VOLTAR

Ponteiros

Apresentar o conceito de ponteiros e aplicá-los utilizando a linguagem C.

NESTE TÓPICO

Introdução

Na maioria das linguagens de programação, as variáveis, quando declaradas, devem ter um identificador (nome) e um tipo de dados. Quando o programa é executado, as variáveis declaradas recebem uma identificação para que elas possam ser localizadas na memória do computador e uma quantidade de bytes.

O identificador para as variáveis são os endereços de memória, que, normalmente, são representados por números hexadecimais e ocupam o primeiro byte alocado para elas. No programa a seguir, são utilizadas duas variáveis inteiras, a e b, e, na figura, é apresentada a simulação da memória do computador quando o programa é executado. Observe que o endereço de memória de a é 3FFA00C0 e o de b é 4FFA00C0.

Simulação da alocação de variáveis na memória do computador

Da mesma maneira que existem variáveis do tipo char, int e float na linguagem C, existem variáveis definidas como ponteiro. Uma variável, quando declarada como ponteiro, proporciona um modo de acesso às variáveis sem referenciá-las diretamente.

Os ponteiros podem ser utilizados para diversas finalidades. Dentre elas, cita-se:

- Subprogramas (funções e procedimentos): modificação de argumentos em subrotinas, quando os parâmetros são passados por referência;
- Alocação dinâmica de memória: permitem a criação de estruturas de dados complexas, tais como listas encadeadas e árvores binárias;

Definição

Um ponteiro é uma variável especial que armazena endereço de memória ao invés de armazenar um dado ou valor. O mecanismo usado para isso é o endereço da variável, sendo o ponteiro a representação simbólica de um endereço. Com isso, é possível acessar o conteúdo de uma variável de forma indireta.

Declaração do Ponteiro

A única diferença na declaração de ponteiros com relação às variáveis mais comuns (int, float e char) está no fato dele armazenar endereço de memória. Para informar que a variável é um ponteiro, basta colocar o símbolo de asterisco (*) ao lado do tipo da variável. Por exemplo, se o tipo é definido como int, então a variável declarada como ponteiro só poderá armazenar endereço de memória para um número inteiro. O símbolo de asterisco é que vai indicar à linguagem C que a variável é um ponteiro e não uma variável comum.

A forma geral de declaração de um ponteiro em linguagem C é:

*tipo *nome-variável;*

Onde, tipo é um tipo qualquer de dados e nome-variável é o nome pelo qual o ponteiro será referenciado.

Exemplos:

- Declaração de um ponteiro denominado px, que armazenará o endereço de memória de uma variável do tipo int.

*int *px;*

- Declaração de um ponteiro denominado pc, que armazenará o endereço de memória de uma variável do tipo char.

*char *pc;*

Operadores para Ponteiros

A manipulação de variáveis declaradas como ponteiros se faz por meio da utilização de dois operadores unários, que são os seguintes:

1. * : o operador “asterisco” pode ser utilizado sob 2 formas: na declaração de ponteiros e para acessar o conteúdo da variável que está sendo apontada pelo ponteiro. No exemplo abaixo é mostrada a utilização deste operador.

2. & : este operador permite acessar o endereço de memória de uma variável. Para tanto, ele deve ser utilizado antes do nome da variável.

Exemplo 1

Abaixo é mostrado um exemplo de um programa que possui uma variável inteira x e um ponteiro px, que contém o endereço de memória de x. Na linha 6, a variável px é declarada como um ponteiro, utilizando-se o operador * (asterisco). Na linha 7, a variável px recebe o endereço de memória de x, utilizando-se o operador &. Percebe-se que, na linha 8, para mostrar o endereço de memória de x, é necessário o uso do operador & e o especificador de formato %p. Na linha 9, para exibir o conteúdo da variável que está sendo apontada por px (acesso indireto ao conteúdo de x), é necessário o uso do operador * (asterisco).

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.   main(){
5.       int x = 15;
6.       int *px;
7.       px = &x;
8.       printf ("Endereco de memoria de x = %p", &x);
9.       printf ("Conteudo da variavel x por meio do ponteiro px = %d", *px);
10.      printf ("Conteudo da variavel px = %p", px);
11.      printf ("Endereco de memoria de px = %p", &px);
12.      system ("PAUSE");
13.   }
```

Na Figura abaixo, é mostrada uma simulação do programa acima. Quando o programa for executado, irão aparecer as seguintes mensagens na tela: “Endereco de memória de x = 3FFA00C0”, “Conteudo da variável x por meio do ponteiro px = 15”, ”Conteudo da variável px = 3FFA00C0” e “Endereco de memória de px=4FFA00C0”.

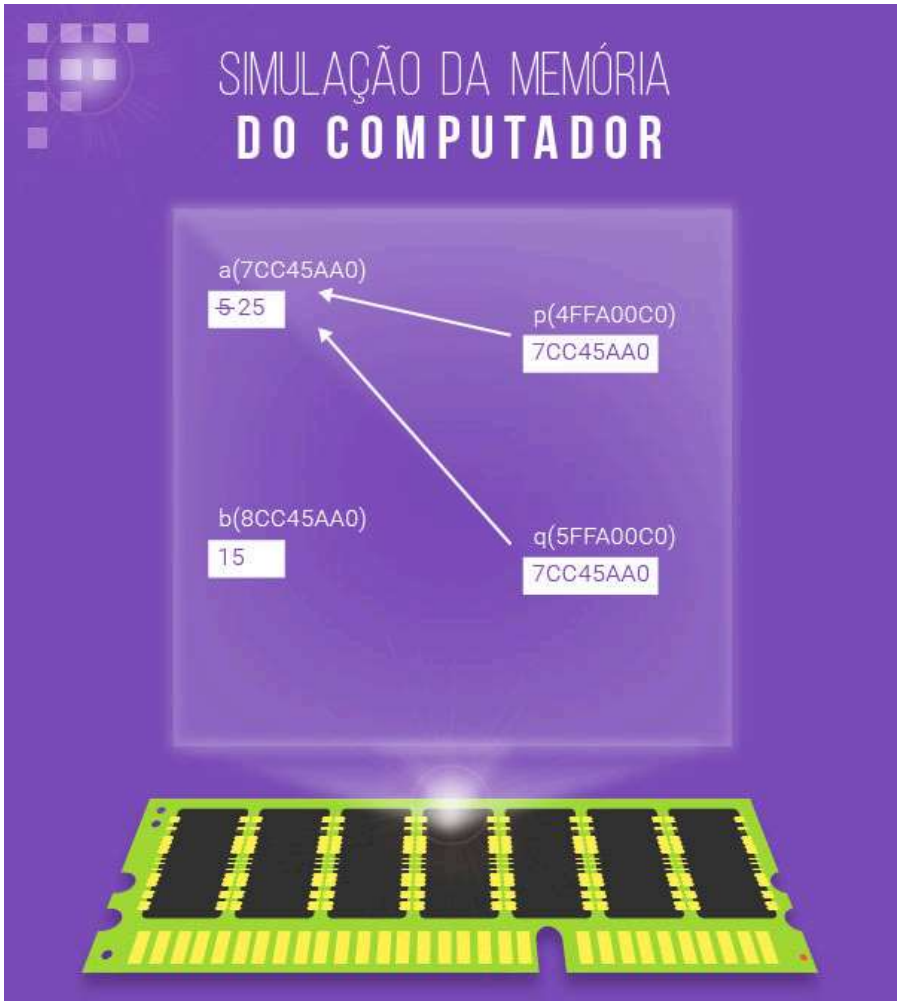
Simulação da alocação de variáveis na memória do computador

Exemplo 2

A seguir, apresentamos um exemplo completo de um programa com as variáveis p e q como ponteiros para número inteiro. Observe que, na linha 10, é feita uma atribuição à variável q, que passa a ter o mesmo valor (endereço de memória de a) armazenado em p. Na linha 11, ao modificarmos o conteúdo apontado por q, o valor armazenado em a será alterado, visto que a variável q também aponta para a variável a.

```
1.  #include <stdio.h>
2.   #include <stdlib.h>
3.
4.   main(){
5.        int a, b;
6.        int *p, *q;
7.        a = 5;
8.        b = 15;
9.        p = &a;
10.       q = p;
11.      *q = 25;
12.       printf ("Conteudo da variavel a = %d", a);
13.       system ("PAUSE");
14.    }
```

Na Figura abaixo, é mostrada uma simulação do programa acima. Quando o programa for executado, irá aparecer a seguinte mensagem na tela: “Conteudo da variavel a = 25”.



Simulação da alocação de variáveis na memória do computador

Exemplo 3

O programa abaixo contém as variáveis pa e pb como ponteiros para número real (float). Observe que, na linha 11, ocorre a atribuição da soma dos valores das variáveis a e b à variável r, de forma indireta, por meio da utilização dos ponteiros.

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  main(){
5.      float a, b, r;
6.      float *pa, *pb;
7.      a = 7.3;
8.      b = 11.5;
9.      pa = &a;
10.     pb = &b;
11.     r = *pa + *pb;
12.     printf ("Soma de a e b via ponteiros = %f", r);
13.     system ("PAUSE");
14.   }
```

Na Figura abaixo, é mostrada uma simulação do programa acima. Quando o programa for executado, irá aparecer a seguinte mensagem na tela: “Soma de a e b via ponteiros = 18.8”.

Simulação da alocação de variáveis na memória do computador

Exemplo 4

Neste programa, uma variável ponteiro armazena o endereço de memória de uma variável *n* do tipo *int.* Em seguida, é feita a divisão de n por 5, sem utilizar o valor de n diretamente.

```c
#include <stdio.h>
#include <stdlib.h>
main () {
    int n = 5;
    int *pn;
    pn = &n;
    *pn = *pn / 5;
    printf ("\n Valor de n = %d", *pn);
    system ("PAUSE");
}
```

Agora que você já estudou essa aula acesse a plataforma AVA, resolva os exercícios e verifique o seu conhecimento. Caso fique alguma dúvida, leve a questão ao Fórum e divida com seus colegas e professor.

Quiz

Exercício

Ponteiros

INICIAR

Quiz

Exercício Final

Ponteiros

INICIAR

Referências

MIZRAHI, V. V. Treinamento em linguagem C. São Paulo: Pearson, 2008.

SCHILDT, H. C – Completo e Total. São Paulo: Pearson, 2006.

Avalie este tópico

ANTERIOR
Strings

Índice

PRÓXIMO
Alocação dinâmica de memória

Biblioteca (https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove (http://www.uninove.br)
Mapa do Site

Ajuda? (https://ava.uninove.br/seu/AVA/topico/topico.php?idCurso=)