INE 5416/5636 - Paradigmas de programação

Turmas 04208/08238

Prof. Dr. João Dovicchi – dovicchi@inf.ufsc.br

http://www.inf.ufsc.br/~dovicchi

Mais listas

Listas Conjunto ordenado de dados, geralmente do mesmo tipo,

que podem ser operados de forma individual ou coletiva em um processo.

Mais listas

Listas Conjunto ordenado de dados, geralmente do mesmo tipo, que podem ser operados de forma individual ou coletiva em um processo.

Exemplos:

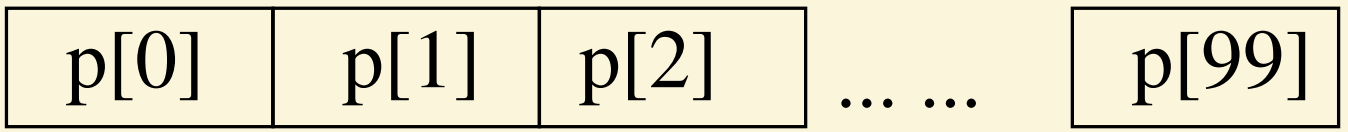
```
(1,3,5,30,12, 18, 27)
(a,b,c,d,e,f,g,h)
(banana, abacaxi, maçã, pera, uva)
(Antônio, Manuel, Alice, Marta, José)
```

Mais listas

Listas (Array):

```
int i, p[100];
    for(i=0; i<=99; i++){
       p[i] = 10*i;
    }
...
```

as variáveis são alocadas em posições consecitivas, na memória.



Mais listas

Listas (Array):

- Elementos declarados como `tipo nome[indice]`, onde `tipo` é qualquer tipo válido em C, `nome` é um nome válido de variável e `indice` é um valor inteiro.
- Índices vão de `0` a `indice - 1`
- A alocação de memória é estática e contígua
- C não detecta índice fora dos limites do array em tempo de execução.

Array

- Array unidimensional ou vetor:

  Ex. `var[10]` $\rightarrow$ um vetor de 10 elementos.

Mais listas

Array

- Array unidimensional ou vetor:

  Ex. `var[10]` $\rightarrow$ um vetor de 10 elementos.

- Array bidimensional ou matriz:

  Ex. `var[3][4]` $\rightarrow$ uma matriz $M_{3,4}$

Mais listas

Array

- Array unidimensional ou vetor:

  Ex. `var[10]` $\rightarrow$ um vetor de 10 elementos.

- Array bidimensional ou matriz:

  Ex. `var[3][4]` $\rightarrow$ uma matriz $M_{3,4}$

- Array multidimensional... etc.

Nota: C não tem array bi, tri ou multidimensional como estrutura. Na verdade podemos criá-los como arrays de array.

Array

O vetor mais comum em C é um vetor de caracteres, terminado por um nulo: uma cadeia (*string*).

```
...
    char ch[10];
    char *cp;
    cp = ch; // equiv. cp = &ch[0];
...
    cp = &ch[9]; // *cp = '\0'
...
    cp = &ch[10]; // cp aponta para fora do array!
...
```

Array

O vetor mais comum em C é um vetor de caracteres, terminado por um nulo: uma cadeia (*string*).

```
...
    char ch[10];
    char *cp;
    cp = ch; // equiv. cp = &ch[0];
...
    cp = &ch[9]; // *cp = '\0'
...
    cp = &ch[10]; // cp aponta para fora do array!
...
```

o tamanho de um array depende do número de elementos e do tipo:

```
tamanho = sizeof(tipo) * tamanho-do-array
```

Vetores, Matrizes e multidimensionais

Um vetor pode ser declarado como um arranjo de elementos ou como os vemos matematicamente.

$$\vec{v_i} = [v_1, v_2, \ldots, v_n]$$

ou

$$\vec{v_i} = [v_1, v_2, \ldots, v_n]^T$$

Vetores, Matrizes e multidimensionais

No primeiro caso (um vetor linha):

$$\vec{v_i} = [v_1, v_2, \ldots, v_n]$$

pode ser declarado como: `<tipo> v[1][n]`

No segundo caso (um vetor coluna):

$$\vec{v_i} = [v_1, v_2, \ldots, v_n]^T$$

pode ser declarado como: `<tipo> v[n][1]`

Vetores, matrizes e multidimensionais

Uma matriz pode ser declarada como um array de array. Por exemplo, seja a matriz M:

$$M_{(3\times3)} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3.3} \end{pmatrix}$$

onde cada elemento $a_{i,j}$, da matriz é um inteiro. Pode-se declarar, em C:

```
...
int a[3][3];
...
```

onde cada linha é um elemento do vetor e o mesmo vale para outras dimensões.

Array e argumento de funções

Arrays são passados para funções por meio de apontadores.

```
#include <stdio.h>

void printarray (int arg[], int length) {
  int n;
  for (n=0; n<length; n++){
    printf("%d\t",arg[n]);
  }
  printf("\n");
}

int main (){
  int firstarray[] = {5, 10, 15};
  int secondarray[] = {2, 4, 6, 8, 10};
  printarray (firstarray,3);
  printarray (secondarray,5);
  return 0;
}
```

Prática

Roteiro Prática 9