Linguagem de Programação I

PONTEIROS

PONTEIROS

- Um ponteiro é uma variável que contém um endereço de memória.
- Ele pode armazenar o endereço de memória de outra variável.
- O valor do endereço de memória não é escolhido pelo programador.
- O programa, em execução, é que se encarrega de achar uma posição de memória livre no computador, e a reserva para uso.

Endereço na Memória	Conteúdo na Memória
1000	
1001	1100
...	
...	
1100	'A'
1101	

DECLARAÇÃO E UTILIZAÇÃO

Para declarar e utilizar um ponteiro usa-se uma simbologia apropriada.

- O símbolo ***** declara um ponteiro e também serve para referenciar um conteúdo.
- O símbolo **&** se refere ao endereço de uma variável.

A declaração é dada por: **tipo *** *nome_da_variável*;

DECLARAÇÃO E UTILIZAÇÃO

Exemplo:

int *vponteiro
- significa que a variável ponteiro pode conter o endereço de memória de um dado do tipo int.

vponteiro=&vnumero;
- & pode-se ler como: "o endereço de memória de"
- estamos atribuindo para a variável vponteiro o endereço de memória de onde fica a variável vnumero.

printf("%d", *vponteiro);
- pode-se ler como: " conteúdo (ou dado) apontado por "
- indica que estamos nos referenciando *indiretamente* à variável numero.

PARA QUE SERVE PONTEIRO?

Em situações práticas, utilizamos ponteiros para:

- tratar variáveis alocadas dinamicamente
- receber parâmetros que foram passados por referência.
- criar estruturas de dados complexas, como listas encadeadas e árvores binárias;
- alocar e desalocar memória dinamicamente do sistema, etc.

EXEMPLO

```
#include <stdio.h>

int main()
{
   int vNumero;
   int *vPonteiro;

   vPonteiro = &vNumero;
   vNumero = 25;

   printf("Conteudo da variavel vNumero.............%d\n",vNumero);
   printf("Conteudo da variavel vPonteiro..........%p\n",vPonteiro);
   printf("Endereco de memoria onde esta vNumero.....%p\n",&vNumero);
   printf("Endereco de memoria onde esta vPonteiro %p\n",&vPonteiro);
   printf("Conteudo da var apontada por vPonteiro..%d\n",*vPonteiro);
   return 0;
}
```

Qual o resultado?

EXEMPLO

```
Conteudo da variavel vNumero.................25
Conteudo da variavel vPonteiro...............0x22cd34
Endereco de memoria onde esta vNumero........0x22cd34
Endereco de memoria onde esta vPonteiro......0x22cd30
Conteudo da variavel apontada por vPonteiro..25
```

EXEMPLO

```
#include <stdio.h>

int main ()
{
  int num,*p;
  num=55;
  p=&num;
  printf ("\nValor inicial: %d\n",num);
  *p=100;
  printf ("\nValor final: %d\n",num);

  return(0);
}
```

Resultado:

Valor Inicial: 55

Valor Final: 100

ARITMÉTICA DE PONTEIRO

Podem ser usadas duas operações com ponteiros: adição e subtração.

Considere ***p1*** um ponteiro para um inteiro com um valor atual 2000. Assuma também que os inteiros são de 2 bytes. Após a expressão

p1++

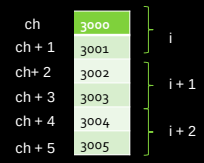
p1 contém 2002, pois ele aponta para o próximo inteiro.

O mesmo acontece com a subtração

ARITMÉTICA DE PONTEIRO

```
char *ch;

int *i;
```



TODA ARITMÉTICA DE PONTEIROS É RELATIVA AO SEU TIPO BASE

PONTEIROS E VETORES

Considere os comandos abaixo:

```
int vetor[10];
int *ponteiro, i;
ponteiro = &i;

/* as operações a seguir são inválidas */
vetor = vetor + 2;
vetor++;
vetor = ponteiro;

/* as operações abaixo são válidas */
ponteiro = vetor;
ponteiro = vetor+2;
```

PONTEIROS E MATRIZES

Considere o exemplo:

```
int main()
{
   int vet[4]={20,30,40,50};

   printf("%i",*vet);
   return 0;
}
```

O *"nome da variável" que você declarou é na verdade um ponteiro para o tipo da variável da matriz.*
Este conceito é fundamental.
Eis porque: Tendo alocado na memória o espaço para a matriz, ele toma o nome da variável (que é um ponteiro) e aponta para o *primeiro elemento da* matriz.

PONTEIROS E MATRIZES

Qualquer operação que possa ser feita com índices de uma matriz pode ser feita com ponteiros.

O nome da matriz representa um endereço de memória. Este endereço é o primeiro elemento da matriz.

Exemplo com índice:

```
#include <stdio.h>

int main()
{
   int M[5]={92,81,70,69,58};
   int i;
   for (i=0;i<5;i++)
   {
      printf("%d\n", M[i]);  // notação matriz
   }
   return 0;
}
```

PONTEIROS E MATRIZES

Usando notação ponteiro:

```
#include <stdio.h>

int main()
{
   int M[5]={92,81,70,69,58};
   int i;
   for (i=0;i<5;i++)
   {
      printf("%d\n", *(M+i));
   }
   return 0;
}
```

PONTEIROS E MATRIZES

Em regra geral temos que:

M + i é equivalente a &M[i], portanto

***(M + i) é equivalente a M[i]**

PONTEIROS CONSTANTES E VARIÁVEIS

O nome de uma matriz é um *ponteiro constante*, portanto a expressão M++ (no exemplo anterior) não poderia ser usada, mas considere o exemplo abaixo:

```
int main()
{
   int M[5]={92,81,70,69,58};
   int i, *p=M; // cria e inicializa um PONTEIRO VARIÁVEL
   for (i=0;i<5;i++)
   {
      printf("%d\n", *(p++)); // OPERAÇÕES PODEM SER
                              //REALIZADAS COM VARIÁVEIS
   }
   return 0;
}
```

MATRIZES DE PONTEIROS

Podemos construir matrizes de ponteiros como declaramos matrizes de qualquer outro tipo.

Uma declaração de uma matriz de ponteiros inteiros poderia ser:

int *pmatrx [10];

No caso acima, **pmatrx é um vetor que armazena 10 ponteiros para inteiros.**

REFERÊNCIA BIBLIOGRÁFICA

Mizrahi, Victorine Viviane. Treinamento em Linguagem C. São Paulo: Editora Pearson, 2008, 2ª edição.

Schildt, Hebert. C completo e Total. São Paulo: Makron Books, 1996.

W. Celes e J. L. Rangel, Vetores e Alocação Dinâmica. Puc-Rio. Apostila.