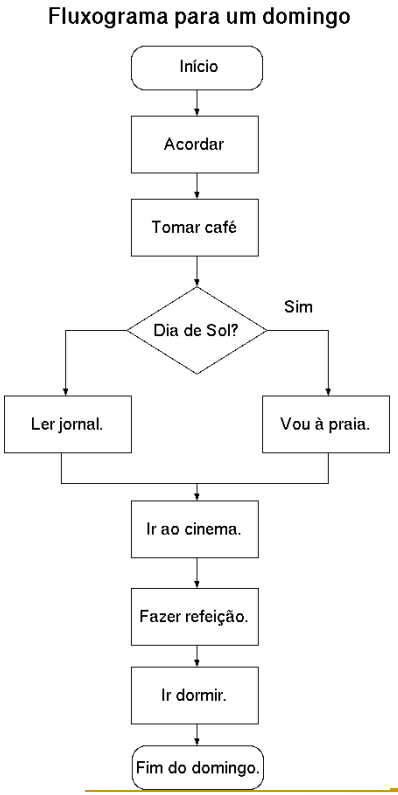
UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de São José do Rio Preto



Ciência da Computação

Prof. Dr. Leandro Alves Neves

Aula 12

Algoritmos e Programação

Conteúdo

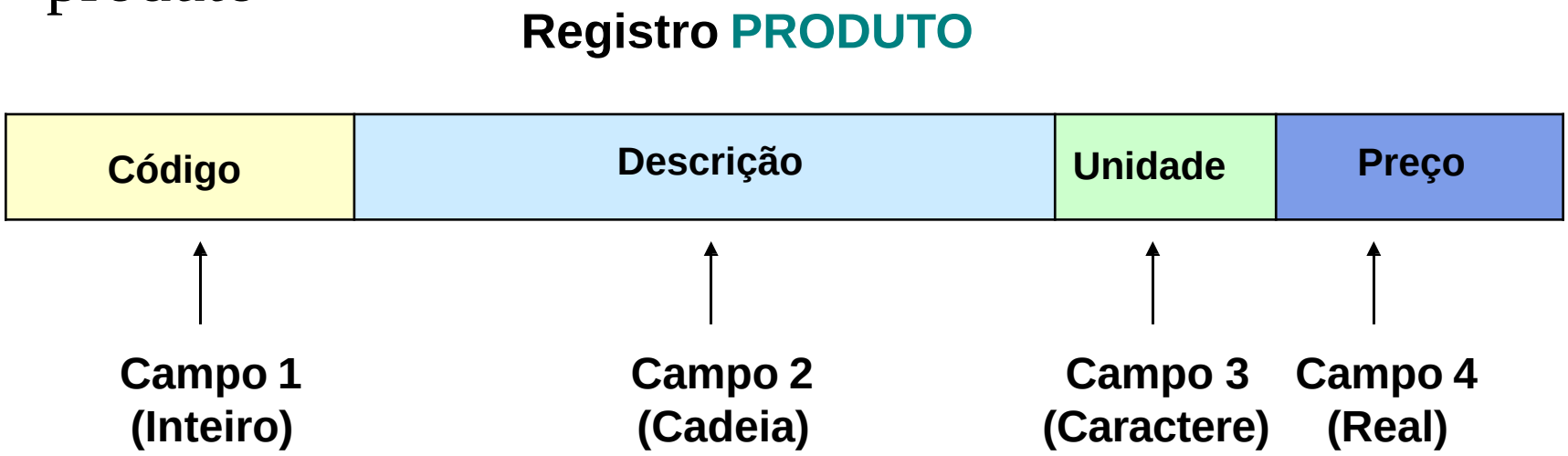
- Estruturas de Dados Compostas Heterogêneas
  - Variáveis compostas heterogêneas – registros
    - Compostos Simples: Registros
    - Leitura e Atribuição
    - Manipulação de registros
    - Vetor de Registros

1. Variáveis Compostas Heterogêneas: Registros

- Registro
  - É formado por um conjunto de dados logicamente relacionados
  - Elementos são geralmente **heterogêneos** (isto é, de tipos diferentes)
  - Cada elemento é chamado de **campo** do registro
  - Outro nome do registro:
    - estrutura ou **struct**

2. Registros

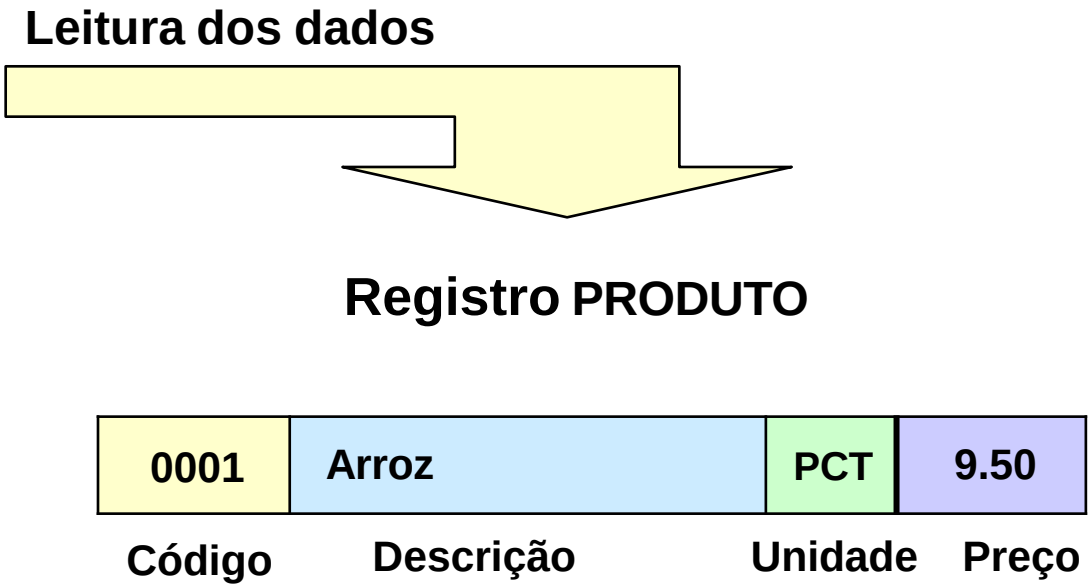
- **Exemplo**: registro que armazena os dados de um produto



- É possível acessar um único **campo** por vez
  - As operações são definidas de acordo com a respectiva ED do campo

2. Registros

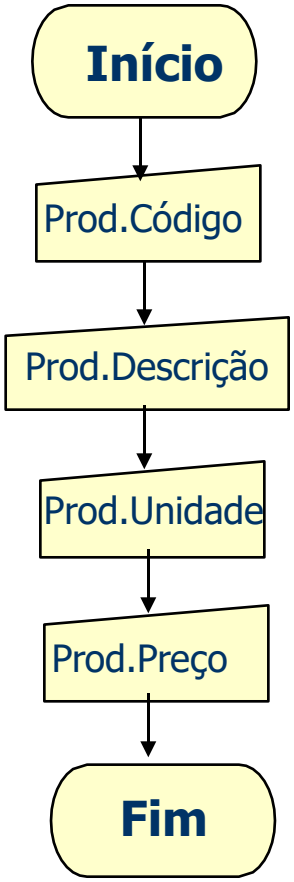
- Campo
  - Cada campo está associado a uma ED
    - primitiva: inteiro, real, lógico, caracter
    - composta: cadeia, registro, arranjo, lista, ...



2.1 Leitura de Registros

Algoritmo Leitura

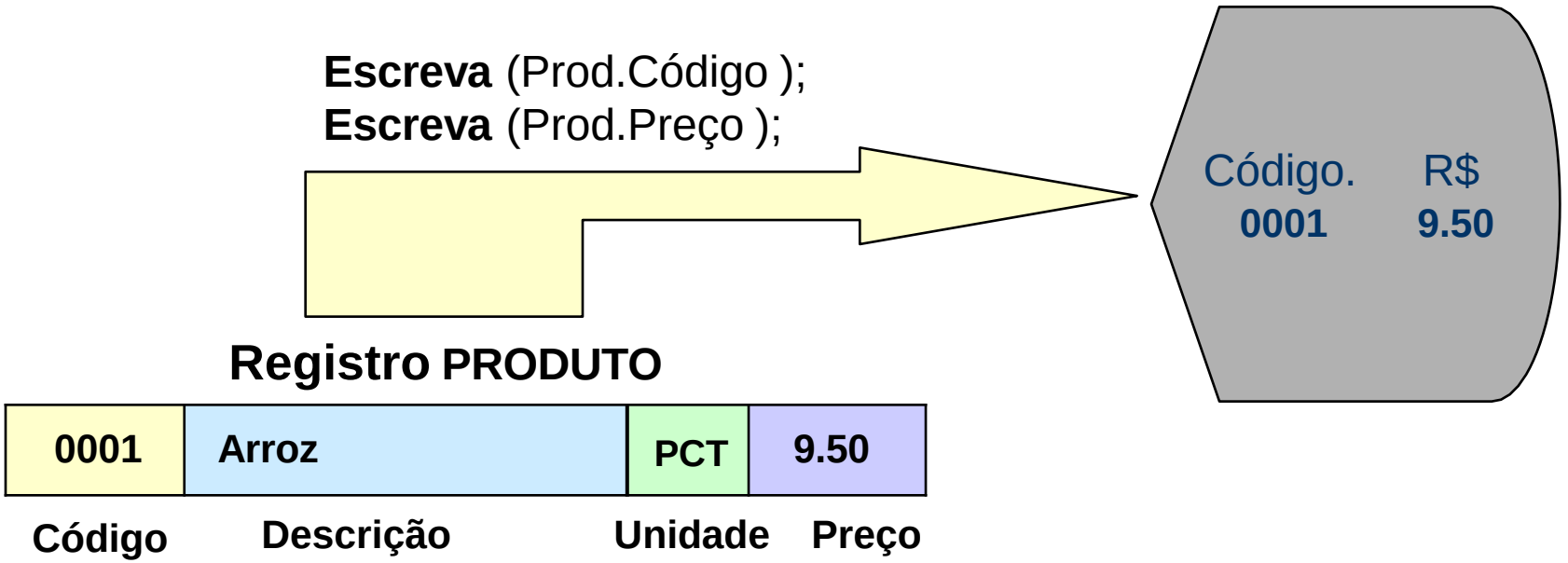
```
TIPO Produto = Registro
                    Código: inteiro
                    Descrição: cadeia
                    Unidade: cadeia
                    Preço: real
               Fim Registro
Var
   Prod: Produto
Início
   leia Prod.Código
   leia Prod.Descrição
   leia Prod.Unidade
   leia Prod.Preço
Fim
```



operador . (ponto): permite acessar cada campo

2.1 Registros

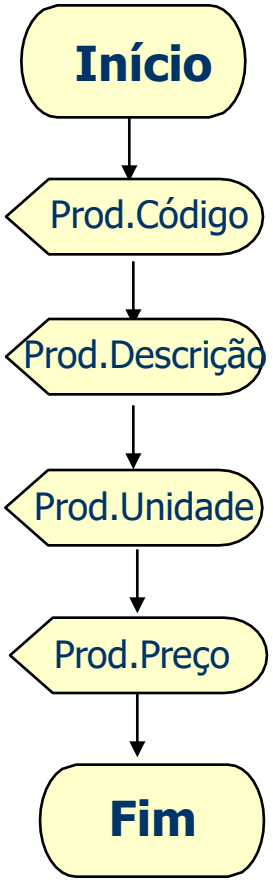
- **Exemplo**: utilização dos dados do registro lido, campo-a-campo
  - operador . (ponto): permite acessar cada campo



2.2 Escrita de Registros

Algoritmo Escrita

```
TIPO Produto = Registro
                    Código: inteiro
                    Descrição: cadeia
                    Unidade: cadeia
                    Preço: real
               Fim Registro
var
   Prod: Produto
início
   Escreva Prod.Código
   Escreva Prod.Descrição
   Escreva Prod.Unidade
   Escreva Prod.Preço
fim
```



operador . (ponto): permite acessar cada campo

2.3 Registros

- Em linguagem **C**
  - Registros são chamados de **structs**

```
struct produto {
   int     codigo;
   char descricao[50];
   char unidade[3];
   float preco;
} prod;
```

Informa ao compilador que um modelo de estrutura está sendo definido (tipo de dado)

Nome da variável do tipo de dado produto

  - Declaração de uma variável struct (aloca memória)
  - Utilização

```
prod.preco = 9.50;
```

2.4 Leitura de Registros em C

- **Implementação do Algoritmo**

```
//Aula - Struct (registro) - Prof. Leandro A. Neves
#include<stdio.h>
#include<stdlib.h>
int main()
{
   //Definição do registro e seus respectivos campos
   struct produto
          {
            int codigo;
            char descricao[50];
            char unidade[3];
            float preco;
          } prod; //Declaração da variável prod com o tipo struct produto
```

2.5 Leitura de Registros em C

- **Implementação do Algoritmo**

```
  //Entrada dos dados
  printf("\nDigite a decrição do produto .: ");
  scanf(" %[^\n]s",prod.descricao);
  printf("\nDigite a unidade do produto ..: ");
  scanf("  %[^\n]s",prod.unidade);
  printf("\nDigite o preco do produto ....: ");
  scanf("%f",&prod.preco);
  printf("\nDigite o codigo do produto ...: ");
  scanf("%d",&prod.codigo);
  //Saída dos dados Lidos

  printf("\n\nCodigo do produto ...: %d",    prod.codigo);
  printf("\nDecrição do produto ...: %s",    prod.descricao);
  printf("\nUnidade  do produto ...: %s",    prod.unidade);
  printf("\nPreco do produto ......: R$ %.2f", prod.preco);
//  system("PAUSE");
  return 0;
}
```

2.6 Outros Exemplos em C

- **Struct considerando apenas o nome da Variável**

```
2   #include <stdio.h>
3   #include <stdlib.h>
4   int main()
5   {
6      //Definição do registro e seus respectivos campos
7      struct{
8                int codigo;
9                char end[20];
10             }cli; //Declaração da variável cli do tipo struct
11
12     //Entrada dos dados
13     printf("\nDigite o codigo.: ");
14     scanf("%d",&cli.codigo);
15     printf("\nDigite o endereco: ");
16     scanf(" %[^\n]s",cli.end);
17
18     //Saída dos dados Lidos
19     printf("\n\nCodigo do cliente ...: %d",    cli.codigo);
20     printf("\nEndereco ...: %s",   cli.end);
21     printf("\n");
22    // system("PAUSE");
23     return 0;
24  }
```

Modelo de Estrutura é utilizado uma única vez.

2.6 Outros Exemplos em C

- **Struct considerando o nome da estrutura e o nome da Variável**

```c
//Aula - Struct (registro) - Prof. Dr. Leandro A. Neves
#include <stdio.h>
#include <stdlib.h>
int main()
{
   //Definição do registro e seus respectivos campos
   struct cliente {
            int codigo;
            char end[20];
         }cli, cli2;
   //Declaração de cli e cli2 do tipo struct
   // ou
   //struct cliente cli, cli2;

   //Entrada dos dados
   printf("\nDigite o codigo.: ");
   scanf("%d",&cli.codigo);
   printf("\nDigite o endereco: ");
   scanf(" %[^\n]s",cli.end);

   //Saída dos dados Lidos
   printf("\n\nCodigo do cliente ...: %d",   cli.codigo);
   printf("\nEndereco ...: %s",  cli.end);
   printf("\n");
   //system("PAUSE");
   return 0;
}
```

Modelo de Estrutura pode ser aplicado para declarar outras variáveis: foi incluído cli2 para exemplificar

2.6 Outros Exemplos em C

- **Struct considerando apenas o nome da estrutura**

```c
//Aula - Struct (registro) - Prof. Dr. Leandro A. Neves
#include <stdio.h>
#include <stdlib.h>
int main()
{
   //Definição do registro e seus respectivos campos
   struct cliente {
            int codigo;
            char end[20];
          };
   struct cliente cli; //declaração da variável cli com o tipo cliente

   //Entrada dos dados
   printf("\nDigite o codigo.: ");
   scanf("%d",&cli.codigo);
   printf("\nDigite o endereco: ");
   scanf(" %[^\n]s",cli.end);

   //Saída dos dados Lidos
   printf("\n\nCodigo do cliente ...: %d",   cli.codigo);
   printf("\nEndereco ...: %s",  cli.end);
   printf("\n");
  // system("PAUSE");
   return 0;
}
```

Modelo de Estrutura pode ser aplicado para declarar diferentes variáveis

2.6 Outros Exemplos em C

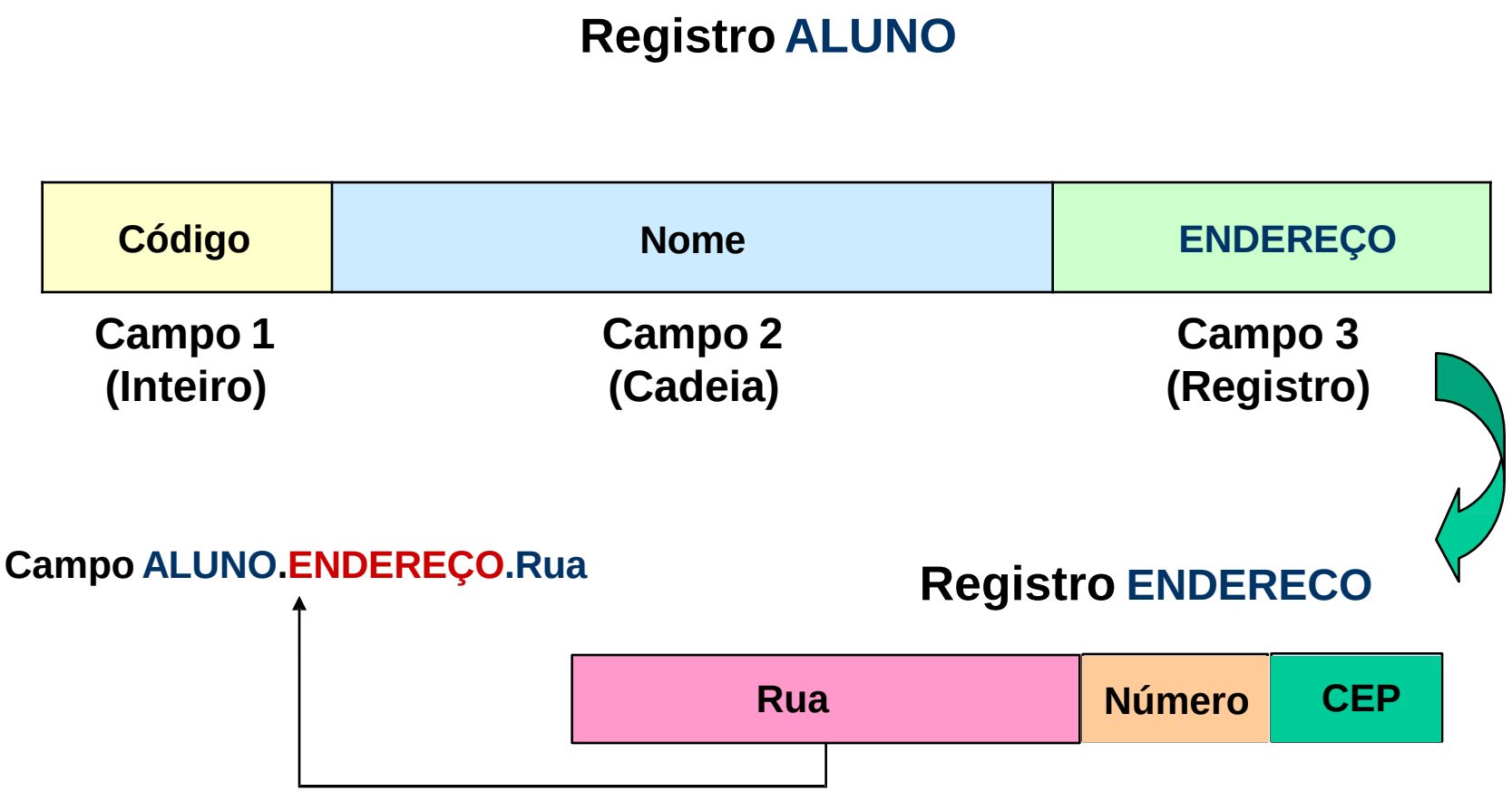
- **Struct com typedef** (permite ao programador definir um novo nome para um determinado tipo)

```
1  //Aula - Struct (registro) - Prof. Dr. Leandro A. Neves
2  #include <stdio.h>
3  #include <stdlib.h>
4  int main()
5  {
6     //Definição do registro e seus respectivos campos
7      typedef struct cliente{ int codigo;
8                char end[20];
9              }Tcli; //Declaração do tipo struct Tcli
10    Tcli cli; //declaração da variável cli com o tipo Tcli
11
12    //Entrada dos dados
13    printf("\nDigite o codigo.: ");
14    scanf("%d",&cli.codigo);
15    printf("\nDigite o endereco: ");
16    scanf(" %[^\n]s",cli.end);
17
18    //Saída dos dados Lidos
19    printf("\n\nCodigo do cliente ...: %d",    cli.codigo);
20    printf("\nEndereco ...: %s",  cli.end);
21    printf("\n");
22   // system("PAUSE");
23    return 0;
24 }
```

Útil para simplificar declarações

2.7 Registros

- **Exemplo**: registro contendo outro registro



2.7 Registros

- **Exemplo em C**: registro contendo outro registro

```
//Aula Exemplo: Struct (registro) - Prof. Leandro A. Neves
#include <stdio.h>
#include <stdlib.h>
int main()
{
    //Definição do registro e seus respectivos campos
   typedef struct endereco{
      char rua[20];
      char numero[4];
      char cep[4];
    }Tend; //Declaração do tipo struct Tend
   typedef struct cliente{
      int codigo;
      Tend end;  //declaração da variável end com o tipo endereco
    }Tcli; //Declaração do tipo struct cliente

   Tcli cli; //declaração da variável cli com o tipo (modelo) cliente
   //Entrada dos dados
```

2.7 Registros

- **Exemplo em C**: registro contendo outro registro

```
//Entrada dos dados
   printf("\nDigite o codigo.: ");
   scanf("%d",&cli.codigo);
   printf("\nDigite o nome da rua: ");
   scanf(" %[^\n]s",cli.end.rua);
   printf("\nDigite o numero da residencia: ");
   scanf(" %[^\n]s",cli.end.numero);
   printf("\nDigite o cep: ");
   scanf(" %[^\n]s",cli.end.cep);

   //Saída dos dados Lidos
   printf("\n\nCodigo do cliente ...: %d", cli.codigo);
   printf("\nNome da rua ...: %s", cli.end.rua);
   printf("\nNumero da residencia ...: %s", cli.end.numero);
   printf("\nNumero da residencia ...: %s", cli.end.cep);
   printf("\n");
 //  system("PAUSE");
   return 0;

}
```

2.8 Vetor de Registros

- **Exemplo em C**

```
#define registros 20

int main()
{  typedef struct endereco{
          char rua[20];
          char numero[4];
          char cep[4];
    }Tend; //Declaração do tipo struct Tend

   typedef struct cliente{
          int codigo;
          char nome[41];
          Tend end; //declaração da variável end com o tipo endereco
         }Tcli; //Declaração do tipo struct cliente
   Tcli cli[registros]; //declaração do vetor cli, tipo (modelo) cliente, com n elementos
...
```

Útil para armazenar n registros com a mesma estrutura

2.8 Vetor de Registros

- **Exemplo em C**: registro contendo outro registro

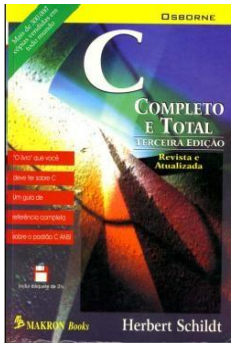
```
//Aula Exemplo: Struct (vetor de registros) - Prof. Dr. Leandro A. Neves
#include <stdio.h>
#include <stdlib.h>
#define registros 3
int main()
{
    //Definição do registro e seus respectivos campos
   typedef struct endereco{
      char rua[20];
      char numero[4];
      char cep[4];
    }Tend; //Declaração do tipo struct Tend
   typedef struct cliente{
      int codigo;
      Tend end;  //declaração da variável end com o tipo endereco
    }Tcli; //Declaração do tipo struct cliente
   Tcli cli[registros]; //declaração da variável cli com o tipo (modelo) cliente
```

2.8 Vetor de Registros

- **Exemplo em C**

```
for(int i=0; i<registros; i++)
{    //Entrada dos dados
     printf("\nDigite o codigo.: ");
     scanf("%d",&cli[i].codigo);
     printf("\nDigite o nome da rua: ");
     scanf(" %[^\n]s",cli[i].end.rua);
     printf("\nDigite o numero da residencia: ");
     scanf(" %[^\n]s",cli[i].end.numero);
     printf("\nDigite o cep: ");
     scanf(" %[^\n]s",cli[i].end.cep);
 }
 for(int i=0; i<registros; i++)
 {  //Saída dos dados Lidos
     printf("\n\nCodigo do cliente ...: %d", cli[i].codigo);
    printf("\nNome da rua ...: %s", cli[i].end.rua);
    printf("\nNumero da residencia ...: %s", cli[i].end.numero);
    printf("\nNumero da residencia ...: %s", cli[i].end.cep);
    printf("\n");
 }
 //  system("PAUSE");
   return 0;
}
```

Bibliografia



1. Schildt, H. C completo e total. São Paulo: Makron Books, 3ª ed, 1997.

- Páginas 167 a 172: tópicos Estruturas, Referenciando elementos de Estruturas, Atribuição de Estruturas e Matrizes de Estruturas (somente páginas 171 e 172)
- Páginas 185 e 186: tópico Matrizes e Estruturas Dentro de Estruturas
- Página 196: tópico Typedef

Bibliografia

2. Ascencio, A. F. G.; Campos, E. A. V. Fundamentos da Programação de Computadores: Algoritmos, Pascal, C/C++ e Java. 3ª. Ed. São Paulo: Pearson Education do Brasil, 2012.

- Registros: Páginas 333 a 335. Seções 10.1 e 10.2;
- Seção 10.4: Declaração de registros em C/C++, a partir da página 338.
- Exercícios Resolvidos, páginas 349 a 413