Ponteiros

Módulo 8 Aula 0

Linguagem C, o Curso Definitivo WR Kits

Autor: Dr. Eng. Wagner Rambo

Um ponteiro consiste em uma variável que contém um endereço de memória, que normalmente é o endereço de outra variável. Em outras palavras, são apontadores de endereços de memória. Quando um ponteiro contém o endereço de uma variável, dizemos que o mesmo aponta para o endereço da referida variável. Com ponteiros, podemos modificar argumentos de funções, trabalhar com alocação dinâmica de memória e também tornar certas rotinas mais eficientes. Para declarar um ponteiro, precisamos utilizar o operador *, como no Box 1.

```
tipo *nome_do_ponteiro;
```

Box 1 - Sintaxe da declaração de um ponteiro.

O tipo refere-se aos tipos de dados da Linguagem C, um ponteiro só poderá apontar para endereços de variáveis que tenham o mesmo tipo de dados que o próprio ponteiro.

Os dois operadores especiais para ponteiros são o * e o &. O operador & é unário e devolve o endereço de memória do seu operando. O programa do Box 2 mostra o endereço onde a variável x está localizada na memória.

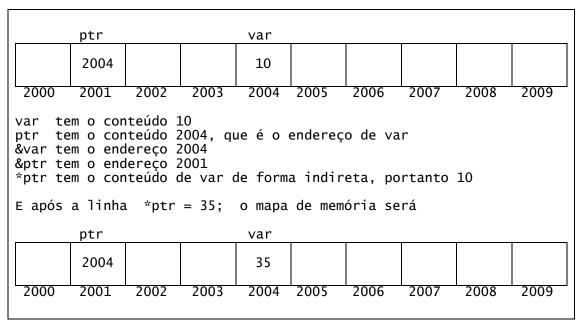
Box 2 - Imprimindo o endereço de uma variável.

O operador * é conhecido como operador de indireção, ou podemos ler como "apontado por", sendo também unário e utilizado para lermos o conteúdo do endereço do qual o ponteiro aponta. Além disso, poderá ser usado para escrever no conteúdo, atualizando assim o valor da variável por ele apontada, de forma indireta. O programa a seguir ilustra a operação com o operador de indireção, confira no Box 3.

```
main()
  int *ptr, var;
                            /* ptr é ponteiro para inteiro,
                                     var é uma variável comum. */
                             /* var recebe o valor 10 */
  var = 10;
  ptr = &var;
                             /* ptr apontará para o endereço de var */
                            /* imprime o conteúdo de var, 10 no caso */
/* imprime o endereço de var */
  printf("%d\n",
                    var);
  printf("%p\n", ptr);
printf("%d\n", *ptr);
printf("%p\n", &ptr);
                            /* imprime o conteúdo de var, 10 no caso */
                             /* imprime o endereço de ptr, pois
                                um ponteiro também existirá em um
                                endereço da memória */
  *ptr = 35;
                             /* esta linha atualiza o conteúdo de var
                                para 35 */
  printf("%d\n", var); /* imprime o conteúdo de var, 35 no caso */
} /* end main */
```

Box 3 - Utilizando o operador *.

No caso do programa do Box 3, podemos enxergar a organização de memória conforme demonstrado no Box 4.



Box 4 - Mapa de memória do programa do Box 3.

Qualquer ponteiro deve ser previamente inicializado em seu código, do contrário, poderão ocorrer problemas no sistema. Caso o seu ponteiro inicialmente não aponte para nada, você deve atribuir a ele a constante simbólica NULL, que significa o endereço 0 de memória.

int *ptr = NULL;

Box 5 - Inicializa o ponteiro com NULL para que ele não aponte para nada.

Bibliografia: DAMAS, Luís; Linguagem C, décima edição.

Disponível em: https://amzn.to/3nGdlbN