UFRJ

INSTITUTO DE COMPUTAÇÃO UFRJ

Programação de Computadores II

Ponteiros e Funções

Profa. Giseli Rabello Lopes

Sumário

- Ponteiros
- Funções
  - Passagem de parâmetros por referência
- Exemplos
- Exercícios

Adaptado de slides preparados por Profa. Silvana Rosseto e Profa. Cora H. F. Pinto Ribeiro

Ponteiros e Vetores

- Declaração de vetor:
  - O compilador automaticamente reserva um bloco de memória para que o vetor seja armazenado, do tamanho especificado na declaração (mas **não controla** se o uso é limitado à área reservada).
  - O nome de um vetor também pode ser usado como sendo um ponteiro para o primeiro elemento do vetor (contém o endereço do 1º elemento).
- Vetores como parâmetros de função:
  - Vetores são sempre passados por referência para funções!

Passagem de vetor como parâmetro

```
tipo nome[] ou tipo *nome
```

- **tipo**: corresponde ao tipo dos elementos do vetor
- **nome**: é o nome atribuído ao vetor
- **[]**: indica que a variável é do tipo vetor
  - Pode ser utilizado sem um valor, pois em C não interessa qual a dimensão do vetor que é passado a uma função, mas sim o tipo dos seus elementos
- *: indica que é um ponteiro para o tipo primitivo de dado do vetor

Exemplo 3

- Crie uma função **`inic()`** que inicializa os elementos de um vetor com zeros e faça um programa principal que teste a mesma.

```
#include <stdio.h>
#define DIM 10
void inic(int s[], int n) //ou int *s ou int s[DIM]
{
     int i;
     for (i=0; i<n; i++)
          s[i]=0; //ou *(s+i)=0;
}
int main()
{
     int v[DIM], i;
     inic(v,DIM);
     for(i=0; i<DIM; i++)
          printf("%do. elemento = %d \n",i+1,v[i]);
     return 0;
}
```

Exemplo de implementação da função `strcpy`

- A linguagem C não provê operadores para processar *strings* diretamente (a biblioteca **`string.h`** oferece um conjunto de funções para manipulação de *strings*)
- Tomaremos como exemplo a função **`strcpy(dest, fonte)`**: copia caracter-a-caracter a *string* **`fonte`** para a *string* **`dest`**
- Veremos diferentes possibilidades de implementação dessa função

Implementação de **`strcpy`** usando vetores

```c
#include <stdio.h>

void strcpy1 (char *dest, char *fonte)
{
   int i = 0;
   while ((dest[i] = fonte[i]) != '\0')
      i++;
}

int main()
{
  char a[100], b[] = "Ola mundo";
  strcpy1(a, b);
  printf("a: %s\nb: %s\n", a, b);
  return 0;
}
```

```
a: Ola mundo
b: Ola mundo
```

Vetor como argumento de função

- Quando um vetor é passado como argumento para uma função, o que é passado é uma cópia privada da localização do primeiro elemento

Implementação de **`strcpy`** usando ponteiros

```
void strcpy2 (char *dest, char *fonte)
{
   while ((*dest = *fonte) != '\0')
   {
      dest++;
      fonte++;
   }
}

void strcpy3 (char *dest, char *fonte)
{
   while ((*dest++ = *fonte++) != '\0');
}

void strcpy4 (char *dest, char *fonte)
{
   while (*dest++ = *fonte++);
}
```

Passagem de matriz como parâmetro

- A passagem de vetores com mais de uma dimensão para uma função é realizada indicando no cabeçalho desta, obrigatoriamente, o número de elementos de cada um das n-1 dimensões à direita.
- Apenas a dimensão mais à esquerda pode ser omitida, colocando-se [].

Exemplo

- Construir uma função **`inicM()`** que inicializa uma matriz quadrada com 9 elementos e outra função **`mostraM()`** que mostra os elementos dessa matriz.

```c
#include <stdio.h>
#define DIM 3
void inicM(int s[ ][DIM])
{
    int i, j;
    for (i=0;i<DIM;i++)
      for (j=0; j <DIM;j++)
         s[i][j] = 0;
}
void mostraM(int s[DIM][DIM])
{
    int i, j;
    for (i=0; i<DIM;i++)
    {
       for (j=0;j<DIM;j++)
         printf(" [%d] ",s[i][j]);
      printf("\n");
    }
}
int main()
{
    int s[DIM][DIM];
    inicM(s);
    mostraM(s);
    return 0;
}
```

Vetores para ponteiros:
`*(s[i]+j)`

Ponteiros para ponteiros:
`*(*(s+i)+j)`

Saída:

```
[0] [0] [0]
[0] [0] [0]
[0] [0] [0]
```