

27. Registros

DIM0321

2015.1

Outline

- 1 Introdução
- 2 Exemplos
- 3 Definição de novos tipos

- 1 Introdução
- 2 Exemplos
- 3 Definição de novos tipos

Tipos de variáveis

Básicas

- `int`
- `float`
- `char`

Compostas

Arranjos estruturas **homogêneas** (conjunto do mesmo tipo dado indexado)
Registros estruturas **heterogêneas** (conjunto de tipos diferentes, **nomeados**)

Registro

Podem conter qualquer tipo de dados

- Tipos básicos / primitivos
- Ponteiros / arranjos
- **Registros**

Cada elemento — chamado de **campo** — tem um nome associado.

Elementos sintáticos

- Uso da palavra-chave `struct`
- Separação dos campos por `*','*`

Declaração / inicialização

- É a declaração de um **novo tipo de dados** `struct <nome>`
- A inicialização tem a mesma sintaxe que a dos arranjos.

Acesso

- `<nome>.<campo>`

Exemplo introdutivo

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      struct produto {
6          int codigo;
7          char descricao[50];
8          float valor;
9          int estoque;
10     };
11
12     struct produto q;
13
14     struct {
15         int foo;
16     } bar;
17
18     struct produto p;
19
20     // Inicialização no ponto de declaração
21     struct produto pc = { 104, "caneta", 3.25, 1598 };
22     pc.codigo = 104, pc.valor = 3.25;
23     // Leitura de campos
24     scanf("%d %s %f %d", &p.codigo, &p.descricao, &p.valor, &p.estoque);
25     // Atribuição
26     bar.foo = 42;
27     /* Acesso */
28     printf("%d", pc.codigo);
29 }
```

- 1 Introdução
- 2 Exemplos
- 3 Definição de novos tipos

Acesso, inicialização, declaração

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      struct { /* nome do registro omitido */
6          int codigo;
7          char descricao[50];
8          float valor;
9          int estoque;
10     } produto = { 104, "caneta", 3.25, 1598 };
11
12     printf("descricao      = %s\n",   produto.descricao);
13     printf("estoque        = %i\n",   produto.estoque);
14     printf("valor unitario = %.2f\n", produto.valor);
15     printf("valor total    = %.2f\n", produto.estoque * produto.valor);
16     return 0;
17 }
```


Registros e arranjos

```
1  #include <stdio.h>
2  int main (void)
3  {
4      struct produto {
5          int codigo;
6          char descricao[50];
7          float valor;
8          int estoque;
9      };
10     struct produto produto1 = { 104, "caneta", 3.25, 1598 };
11     struct produto produto2 = { 105, "lapis", 2., 2365 };
12     printf("Produto 1:\n");
13     printf("descricao   = %s\n", produto1.descricao);
14     printf("estoque      = %i\n", produto1.estoque);
15     printf("valor total = %.2f\n", produto1.estoque * produto1.valor);
16     printf("Produto 2:\n");
17     printf("descricao   = %s\n", produto2.descricao);
18     printf("estoque      = %i\n", produto2.estoque);
19     printf("valor total = %.2f\n", produto2.estoque * produto2.valor);
20     struct produto produtos[3] = { {104, "caneta", 3.25, 1598},
21                                     {105, "lapis", 2., 2365},
22                                     {106, "caderno", 10.85, 127} };
23     for (int i = 0; i < 3; ++i) {
24         printf("Produto %i:\n", i + 1);
25         printf("descricao   = %s\n", produtos[i].descricao);
26         printf("estoque      = %i\n", produtos[i].estoque);
27         printf("valor total = %.2f\n", produtos[i].estoque * produtos[i].valor);
28     }
29     return 0;
30 }
```

- 1 Introdução
- 2 Exemplos
- 3 Definição de novos tipos**

Definição de tipos: typedef

Uso

A palavra reservada `typedef` permite que um determinado tipo possa ser denominado de forma diferente de acordo com as necessidades do usuário.

Exemplo

```
1 int main(void)
2 {
3     typedef int meu_inteiro;
4     meu_inteiro n = 10;
5     n = n * 5;
6
7     typedef struct {
8         char nome[50];
9         char sexo;
10        char estado_civil;
11        int idade;
12    } pessoa; /* não pode criar variáveis */
13
14    pessoa homem, mulher; /* struct eliminado */
15 }
```

Registros de registros com typedef

```
1 typedef struct {
2     int dia;
3     char mes[3 + 1]; /* 1 caractere reservado para o '\0' */
4     int ano;
5 } data;
6
7 typedef struct pessoa {
8     char nome[50];
9     char sexo;
10    char estado_civil;
11    data data_nascimento; /* registro definido acima */
12    struct pessoa *pai;
13 } pessoa_t, person, *pessoa_p;
14
15
16 pessoa_t a;
```

Referências

Precisões

[KR88] 6

[Bac13] 8.1



André Backes, *Linguagem C completa e descomplicada*, Elsevier, 2013.



Brian W. Kernighan and Dennis M. Ritchie, *The (ANSI) C Programming Language*, 2nd ed., Prentice Hall Professional Technical Reference, 1988.

Perguntas ?



<http://dimap.ufrn.br/~richard/dim0321>