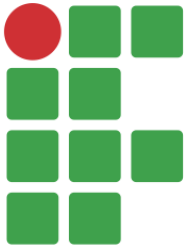
ARRAYS



INSTITUTO FEDERAL
Paraíba

Professor: Paulo de Tarso F. Júnior

paulodt@gmail.com

Roteiro

- Introdução
- *Arrays*
- Declaração de *Arrays*
- Tópicos
- Exemplos de Uso de *Arrays*
- Passagem de *Arrays* para Funções
- *Arrays* com Vários Subscritos

Introdução

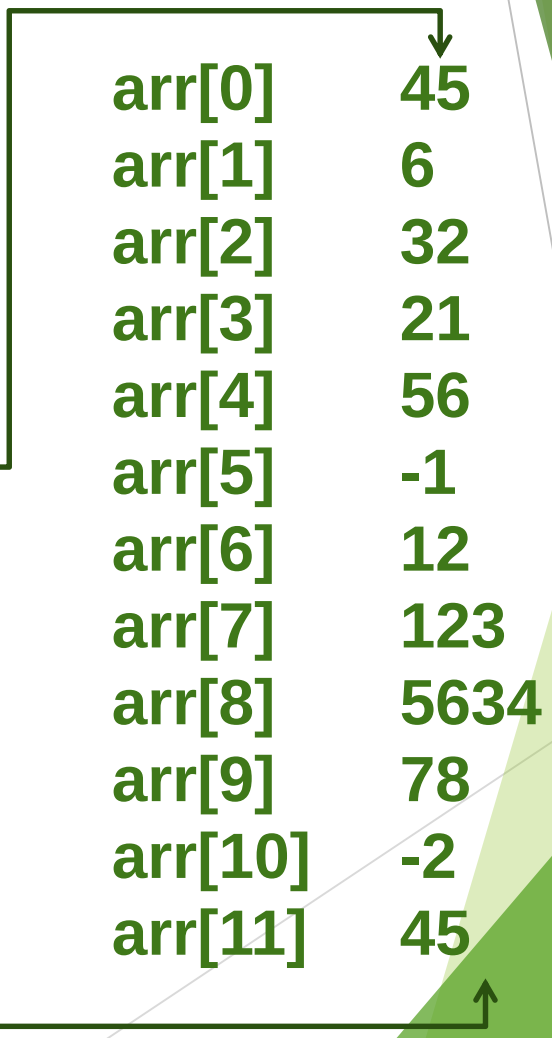
- *Arrays*
  - Estruturas de itens de dados relacionados
  - Entidade estática → Tamanho constante ao longo de todo o programa
  - Estruturas de dados dinâmicas → Capítulo 12 (malloc, calloc)
  - Grupo de locações consecutivas de memória
  - Mesmo nome e tipo

Arrays

- Nome do *array* ***arr***
- Número de elementos 12

Nome do array
Todos os elementos do array têm o mesmo nome, vet

Posição do elemento
Número que indica a posição do elemento no array acompanha o nome, entre colchetes



Arrays

- Referência a elementos de um *array*
  - Nome do *array*
  - Número da posição do elemento no *array*
- Formato
    - **nome_array[número_posição]**
- Primeiro elemento → Posição 0 (zero) do *array*
- Array **c** de **n** elementos
    - **c[0],c[1] ...c[n - 1]**

Arrays

- Elementos de *array são semelhantes a variáveis* normais
  - c[0] = 3;
  - printf("%d", c[0]);
- Realização de operações em subscritos. Se **x** igual a **3**
  - c[5 - 2] == c[3] == c[x]

Declaração de Arrays

- Declaração de *arrays*
  - Nome
  - Tipo
  - Número de elementos
    - **tipo_array nome_array[número_elementos];**
  - Exemplos
    - **int c[10];**
    - **float meu_array[3284];**
- Declaração de múltiplos *arrays do mesmo tipo*
  - Formato similar para variáveis regulares
  - Exemplo
    - **int b[100], x[27];**

Exemplo do Uso de Arrays

- Inicializadores
  - **int n[5] = { 1, 2, 3, 4, 5 };**
  - Número de inicializadores ***insuficiente*** → *Atribuição de 0* aos elementos mais à direita
    - **int n[5] = {0}**
    - **Todos os elementos iguais a 0**
  - Número de inicializadores ***excessivo*** → *Produção de um* erro de sintaxe
  - *Arrays em C **não** têm verificação de limites*

Exemplo do Uso de Arrays

- Omissão do tamanho → Determinação a partir dos inicializadores
  - **int n[ ] = { 1, 2, 3, 4, 5 };**
  - 5 inicializadores → *Array com 5 elementos*

Exemplo do Uso de Arrays

```
/*Programa de impressão de histograma */
#include <stdio.h>

int main()
{
int n[10] = { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 };
int i, j;
printf( "%s%13s%17s\n", "Elemento", "Valor", "Histograma" );
for ( i = 0; i <= TAM - 1; i++ ) {
    printf( "%7d%13d ", i, n[ i ]) ;
    for ( j = 1; j <= n[ i ]; j++ ) /* imprime uma barra */
        printf( "%c", '*' );
        printf( "\n" );
    }
    return 0;
}
```

Exemplo do Uso de Arrays

Elemento	Valor	Histograma
0	19	**********
1	3	***
2	15	***************
3	7	*******
4	11	**********
5	9	*********
6	13	*************
7	5	*****
8	17	**********
9	1	*

Exemplo do Uso de Arrays

- *Arrays de caracteres*
  - Cadeia de caracteres **primeiro** →*Array estático*
  - Possibilidade de inicialização de *arrays de caracteres a* partir do uso de literais do tipo cadeia de caracteres (*string)*
    - **char string1[ ] = “primeiro”;**
    - Terminação de cadeias de caracteres NULL (\0)
    - string1 é realmente composta por 9 elementos
    - **char string1[]={'p','r','i','m','e','i','r','o','\0' };**
  - Possibilidade de acesso a caracteres individuais
    - **string1[3] → Caractere ‘m’**

Exemplo do Uso de Arrays

- Nome do *array* → *Endereço do array*
  - Uso de **“&”** desnecessário
    - **scanf("%s", string2);**
    - Leitura de caracteres até a identificação de um espaço em branco
  - Possibilidade de escrita além do *array* → *Atenção e cuidado*

Passagem de *Arrays* para Funções

- Passagem de *arrays para funções*
  - Passagem de argumentos do tipo *array* → *Especificação do* nome do *array (sem colchetes)*

    Declaração: tipo array[tamanho];

    Protótipo da função: tipo funcao(tipo [ ])

    Chamada: funcao(array);

    Programa da função: tipo funcao(tipo array[ ]);
  - Tamanho do *array usualmente passado para a função*
- *Arrays são passados por referência*
- Nome do *array* → *Endereço do primeiro elemento*
- Conhecimento do espaço de armazenamento do *array pela* função
  - Modificação das locações originais de memória

Passagem de *Arrays* para Funções

- Passagem de elementos do *array*
  - Passagem por valor
  - Passagem do nome do *array com subscrito para a função (e.g.* **array[3]**)
- Protótipo da Função

  void modificaArray(int b[ ] , int tamanho);
- Nomes de parâmetros → Opcionais no protótipo
  - int b[ ] → Possibilidade de escrita como int [ ]
  - int tamanho → int

Passagem de *Arrays* para Funções

```
01  /* Passagem de arrays e elementos isolados de arrays para funções */
02  #include <stdio.h>
03  #define TAM      5
04
05  void modificaArray(int [], int);
06  void modificaElemento(int);
07  int main()
08  {
09     int a[TAM] = {0, 1, 2, 3, 4}, i;
10
11     printf( "Efeitos da passagem de arrays inteiro em chamadas por referência: "
12             "\nOs valores do array original são:\n "
13     for (i = 0; i <= TAM - 1; i++)
14        printf( "%3d", a[ i ] );
15     printf( "\n" );
16     modificaArray(a, TAM);  /* passagem por referência */
17     printf( "Os valores do array modificado são:\n" );
18     for (i = 0; i <= TAM - 1; i++)
19        printf( "%3d", a[ i ] );
20     printf( "\ Efeitos da passagem de elementos em chamadas por valor:"
21             "\nO valor de a[3] é %d\n", a[3] );
22     modificaElemento(a[ 3 ]);
23     printf( "O valor de a[3] é %d\n", a[3] );
24     return 0;
25  }
```

Arrays inteiros passados por referência podem ser modificados

Elementos de array passados por valor não podem ser modificados

Passagem de *Arrays* para Funções

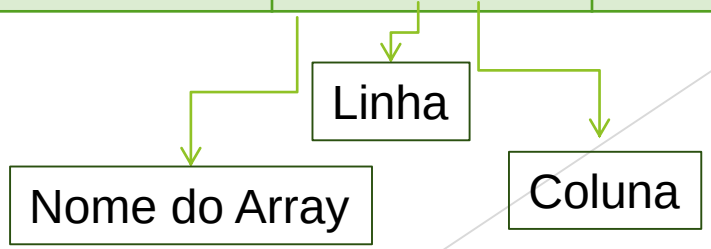
```
void modificaArray(int b[], int tam)
{
int j;
for (j = 0; j <= tam - 1; j++)
b[ j ] *= 2;
}
void modificaElemento(int e)
{
printf("Valor em modificaElemento é
%d\n", e *= 2);
}
```

Efeitos da passagem de arrays inteiros em chamadas por referência:
Os valores do array original são:
0 1 2 3 4
Os valores do array modificado são:
0 2 4 6 8
Efeitos da passagem de elementos em chamadas por valor:
O valor de a[3] é 6
Valor em modificaElemento é 12
O valor de a[3] é 6

Arrays com vários Subscritos

- Tabelas com linhas e colunas (*arraym x n)*
- Matrizes → Especificação da linha e, em seguida, da coluna

	Coluna 0	**Coluna 1**	**Coluna 2**
Linha 0	mat[0][0]	mat[0][1]	mat[0][2]
Linha 1	mat[1][0]	mat[1][1]	mat[1][2]
Linha 2	mat[2][0]	mat[2][1]	mat[2][2]



Arrays com vários Subscritos

- Inicialização
  - int b[2][2] = { { 1, 2 }, { 3, 4 } };
  - Agrupamento de inicializadores → Linhas entre chaves
  - Se não forem suficientes → Elementos não especificados ajustados para zero

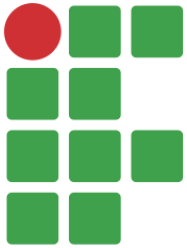
    int b[2][2] = { { 1 }, { 3, 4 } };
- Referência de elementos
  - Especificação da linha e, em seguida, da coluna

    printf("%d", b[0][1]);

Exercício - Cálculo da Média, Mediana e Moda

- Escrever um programa que calcule a Média, Mediana e Moda de uma lista
  - Média → Média aritmética
  - Mediana → Valor central de uma sequência ordenada
    - Exemplo: ***2 3 4 6 7*** Mediana = *4*
  - Moda → Valor mais frequente em uma sequencia
    - Exemplo: ***1 1 1 2 3 3 4 5 6 5*** Moda = *1*

ARRAYS



INSTITUTO FEDERAL
Paraíba

Professor: Paulo de Tarso F. Júnior

paulodt@gmail.com