

Tecnologia em Análise e Desenvolvimento de Sistemas - TADS

Estrutura de Dados I

Atividades Pedagógicas Não Presenciais – APNP 2020

Prof. Luciano Vargas Gonçalves

E-mail: luciano.goncalves@riogrande.ifrs.edu.br



Estrutura da Dados

- **Aula 3 – Structs em Programação em C**

Structs

- Estrutura estática de Dados em C
 - Uma estrutura é um grupo de itens no qual cada item é identificado por um identificador próprio, sendo cada um deles conhecido como um **membro** da estrutura.
 - Estrutura é uma versão resumida de uma classe, cada membro pode ser interpretado como um atributo.
 - Os vários atributos formam um Struct;

Structs

- Estrutura de Dados em C
 - Exemplo estrutura para armazenar os dados de uma pessoa;

```
struct nome_da_estrutura{  
    tipo nome_parametro;  
    tipo nome_parametro;  
};
```

Definição

```
struct pessoa{  
    int cod;  
    char nome [10];  
    char sobrenome [20];  
    int idade;  
    char telefone [10];  
};
```

Exemplo

Structs

- Estrutura de Dados em C

- Declarar

```
struct pessoa p1;
```

Tipo de dados

- Instanciar uma estrutura com dados conhecidos

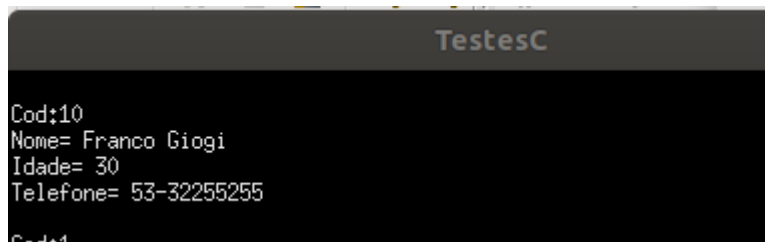
```
p1.cod = 10;  
strcpy(p1.nome, "Franco");  
strcpy(p1.sobrenome, "Giogi");  
p1.idade = 30;  
strcpy(p1.telefone, "53-32255255");
```

—

Structs

- Estrutura de Dados em C
 - Acessar um atributo operador “.”;
 - *Struct.atributo;*
 - Imprimir os dados de uma Struct;

```
printf("\nCod:%d\nNome= %s %s\nIdade= %d\nTelefone= %s\n",  
      p1.cod,p1.nome,p1.sobrenome,p1.idade,p1.telefone);
```



```
TestesC  
Cod:10  
Nome= Franco Giogi  
Idade= 30  
Telefone= 53-32255255  
Cod:1
```

Structs

- Estrutura de Dados em C
 - Declarar e instanciar uma estrutura, dados conhecidos, usar chaves { };

```
struct pessoa p2 = {1, "Alex", "Silva Souza", 34, "53-98345674"};
```

Structs

- Estrutura de Dados em C
 - Estrutura usando ponteiro ***p**. Declarar o ponteiro;

```
//usando ponteiro para struct  
struct pessoa *p3;
```

- **MALLOC = Alocar memória para armazenar os dados;**

```
//aloca memória e copia o endereço para ponteiro  
p3 = (struct pessoa *)malloc(sizeof(struct pessoa));
```

Cast (tipar)

Alocação de Memória

Structs

- Estrutura usando ponteiro **p*
 - Atribuir valores a estrutura através do ponteiro
 - Operador “->”

```
p3->cod = 2;  
printf("Informe o nome:");  
scanf("%s", p3->nome);  
printf("Informe o sobrenome:");  
scanf("%s", p3->sobrenome);  
printf("Informe a idade:");  
scanf("%d", &p3->idade);  
printf("Informe o telefone:");  
scanf("%s", p3->telefone);
```



Atenção &

Structs

- Estrutura usando ponteiro **p*
 - Saída na tela
 - Operador “->”

```
printf("\nCod:%d\nNome= %s %s\nIdade= %d\nTelefone= %s",  
      p3->cod,p3->nome,p3->sobrenome,p3->idade,p3->telefone);
```

```
Cod:2  
Nome= juca farias  
Idade= 34  
Telefone= 23432432
```

Comando TypeDef

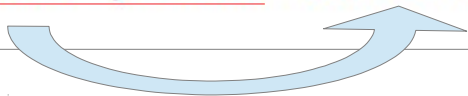
- Fornece um mecanismo para criação de sinônimos para tipos de dados;
 - Escrever **struct nametype** é igual a **novo_tipo**;

Definição

```
typedef struct nome_struct novo_tipo (apelido);
```

Exemplo;

```
typedef struct pessoa cliente;
```



São equivalentes
(struct pessoa) = cliente

Comando TypeDef

Sem a palavra Struct

```
//uso da estrutura pessoa, com nome de cliente
cliente cl1 = {1, "Alex", "Silva Souza", 34, "98345674"};

printf("\nCod:%d\nNome= %s %s\nIdade= %d\nTelefone= %s\n",
       cl1.cod, cl1.nome, cl1.sobrenome, cl1.idade, cl1.telefone);
```

Comando typedef - simplificado

- Apelido para struct pessoa = Comprador;

```
typedef struct pessoa{  
    int cod;  
    char nome [10];  
    char sobrenome [20];  
    int idade;  
    char telefone [10];  
}compardor;
```

```
compardor *f1 = NULL;
```

Declaração de um ponteiro

Array de struct

- Criando um array de estruturas

```
compador cp [10];
```

```
cp[0].cod =1; strcpy(cp[0].nome,"Franco");  
strcpy(cp[0].nome,"Giorgi");  
strcpy(cp[0].telefone,"23423432");  
cp[0].idade = 50;
```

```
cp[1].cod =2;  
strcpy(cp[1].nome,"Renner");  
strcpy(cp[1].nome,"Renner");  
strcpy(cp[1].telefone,"23423432");  
cp[1].idade = 50;
```

```
cp[2].cod =3;  
strcpy(cp[2].nome,"C&A");  
strcpy(cp[2].nome,"cea");  
strcpy(cp[2].telefone,"23423432");  
cp[2].idade = 50;
```

```
for (int i=0;i<3;i++){  
    printf("\nCod:%d\nNome= %s %s\nIdade= %d\nTelefone= %s\n",  
        cp[i].cod,cp[i].nome,cp[i].sobrenome,cp[i].idade,cp[i].telefone);  
}
```

Array de struct

- Saída do vetor de comprador;

```
Cod:1  
Nome= Giorgi  
Idade= 50  
Telefone= 23423432  
  
Cod:2  
Nome= Renner  
Idade= 50  
Telefone= 23423432  
  
Cod:3  
Nome= cea  
Idade= 50  
Telefone= 23423432  
  
Process returned 0 (0x0)   execution time : 0,001 s  
Press ENTER to continue.  
]
```

Sistema de Locadora

- Vamos desenvolver um sistema para armazenar dados de clientes, filmes(dvd) e locações;
 - Para clientes armazenar:
 - Código, Nome e Sobrenome, Idade, Telefone;
 - Para DVDs armazenar:
 - Código, Título, Ano e Status(1-disponível, 0- locado);
 - Para Locação armazenar:
 - Código, Cliente, DVD, Dias, Valor diária, Valor Total;

Sistema de Locadora - Estruturas

```
typedef struct pessoa{  
    int cod;  
    char nome [10];  
    char sobrenome [20];  
    int idade;  
    char telefone [10];  
}Cliente;
```

```
typedef struct dvd{  
    int cod;  
    char titulo [20];  
    int ano;  
    int status;  
}Dvd;
```

```
typedef struct locacao{  
    Cliente *cl;  
    Dvd *dv;  
    int cod;  
    int dias;  
    float valor;  
    float total;  
}Locacao;
```

Sistema de Locadora -

- O sistema deverá armazenar os dados de até 10 clientes, 10 DVDs, e 10 Locações. Utilizar vetor ou array;

```
Cliente cls [10];  
Dvd dvds [10];  
Locacao locados [10];
```

Vetores de Structs

Sistema de Locadora -

- No uso de função e Structs recomenda-se o uso de ponteiros.
Função: ***void leDadosCliente(Cliente *c, int cod);***

```
void leDadosCliente(Cliente *c, int cod){  
    c->cod = cod;  
    printf("Informe o nome:");  
    scanf("%s", c->nome);  
    printf("Informe o sobrenome:");  
    scanf("%s", c->sobrenome);  
    printf("Informe a idade:");  
    scanf("%d", &c->idade);  
    printf("Informe o telefone:");  
    scanf("%s", c->telefone);  
}
```

Ponteiro *c, manipula os dados do dentro do vetor;

Sistema de Locadora -

- No uso de função e Structs recomenda-se o uso de ponteiros.
Função: ***void imprimeCliente(Cliente *c);***

```
void imprimeCliente(Cliente *c){  
    printf("\nCod:%d\nNome= %s %s\nIdade= %d\nTelefone= %s\n",  
        c->cod, c->nome, c->sobrenome, c->idade, c->telefone);  
}
```

Ponteiro *c, acessa os dados dentro do vetor;

Sistema de Locadora -

- No uso de função e Structs recomenda-se o uso de ponteiros.
Chamada das Funções: ***leDadosCliente*** e ***imprimeCliente***;
 - ***Uso do “&” para passar o endereço da posição do vetor;***

```
printf("\nEntrada de Dados de Cliente:\n");  
leDadosCliente(&cls[0],cod_cliente);
```

```
printf("\nSaida de Dados de Cliente:\n");  
imprimeCliente(&cls[0]);
```

Chamada das Funções

Sistema de Locadora -


- Defina um sistema com as seguintes funções:
 - void leDadosDvd(Dvd *d, int cod);
 - Preenche os dados referente ao cadastro de um DVD;
 - void imprimeDvd(Dvd *d);
 - Imprime os dados de um cadastro de DVD;
 - void leDadosLocacao(Locacao *l, Cliente *c, Dvd *d, int cod);
 - Preenche os dados de uma locação;
 - void imprimeLocacao(Locacao *l);
 - Imprime os dados de uma locação;

Sistema de Locadora -

- Defina um sistema com as seguintes funções:
 - void leDadosDvd(Dvd *d, int cod);
 - Preenche os dados referente ao cadastro de um DVD;
 - void imprimeDvd(Dvd *d);
 - Imprime os dados de um cadastro de DVD;
 - void leDadosLocacao(Locacao *l, Cliente *c, Dvd *d, int cod);
 - Preenche os dados de uma locação. Verificar se o DVD está disponível(status = 1)
 - void imprimeLocacao(Locacao *l);
 - Imprime os dados de uma locação;

Atividade 2,

- Realizar o cadastro de no mínimo 4 Clientes, DVDs, e Locações;
- Realizar as locações;
- Realizar as devoluções;
- Realizar novas locações;

- 
- Bom material sobre Ponteiros e Struct
 - Livro: Linguagem C completa e descomplicada (André Backes)
 - PDF