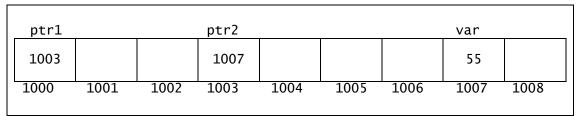
Ponteiros de Ponteiros

Módulo 8 Aula 4

Linguagem C, o Curso Definitivo WR Kits

Autor: Dr. Eng. Wagner Rambo

Sendo ponteiros variáveis como outras quaisquer, já vimos que os mesmos encontramse em endereços específicos da memória. Logo, se um ponteiro existe em uma posição de memória, podemos utilizar um segundo ponteiro para aponta-lo. Esta técnica é conhecida como ponteiros de ponteiros, ou também indireção múltipla. Veja o mapa de memória do Box 1.



Box 1 - ptr1 é um ponteiro de ponteiro e aponta para o endereço de ptr2.

Como pode ser visto, ptr1 é um ponteiro que aponta para o endereço de ptr2, outro ponteiro, este por sua vez aponta para o endereço de var, uma variável comum. Para declarar um ponteiro que apontará para o endereço de outro ponteiro, utilizamos dois operadores de indireção antes do seu nome (**). No Box 2, os 3 printf's apresentarão o resultado 55 na tela.

```
main()
{
  int **ptr1, *ptr2, var = 55;

  ptr2 = &var;
  ptr1 = &ptr2;

  printf("%d\n", var); /* imprimirá 55 */
  printf("%d\n", *ptr2); /* imprimirá 55 */
  printf("%d\n", **ptr1); /* imprimirá 55 */
} /* end main */
```

Box 2 - Utilizando ponteiros de ponteiros.

Em C não existe limite para a indireção múltipla, você pode ter ponteiros de ponteiros de ponteiros e assim por diante, de acordo com o número de asteriscos usados na declaração.

```
float ***p1; /* todas estas declaração são válidas */
int ***p2;
char ******************************
```

Box 3 - Declarando ponteiros de ponteiros de ponteiros...

Convém destacar que será muito improvável a necessidade de utilizar mais do que um ponteiro de ponteiro para qualquer solução com C. Códigos com uma indireção múltipla muito longa podem ser difíceis de analisar e levar a resultados imprevisíveis.

Um exemplo prático e eficiente de uso de ponteiros para ponteiros é na passagem de vetores de *string* para funções. Tomando como exemplo o código do Box 4, onde passaremos o vetor de *string* que, com auxílio da função, será exibido na tela. Um vetor de *string* basicamente consiste em uma matriz bidimensional, onde cada linha armazena uma *string* e cada coluna é um espaço reservado para caracteres. Lembrando sempre de prever um espaço extra para o caractere nulo.

Box 4 - Utilizando ponteiros de ponteiros para passar matriz bidimensional para função.

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

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