

# Linguagem C:

## Tipos básicos de dados escalares e cadeia de caracteres

# Sumário

- Dados Escalares;
- Strings:
  - Strings Constantes;
  - Strings Variáveis;
  - A função gets();
  - A função strcpy();
  - A função strcat();
  - A função strlen();
  - A função strcmp();
- Matriz Unidirecional;
- Matriz Bidimensional;

# Dados Escalares

- Os **cinco** tipos básicos de dados em C são<sup>\*</sup>:

TIPO	BIT	BYTES	ESCALAS
char	8	1	-128 a 127
int	16	2	-32768 a 32767
float	32	4	3.4E-38 a 3.4E+38
double	64	8	1.7E-308 a 1.7E+308
void	0	0	Nenhum valor

- Dados no formato inteiro(int) ou caracter(char) são mais fáceis de serem trabalhados pois são dados inteiros.

<sup>\*</sup>Tipos vistos na Aula 02

# Strings

- String é uma **coleção** (matriz unidimensional) de **caracteres** e sempre é terminada pelo caractere zero ('\0');

J	o	ã	o	'\0'
---	---	---	---	------

- A declaração geral para uma string em C é:  
`char nome_da_string [tamanho];`
- Devemos ficar **atentos** para o fato de que as strings têm seu último elemento como um '\0' (Null).

# Strings

- Caso você necessite guardar 10 caracteres em uma string a declaração deve ser feita da seguinte forma:

**char str[11];**

- Isso reserva espaço nulo no final da string.
- Embora C não tenha o tipo de dado **string**, ela permite constantes **string**;
- Uma constante string é uma lista de caracteres entre aspas;

Ex: "Engenharia"

# String Constantes

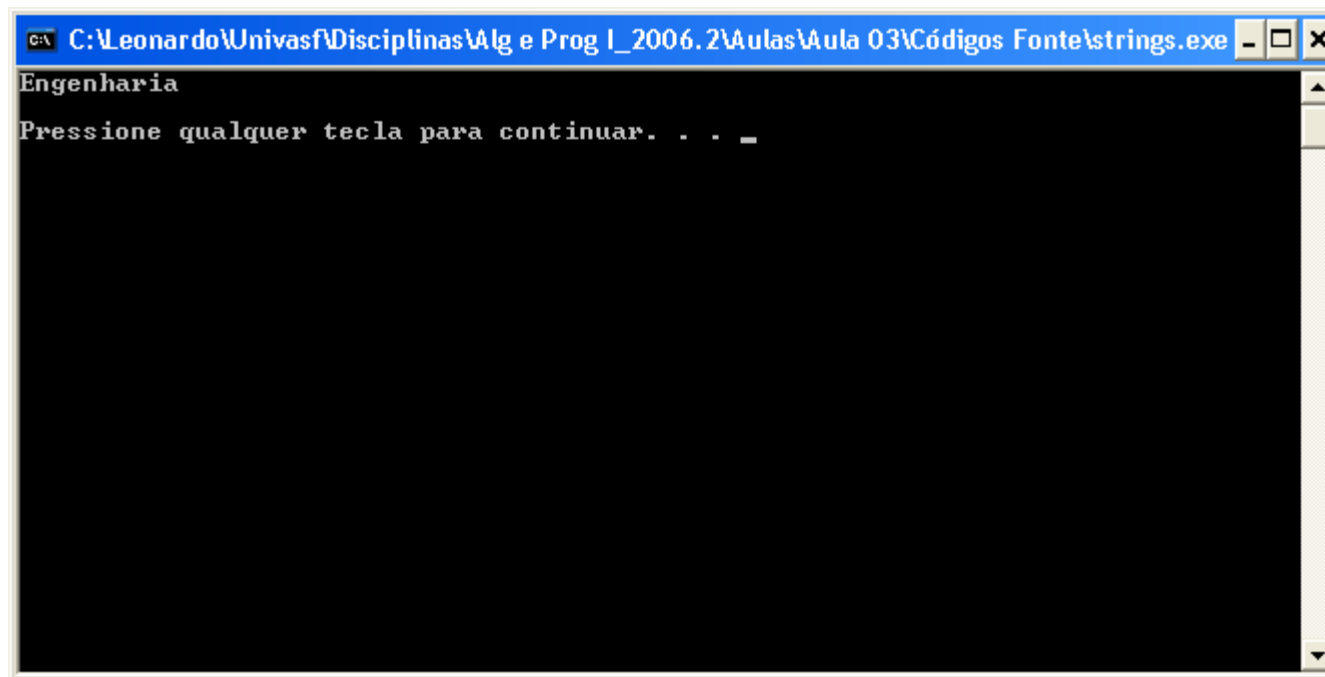
- Não é preciso adicionar o nulo no final das constantes string manualmente - o compilador C faz isso automaticamente, vejamos o programa abaixo:

```
#include <stdio.h>
/* Este programa testa o tratamento dado às strings constantes */

int main()
{
    char str[11]="Engenharia";
    printf("%s\n\n", str);
    system("pause");
    return 0;
}
```

# String Constantes

- A saída no console para o programa anterior será:



The screenshot shows a Windows console window with a blue title bar. The title bar text is "C:\Leonardo\Univasf\Disciplinas\Alg e Prog I\_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe". The console area has a black background and white text. The first line of output is "Engenharia". The second line is "Pressione qualquer tecla para continuar. . . \_", where the underscore represents a cursor. The window has standard Windows controls (minimize, maximize, close) in the top right corner.

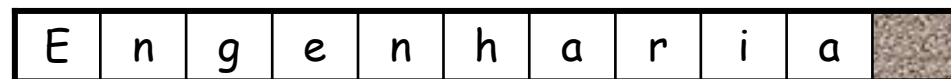
```
C:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe
Engenharia
Pressione qualquer tecla para continuar. . . _
```

# String Constantes

- O que ocorrerá caso uma string constante extrapole o espaço de memória reservado na declaração?

```
#include<stdio.h>
/* Este programa testa o tratamento dado às strings constantes */

int main()
{
    char str[11]="Engenharia Civil";
    printf("%s\n\n", str);
    system("pause");
    return 0;
}
```

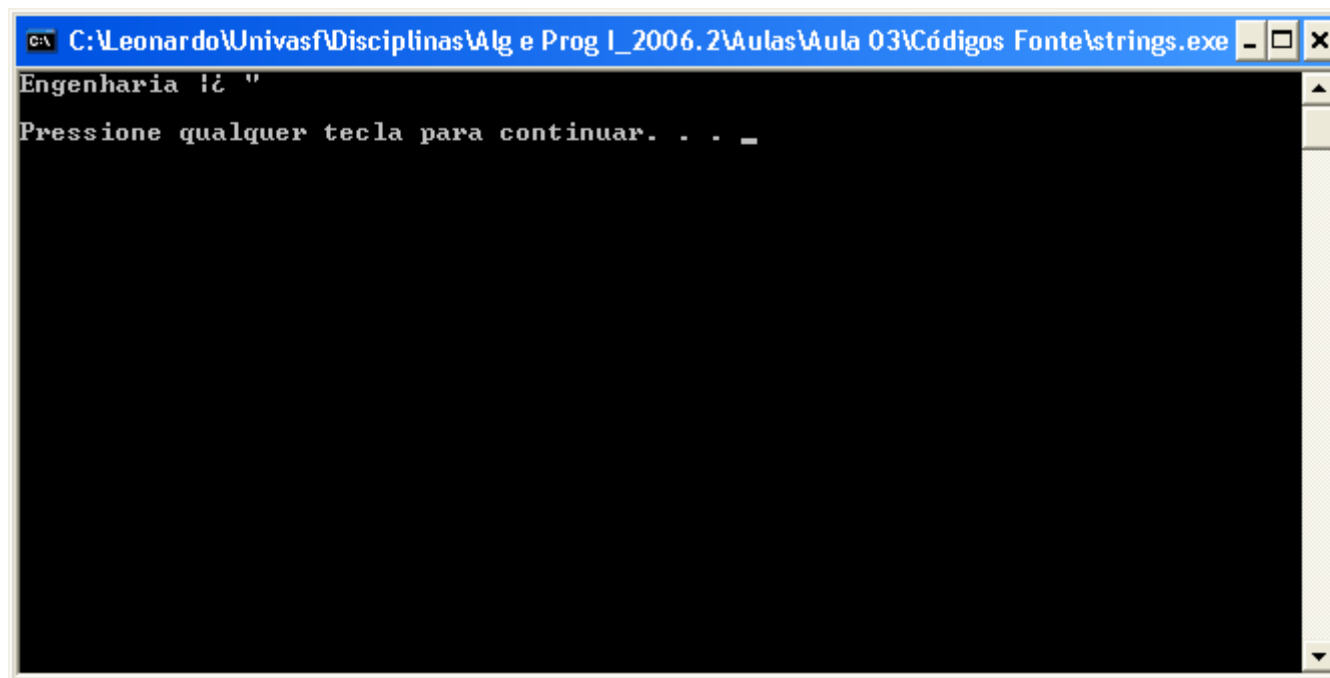


Lixo de memória



# Strings Constantes

- Leitura de uma região de memória não prevista na declaração da variável



```
C:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe
Engenharia 12 "
Pressione qualquer tecla para continuar. . . _
```

# Strings Variáveis

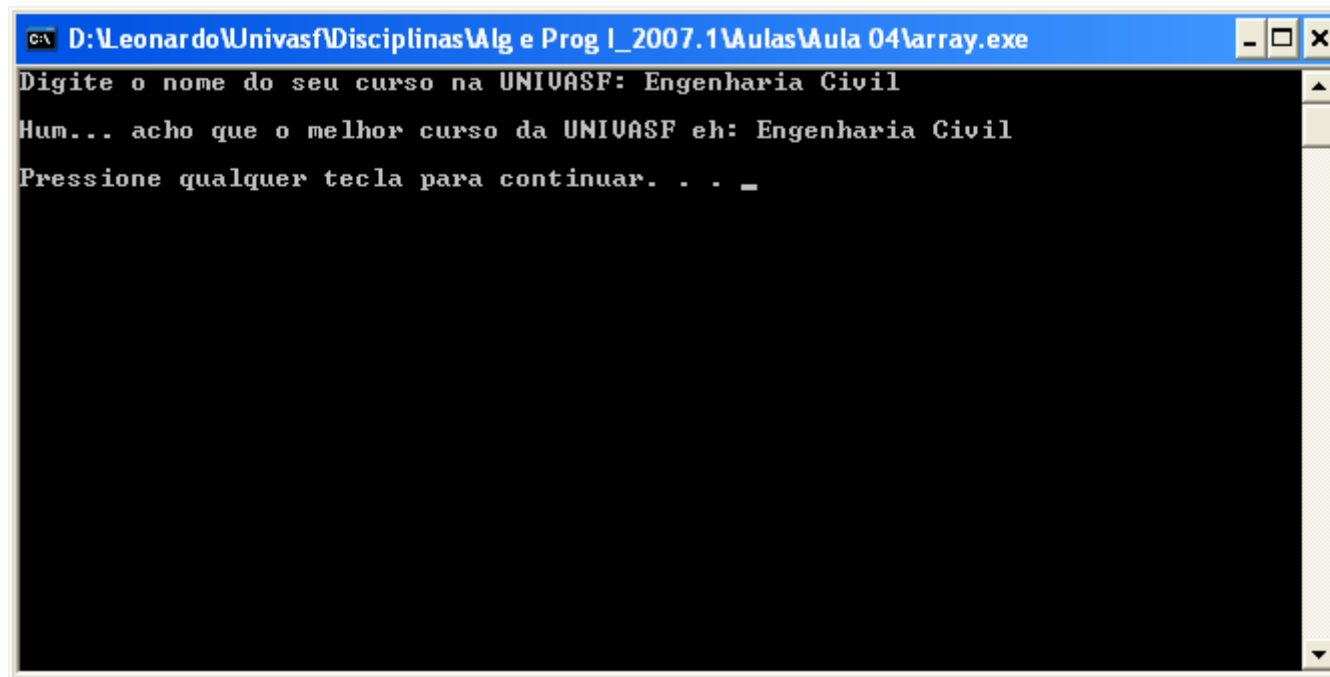
- Nas strings variáveis o caracter '\0' é inserido automaticamente no final da entrada, veja:

```
#include <stdio.h>
/* Este programa testa o tratamento dado às strings variáveis */

int main()
{
    char str[30];
    printf("\nDigite o nome de seu curso na UNIVASF: ");
    gets(str);
    printf("\nHum... acho que o melhor curso da UNIVASF eh: %s\n\n", str);
    system("pause");
    return 0;
}
```

# Strings Variáveis

- A saída no console para o programa anterior será:



```
C:\D:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2007.1\Aulas\Aula 04\array.exe
Digite o nome do seu curso na UNIVASF: Engenharia Civil
Hum... acho que o melhor curso da UNIVASF eh: Engenharia Civil
Pressione qualquer tecla para continuar. . . _
```

# A função gets()

- A função gets() lê uma string do teclado. Sua forma geral é:

**gets()** (nome\_da\_string);

- O programa abaixo demonstra o funcionamento da função gets():

```
#include <stdio.h>
/* Este programa demonstra o funcionamento da função gets() */

int main()
{
    char str[30];
    printf("\nDigite seu nome: ");
    /* Função responsável por ler uma string do teclado */
    gets(str);
    printf("\nOla %s\n\n", str);
    system("pause");
    return 0;
}
```

# Strings

- C suporta uma ampla gama de funções de manipulação de strings. As mais comuns são:

Nome	Função
<code>strcpy(s1, s2)</code>	Copia s2 em s1
<code>strcat(s1, s2)</code>	Concatena s2 ao final de s1
<code>strlen(s1)</code>	Retorna o tamanho de s1
<code>strcmp(s1, s2)</code>	Retorna 0 se s1 e s2 são iguais, menor que 0 se $s1 < s2$ e maior que 0 se $s1 > s2$

- Essas funções usam o cabeçalho padrão `string.h`

# A função strcpy()

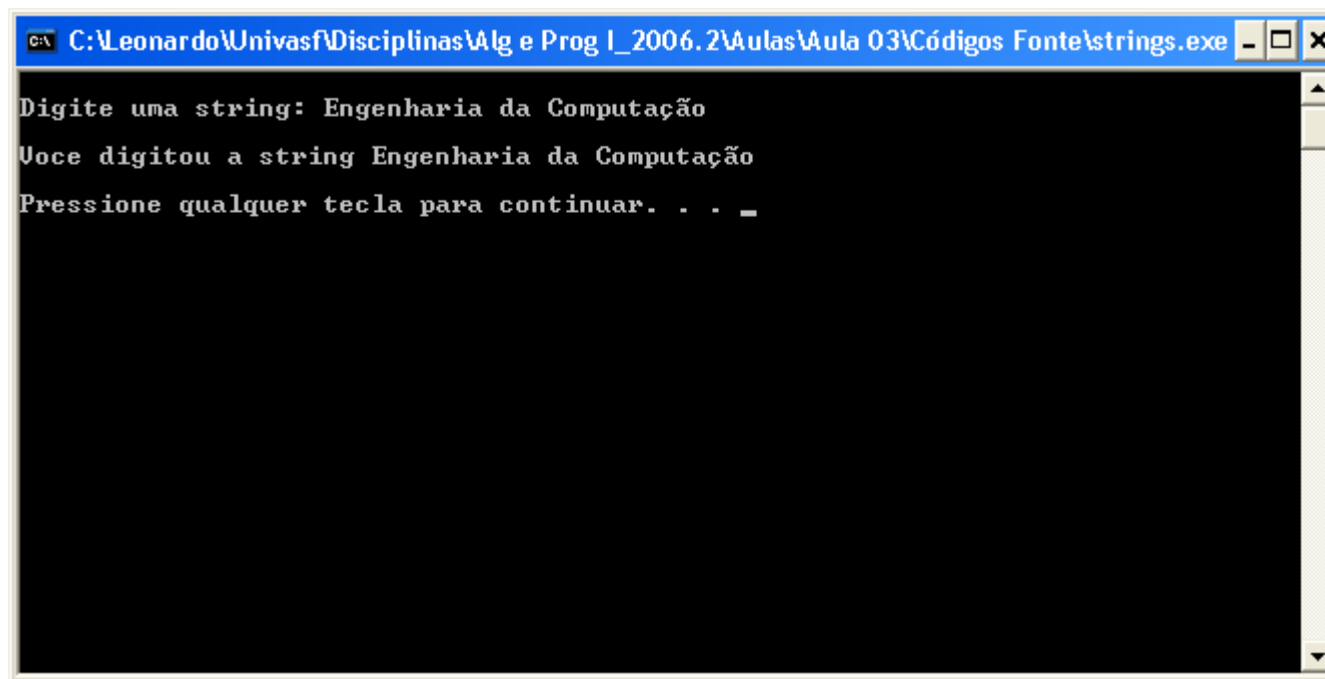
- A função **strcpy()** apresenta a seguinte forma geral:  
**strcpy(string\_destino, string\_origem);**
  - copia a string\_origem para a string\_destino.
- Vejamos o exemplo a seguir:

```
#include <stdio.h>
#include <string.h>
/* Este programa demonstra o funcionamento da função strcpy() */

int main()
{
    char str1[30], str2[30], str3[30];
    printf("\nDigite uma string: ");
    gets(str1);
    /* Copia str1 em str2 */
    strcpy(str2, str1);
    /* Copia "Voce digitou a string " em str3 */
    strcpy(str3, "Voce digitou a string ");
    printf("\n%s%s\n\n", str3, str2);
    system("pause");
    return 0;
}
```

# A função strcpy()

- A saída no console para o programa anterior será:



The screenshot shows a Windows console window titled "C:\Leonardo\Univasf\Disciplinas\Alg e Prog I\_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe". The console output is as follows:

```
Digite uma string: Engenharia da Computação
Voce digitou a string Engenharia da Computação
Pressione qualquer tecla para continuar. . . _
```

# A função strcat()

- A função **strcat()** apresenta a seguinte forma geral:

**strcat(string\_destino, string\_origem);**

- Concatena a string\_destino à string\_destino.

- Vejamos o exemplo a seguir:

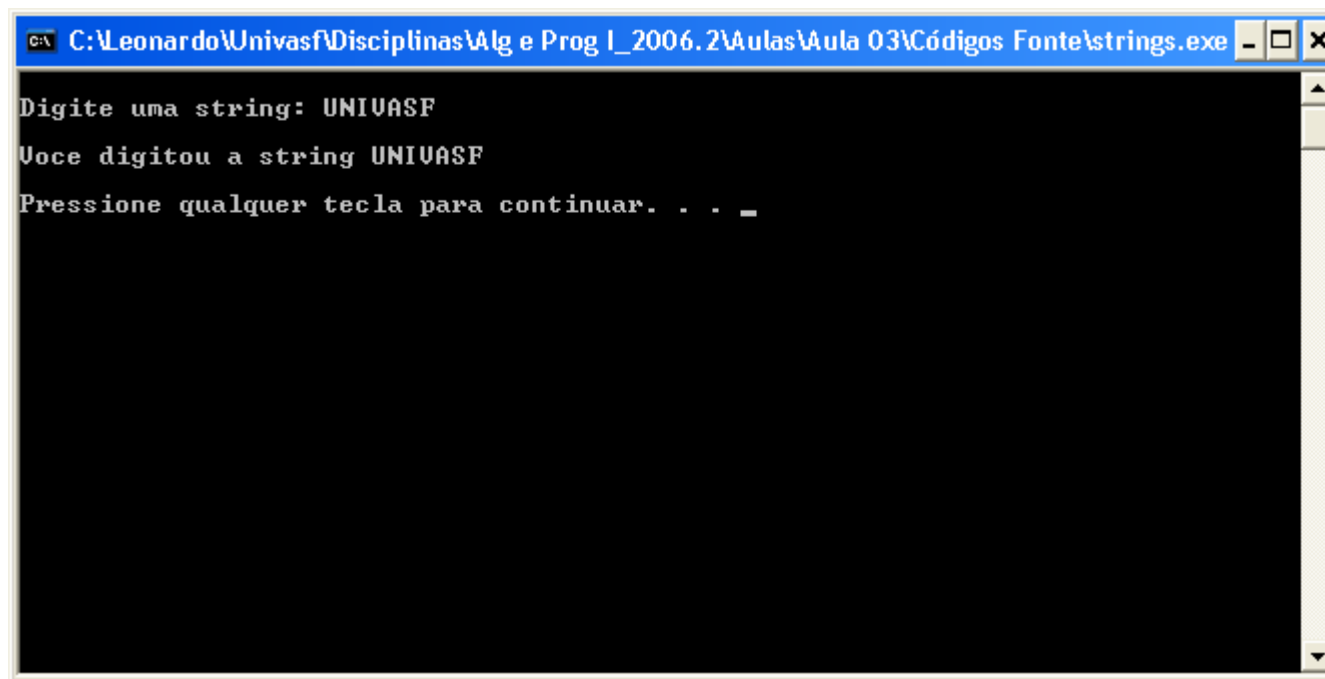
```
#include <stdio.h>
#include <string.h>
/* Este programa demonstra o funcionamento da função strcat() */

int main()
{
    char str1[30], str2[30];
    printf("\nDigite uma string: ");
    gets(str1);
    /* Copia "Voce digitou a string" em str2 */
    strcpy(str2, "Voce digitou a string ");
    /* str2 armazenará "Voce digitou a string " + o conteúdo de str1 */
    strcat(str2, str1);
    printf("\n%s\n\n", str2);
    system("pause");
    return 0;
}
```



# A função strcat()

- A saída no console para o programa anterior será:



The screenshot shows a Windows console window titled "C:\Leonardo\Univasf\Disciplinas\Alg e Prog I\_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe". The console output is as follows:

```
Digite uma string: UNIVASF
Voce digitou a string UNIVASF
Pressione qualquer tecla para continuar. . . _
```

# A função strlen()

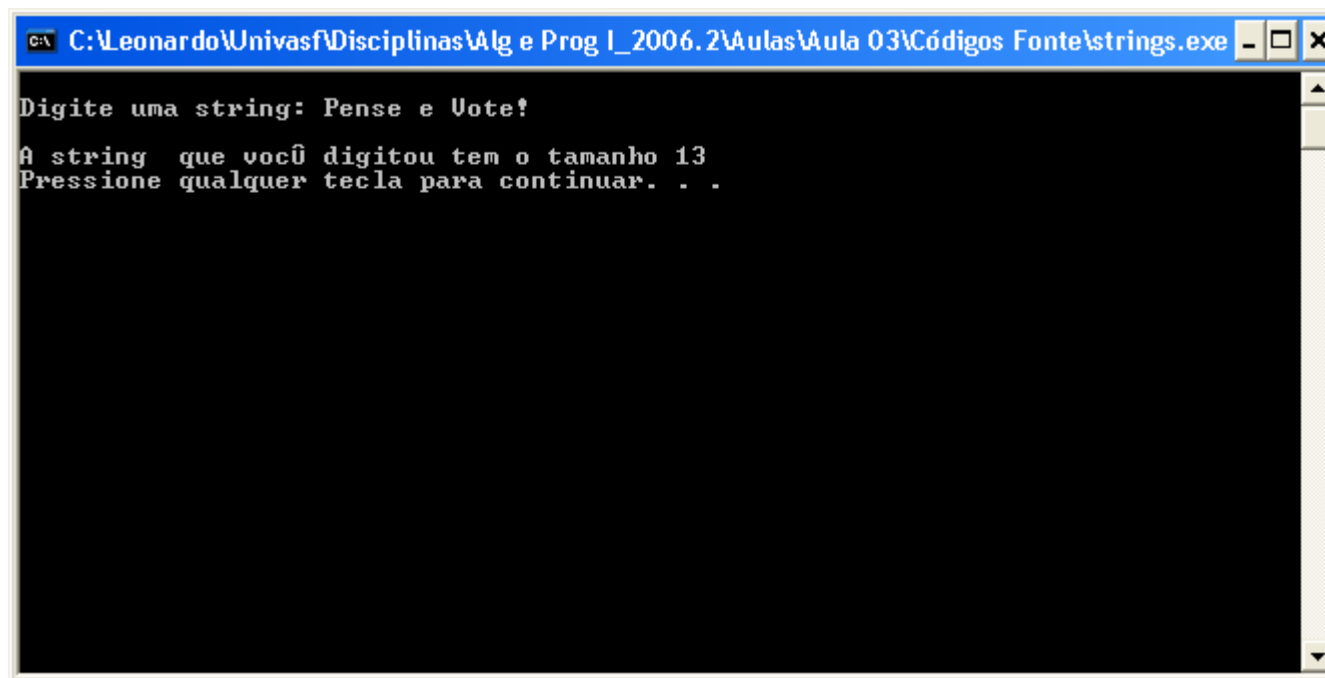
- A função **strlen()** apresenta a seguinte forma geral:  
**strlen(string);**
  - Retorna o comprimento da string fornecida.
- O terminador NULL ('\0') não é contado, vejamos:

```
#include <stdio.h>
#include <string.h>
/* Este programa demonstra o funcionamento da função strlen() */

int main()
{
    int tamanho;
    char str[30];
    printf("\nDigite uma string: ");
    gets(str);
    tamanho = strlen(str);
    /* Mostra o tamanho da string digitada */
    printf("\nA string que você digitou tem o tamanho %d\n", tamanho);
    system("pause");
    return 0;
}
```

# A função strlen()

- A saída no console para o programa anterior será:



```
C:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe
Digite uma string: Pense e Vote!
A string que você digitou tem o tamanho 13
Pressione qualquer tecla para continuar. . .
```

# A função strcmp()

- A função **strcmp()** apresenta a seguinte forma geral:

**strcmp(string1, string2);**

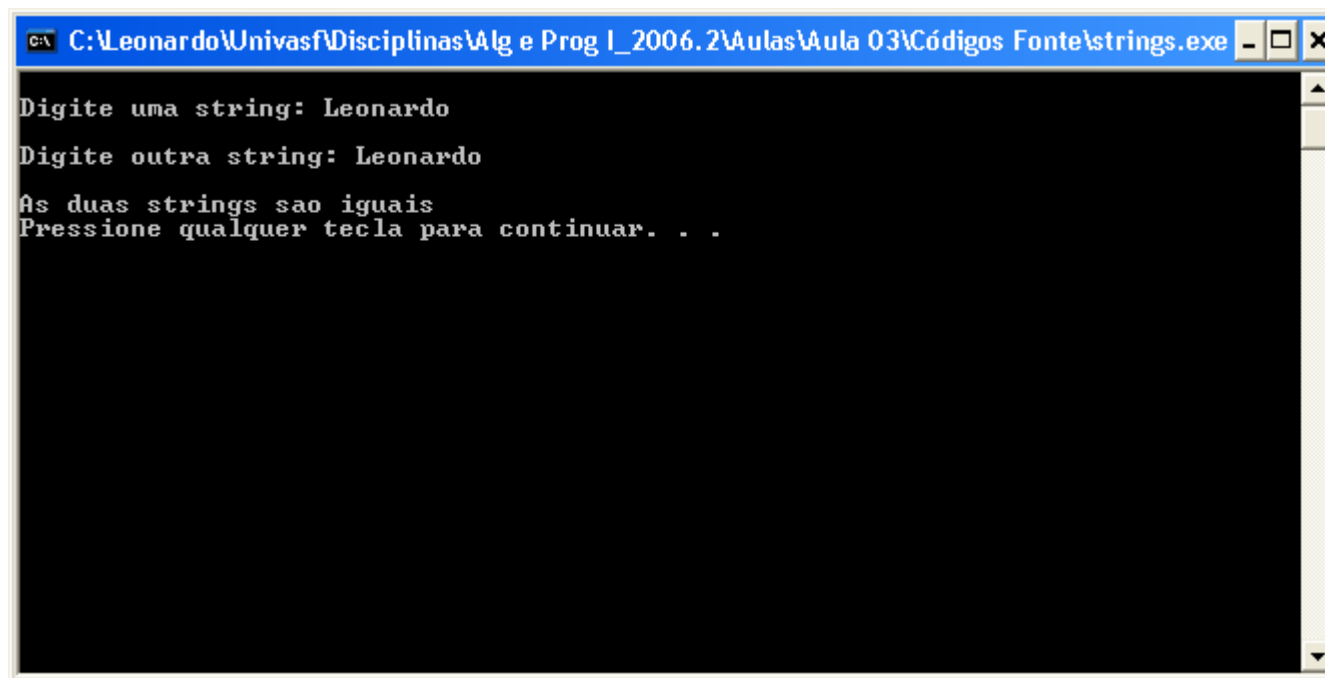
- Compara a string 1 com a string 2. Se as duas forem idênticas retorna 0. Se elas forem diferentes a função retorna não-zero

```
#include <stdio.h>
#include <string.h>
/* Este programa demonstra o funcionamento da função strcmp() */

int main()
{
    char str1[30], str2[30];
    printf("\nDigite uma string: ");
    gets(str1);
    printf("\nDigite outra string: ");
    gets(str2);
    if(strcmp(str1, str2))
        printf("\nAs duas strings são diferentes\n");
    else
        printf("\nAs duas strings são iguais\n");
    system("pause");
    return 0;
}
```

# A função strcmp()

- A saída no console para o programa anterior será:



```
C:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe
Digite uma string: Leonardo
Digite outra string: Leonardo
As duas strings são iguais
Pressione qualquer tecla para continuar. . .
```

# Matriz Unidirecional

- Uma matriz é uma coleção de variáveis do mesmo tipo que é referenciada por um nome comum.

```
float exemplo[20];  
int  numeros[3];
```

- Um elemento específico em uma matriz é acessado por meio de um índice:

```
exemplo[0] = 10.5;  
numeros[1] = 20
```

# Matriz Unidirecional

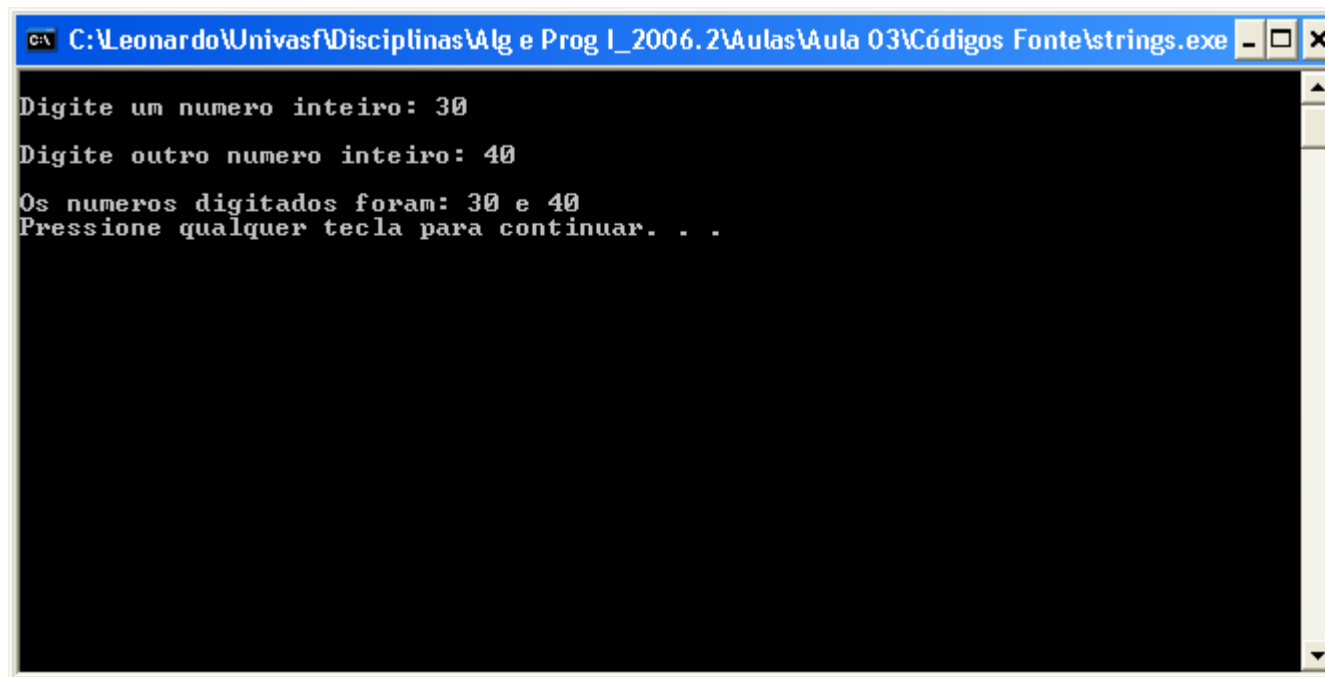
- Na linguagem C a numeração começa sempre em zero. Isto significa que os dados de uma matriz declarada como `int numeros[3]` serão indexados de 0 a 2, vejamos:

```
#include <stdio.h>
/* Este programa demonstra o funcionamento das matrizes unidirecionais() */

int main()
{
    int numeros[3];
    printf("\nDigite um numero inteiro: ");
    scanf("%d", &numeros[0]);
    printf("\nDigite outro numero inteiro: ");
    scanf("%d", &numeros[2]);
    printf("\nOs numeros digitados foram: %d e %d\n", numeros[0], numeros[2]);
    system("pause");
    return 0;
}
```

# Matriz Unidirecional

- A saída no console para o programa anterior será:



A screenshot of a Windows console window titled "C:\Leonardo\Univasf\Disciplinas\Alg e Prog I\_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe". The console displays the following text: "Digite um numero inteiro: 30", "Digite outro numero inteiro: 40", "Os numeros digitados foram: 30 e 40", and "Pressione qualquer tecla para continuar. . .".

```
C:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2006.2\Aulas\Aula 03\Códigos Fonte\strings.exe
Digite um numero inteiro: 30
Digite outro numero inteiro: 40
Os numeros digitados foram: 30 e 40
Pressione qualquer tecla para continuar. . .
```



# Matriz Bidimensional

- Para declarar uma matriz bidimensional mat de inteiros com tamanho 3, 4 teremos:

```
int mat [3][4];
```

- Para acessar o primeiro elemento da primeira coluna e da primeira fila de mat teremos:

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

- Matrizes bidimensionais são armazenadas em uma matriz linha-coluna, onde o primeiro índice indica a linha e o segundo, a coluna.

# Matriz Bidimensional

- Portanto, a visualização da matriz `mat [3][4]` é a seguinte:

`mat [i][j]`

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

# Matriz Bidimensional

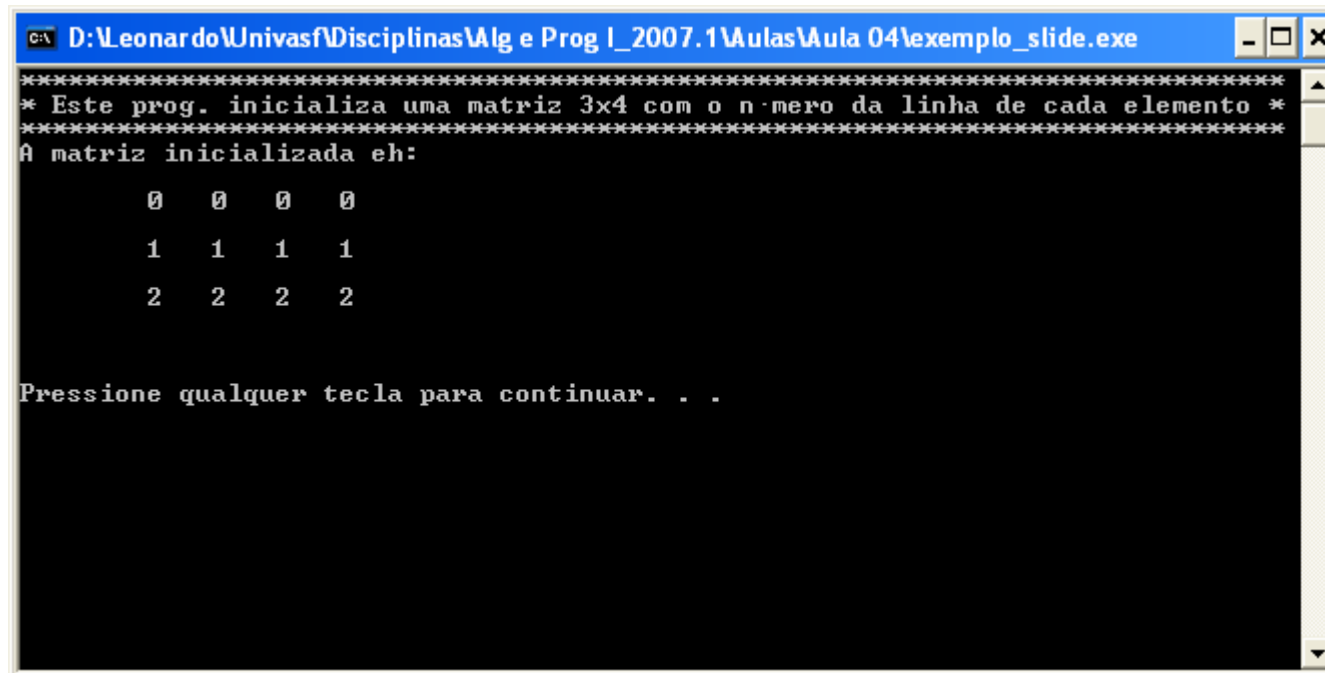
- Vejamos um exemplo de como manipular matrizes bidimensionais:

```
#include<stdio.h>
/* Este programa inicializa uma matriz de inteiros 3 x 4 com o número da linha de cada elemento*/
int main()
{
    int lin, col, mat[3][4], cont=0;
    printf("*****");
    printf("\n* Este prog. inicializa uma matriz 3x4 com o número da linha de cada elemento *");
    printf("\n*****");
    for(lin=0; lin<3; lin++)
        for(col=0; col<4; col++)
            mat[lin][col] = lin;
    printf("\nA matriz inicializada eh: \n\n\t");

    for(lin=0; lin<3; lin++)
        for(col=0; col<4; col++, cont++)
        {
            printf("%d\t", mat[lin][col]);
            if( (cont==3) || (cont==7) )
                printf("\n\n\t");
        }
    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

# Matriz Bidimensional

- A saída no console para o programa anterior será:



```
D:\Leonardo\Univasf\Disciplinas\Alg e Prog I_2007.1\Aulas\Aula 04\exemplo_slide.exe
*****
* Este prog. inicializa uma matriz 3x4 com o n-mero da linha de cada elemento *
*****
A matriz inicializada eh:
    0  0  0  0
    1  1  1  1
    2  2  2  2

Pressione qualquer tecla para continuar. . .
```

# Bibliografia

- SCHILDT H. *"C Completo e Total"*, Makron Books. SP, 1997.
- MIZRAHI, V. V. *"Treinamento em Linguagem C++ Módulo 1"*, Makron Books, SP, 1995.
- FORBELLONE, A. L. V. *"Lógica de Programação: A construção de algoritmos e estruturas de dados"*, Prentice Hall, SP, 2005.