

Programação de Computadores IV

Tipos compostos e Estruturas

Slides cedidos por Prof. Bruno
Augusto Dorta Marques

Tipos e estruturas

- Além dos tipos primitivos da linguagens (int, float, char, etc), podemos definir tipos compostos.
- Uma estrutura é um tipo composto definido pelo usuário.
- Uma estrutura permite combinar dados com diferentes tipos.

Estruturas em C

- Para definir uma estrutura em c, usamos a palavra reservada **typedef struct**:

```
typedef struct <NomeDaStruct> {  
    <membro>;  
    <membro>;  
    ...  
    <membro>;  
} <NomeDoTypedef>;
```

- <NomeDoTypedef> é um identificador do tipo criado pelo usuário
- <NomeDaStruct> é o nome dado a sua struct
- <membro> é a declaração de uma variável que faz parte da struct

Exemplo:

- Queremos criar uma estrutura para armazenar informações de um Livro
- Essa estrutura deve conter:
 - Nome do livro
 - Autor
 - Categoria
 - Identificador do Livro

Exemplo (em C)

```
typedef struct Livros{  
    char nome[100];  
    char autor[100];  
    char categoria[100];  
    int id;  
} Livro;
```

- A declaração da struct deve ser feita no início do código fonte do programa. Antes de todas as funções.
- A declaração da struct deve ser feita **fora** da `main( )`

Exemplo (em C)

```
typedef struct Livros{
    char nome[100];
    char autor[100];
    char categoria[100];
    int id;
} livro;

int main(void){
    // declarando duas variáveis do tipo livro
    livro livro1;
    livro livro2;

    ...
    return 0;
}
```

Exemplo (em C)

```
typedef struct Livros{
    char nome[100];
    char autor[100];
    char categoria[100];
    int id;
} livro;

int main(void ){
// declarando duas variáveis do tipo livro
    livro livro1;
    livro livro2;

    strcpy(livro1.nome, "Harry Potter");
    strcpy(livro1.autor, "J. K. Rowling");
    strcpy(livro1.categoria, "Aventura");
    livro1.id = 1;
```

Exemplo (em C)

```
...  
int main(){  
    // declarando duas variáveis do tipo livro  
    livro livro1;  
    livro livro2;  
  
    strcpy(livro1.nome, "Harry Potter");  
    strcpy(livro1.autor, "J. K. Rowling");  
    strcpy(livro1.categoria, "Aventura");  
    livro1.id = 1;  
  
    strcpy(livro2.nome , "O Hobbit");  
    strcpy(livro2.autor , "J. R. R. Tolkien");  
    strcpy(livro2.categoria , "Fantasia");  
    livro2.id = 2;  
    return 0;  
}
```

Exemplo (em C)

...

```
printf( "Livro 1 titulo : %s\n", livro1.titulo);  
printf( "Livro 1 autor : %s\n", livro1.autor);  
printf( "Livro 1 categoria : %s\n", livro1.categoria);  
printf( "Livro 1 id : %d\n", livro1.id);
```

```
printf( "Livro 2 titulo : %s\n", livro2.titulo);  
printf( "Livro 2 autor : %s\n", livro2.autor);  
printf( "Livro 2 categoria : %s\n", livro2.categoria);  
printf( "Livro 2 id : %d\n", livro2.id);
```

```
Return 0;
```

```
}
```

Exemplo 2:

```
typedef struct Livros{  
    char nome[100];  
    char autor[100];  
    char categoria[100];  
    int id;  
} livro;
```

```
int main(){  
  
    livro acervo[2];  
  
    strcpy(acervo[0].nome, "Harry Potter");  
    strcpy(acervo[0].autor, "J. K. Rowling");  
    strcpy(acervo[0].categoria, "Aventura");  
    acervo[0].id = 1;
```

...

Exemplo 2:

...

```
livro acervo[2];  
livro copia;
```

```
strcpy(acervo[0].nome, "Harry Potter");  
strcpy(acervo[0].autor, "J. K. Rowling");  
strcpy(acervo[0].categoria, "Aventura");  
acervo[0].id = 1;
```

```
strcpy(acervo[1].nome = "O Hobbit");  
strcpy(acervo[1].autor = "J. R. R. Tolkien");  
strcpy(acervo[1].assunto = "Fantasia");  
strcpy(acervo[1].id = 2;
```

```
copia = acervo[1]
```

...

Variável copia vai receber os valores do livro armazenado em acervo na posição 1

Exercício

Crie um programa para armazenar informações de 10 livros diferentes.

Cada livro possui as seguintes informações:

Título

Autor

Categoria

Identificador

O Programa deve pedir ao usuário as informações dos 10 livros e ao final imprimir todas as informações.