Struct em C

O que é?

- Uma estrutura (struct) é uma coleção de campos que podem ser referenciados pelo mesmo nome. A estrutura permite que informações relacionadas mantenham-se juntas.
- Uma struct é uma variável especial que contém diversas outras variáveis normalmente de tipos diferentes.

O que é?

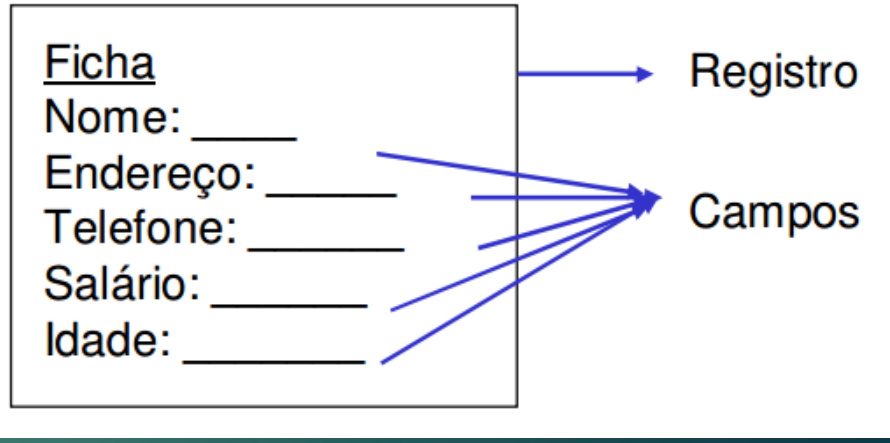
- As variáveis internas contidas pela struct são denominadas membros da struct.

- Structs são muito usadas quando temos elementos em nossos programas que precisam e fazem uso de vários tipos de variáveis e características.
- Usando *struct*, podemos trabalhar com vários tipos de informações de uma maneira mais fácil, rápida e organizada, uma vez que não temos que nos preocupar em declarar e decorar o nome de cada elemento da struct.

Struct: Sintaxe

A sintaxe é:

```
struct <identificador> {
   <tipo>   campo_um   ;
   <tipo>   campo_dois ;
};
```




```
struct endereco{
    char nome[30];
    char rua[40];
    long int cep;
};
```

```
struct cliente {
    char nome[30];
    char rua[50];
    int idade;
} voce;
```

```
/*Criando a struct  */
struct ficha_de_aluno
{
  char nome[50];
  char disciplina[30];
  float nota_prova1;
  float nota_prova2;
};
```

Usando uma Struct

```c
#include <stdio.h>
#include <conio.h>
int main(void)
{
  /*Criando a struct  */
  struct ficha_de_aluno
  {
    char nome[50];
    char disciplina[30];
    float nota_prova1;
    float nota_prova2;
  };
   /*Criando a variável aluno que será do tipo struct ficha_de_aluno       */
  struct ficha_de_aluno aluno;

  gets(aluno.nome);
  aluno.nota_prova1 = 10;
  aluno.nota_prova2=9;
  return 0;
}
```

```c
/* acessando os campos de uma estrutura */
#include <stdio.h>

/* criando um novo tipo de dado "produto" */
struct produto
  {
    int codigo;
    char nome[50];
    int quantidade;
    float valor_compra;
    float valor_venda;
  };

int main()
  {
    struct produto item; /* declarando uma variável "item" do tipo "struct produto" */

    printf("Preenchendo a variável \"item\"\n");
    printf("Item.............:");
    fgets(item.nome,50,stdin);
    printf("Código..........:");
    scanf("%d",&item.codigo);
    printf("Quantidade......:");
    scanf("%d",&item.quantidade);
    printf("Valor de compra.:");
    scanf("%f",&item.valor_compra);
    printf("Valor de revenda:");
    scanf("%f",&item.valor_venda);
    printf("\n");
    printf("Exibindo os dados\n");
    printf("Código..........:%d\n",item.codigo);
    printf("Item.............:%s",item.nome);
    printf("Quantidade......:%d\n",item.quantidade);
    printf("Valor de compra.:%.2f\n",item.valor_compra);
    printf("Valor de revenda:%.2f\n",item.valor_venda);

    return(0);
  }
```

```c
struct produto
  {
    int codigo;
    char nome[50];
    int quantidade;
    float valor_compra;
    float valor_venda;
  };
```

```c
    printf("Quantidade......:");
    scanf("%d",&item.quantidade);
    printf("Valor de compra.:");
    scanf("%f",&item.valor_compra);
    printf("Valor de revenda:");
    scanf("%f",&item.valor_venda);
    printf("\n");
```

```c
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
     int i;
     struct  {                //Criando a struct
       char nome[50];
       float nota_prova1;
       float nota_prova2;
     } ficha[3];

     gets(ficha[0].nome);              // entrada de dados
     ficha[0].nota_prova1 = 10;
     ficha[0].nota_prova2=9;

     gets(ficha[1].nome);
     ficha[1].nota_prova1 = 9;
     ficha[1].nota_prova2=9;

     gets(ficha[2].nome);
     ficha[2].nota_prova1 = 7;
     ficha[2].nota_prova2=7;

     for(i=0; i<3;i++)   printf("\n%s %f %f \n", ficha[i].nome, ficha[i].nota_prova1, ficha[i].nota_prova2);
     return 0;
}
```

Ficha[3]

Typedef

- A palavra reservada typedef nada mais é do que um atalho em C para que possamos nos referir a um determinado tipo existente com nomes sinônimos.

- Por exemplo, com o typedef, em vez de termos que nos referir como 'struct Aluno', poderíamos usar somente 'Aluno' para criar structs daquele tipo.
- Em vez de escrever sempre 'struct Funcionario', poderíamos escrever apenas 'Funcionario' e então declarar várias structs do tipo 'Funcionario'.

Typedef: Sintaxe

- A sintaxe do typedef é bem simples:

typedef tipo_existente nome_que_voce_escolheu;

Typedef

```c
#include <stdio.h>

int main(void)
{
    typedef int meuInteiro;
    typedef char String[20];

    meuInteiro numero = 1;
    String nome;
    scanf("%[^\n]s", nome);

    printf("A variavel do tipo 'meuInteiro' eh um int e vale %d\n", n
umero);
    printf("Ja a variavel 'nome' eh uma String e armazena \"%s\"\n",
nome);

    return 0;
}
```

Typedef e Struct

- Outro exemplo de sintaxe de typedef com estruturas é a seguinte, onde não precisamos criar o nome da struct, apenas do sinônimo:

```
typedef struct
{
   //código da  struct

} Alunos;
```

```c
#include <stdio.h>
#define DIM 5
int main(void)
{
    typedef struct
    {
        char nome[30];
        float matematica, fisica, media;
    }Alunos;

    Alunos alunos[DIM];
    int count;

    for(count = 0 ; count < DIM ; count++)
    {
        printf("\nNome do aluno %d: ", count+1);
        gets(alunos[count].nome);

        printf("Nota de matematica: ");
        scanf("%f", &alunos[count].matematica);

        printf("Nota de fisica: ");
        scanf("%f", &alunos[count].fisica);

        alunos[count].media = (alunos[count].matematica + alunos[count].fisica)/2;
    }

    printf("\nExibindo nomes e medias:\n");

    for(count = 0 ; count < DIM ; count++)
    {
        printf("nAluno %d\n", count+1);
        printf("Nome: %s\n",alunos[count].nome);
        printf("Media: %.2f\n", alunos[count].media);
    }

    return 0;
}
```

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

typedef struct Books
{
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
} Book;

int main( )
{
   Book book;

   strcpy( book.title, "C Programming");
   strcpy( book.author, "Nuha Ali");
   strcpy( book.subject, "C Programming Tutorial");
   book.book_id = 6495407;

   printf( "Book title : %s\n", book.title);
   printf( "Book author : %s\n", book.author);
   printf( "Book subject : %s\n", book.subject);
   printf( "Book book_id : %d\n", book.book_id);

   return 0;
}
```