);
    printf("Media Final: %.2f", s.mediaFinal);
    printf("\nData de nascimento: %d/%d/%d\n\n", s.nascimento.dia,
    s.nascimento.mes, s.nascimento.ano );
}
```

## 7. REFERÊNCIAS

- Márcio Alexandre Marques. Algoritmos - Lógica para Desenvolvimento de Programação de Computadores. 1ª Ed. Editora Érica, 2010.
- Sandra Rita. TREINAMENTO EM LOGICA DE PROGRAMAÇÃO, Digerati Books, 1 ed. 2009.
- SIMÃO, DANIEL HAIASHIDA; REIS, WELLINGTON JOSÉ DOS. LOGICA DE PROGRAMAÇÃO. São Paulo : EDITORA VIENA, 2015. 176p.
- Souza, Marco Antonio Furlan de *et. all*, Algoritmos e Lógica de Programação. 2 ed. São Paulo : Nobel, 2011.