Professor
Edson Pedro FERLIN

Programação em C

Tipos Avançados Definidos pelo Usuário

Prof. Edson Pedro Ferlin

Professor
Edson Pedro FERLIN

Programação em C

Introdução

- A linguagem C permite que se crie cinco tipos diferentes de dados personalizados:
  - Estrutura (struct) → grupo de variáveis sob o mesmo nome;
  - Campo de bit → fácil acesso aos bits dentro de uma palavra;
  - União (union) → define-se a uma mesma parte da memória como dois ou mais tipos diferentes de variável;
  - Enumeração (enum) → lista de símbolos;
  - Tipo novo (typedef) → cria um novo nome para um tipo já existente.

Estruturas
Struct

- Os elementos das estrutura estão relacionados logicamente uns com os outros;

```
struct addr {
        char nome[30];
        char rua[40];
        char cidade[20];
        char estado [3];
        unsigned long int cep;
};

struct addr end_info;

end_info.cep=80000;
```

- Matrizes de estrutura

```
struct addr end_m[100];

end_info[2].cep = 85000;
```


Estruturas
Elementos da struct para funções

- Passar o valor do elemento;

```
struct test {
        char x;
        int y;
        float z;
        char s[10];
} teste1;

func1 (teste1.x);     /* valor de caracter x */
func2 (teste1.s);     /* endereco da string s */
```

Estruturas
Estrutura inteira para funções

```
main()
{
   struct {
          int a, b;
          char ch;
   } argu;
   argu.a=1000;
   f1 (argu);
}
f1(parm)
struct {
       int x, y;
       char ch;
} parm;
{
   prinf("%d", parm.x);
}
```

```
struct estru_d{
       int a, b;
       char ch;
       } ;
main()
{
       struct estru_d argu;
       argu.a=1000;
       f1 (argu);
}
f1(parm)
struct estru_d param;
{
       prinf("%d", parm.a);
}
```

O tipo do argumento deve coincidir com o tipo do parâmetro;

Estruturas
Ponteiros para estruturas

- Mesma maneira que para outro tipo de variável;

```
struct  bal {                  /* definicao estrutura*/
          float balanco;
          char nome[80];
   } pessoa;

  float f;

struct bal *p;                 /* declarou-se um ponteiro */

p = &pessoa;                   /* endereco de pessoa para o ponteiro p  */

f = (*p).balanco;     /* faz referencia ao elemento balanco */
```

Os parenteses são necessários.

(*p).balanco
P->balanco

Campo de Bit

- É um tipo especial que define o comprimento em *bits* de cada elemento;
- Variáveis booleanas;
- Informações codificadas;
- Acesso de *bits* dentro de um *byte*;
- Eficiência;
- Programa mais portátil.

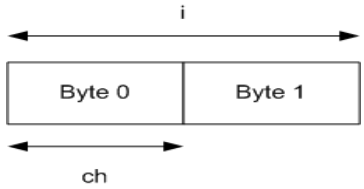
```
struct device {
        unsigned active: 1;
        unsigned ready: 1;
        unsigned error: 1;
} dev_code;

dev_code.active
```

Restrições:

•Não se pode pegar o endereço de uma variável de campo de *bit*;
•Não se pode colocar em matrizes;
•Não se pode ultrapassar os limites de inteiros.

Uniões
Union

- É uma localização de memória que é usada por muitas variáveis, que podem ser de tipos diferentes;
- São usadas quando as conversões de tipo são necessárias.




```
union u_type {
        int i;
        char ch;
} cntv;

cntv.i = 10;

func1(un)
union u_type *un;
{
        un ->i = 10;
}
```

Enumerações
Enum

- É um conjunto de constantes inteiras com nome e especifica todos os valores legais que uma variável daquele tipo pode ter;
- O ponto-chave sobre a enumeração é que cada símbolo significa um valor inteiro.

```
enum moeda {penny, nickel, dime, quarter, half-dollar, dollar}
enum moeda dinheiro;

dinheiro = dime;

If (dinheiro == quarter) printf ("Eh um quarter \n");

printf ("%d %d", penny, dime);          /* 0 2 */
```


Typedef

- Usada para se definir um novo nome para um tipo existente;
- O ponto-chave sobre a enumeração é que cada símbolo significa um valor inteiro.

```
typedef float balanco;

balanco extrato_conta;
```

Contato

eferlin@live.com

(BLOG) professorferlin.blogspot.com

(SITE) professorferlin.webnode.com.br

(YOUTUBE) ProfEdsonPedroFerlin