

MATRIZES EM LINGUAGEM C

2º PERÍODO - ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

Linguagem de Programação - ILP010

Profa. Me Juliana Pasquini

E-mail: pasquini.juliana01@fatec.sp.gov.br

21-08-2014

Vetores (Matriz Unidimensional)

- ❑ Um vetor é um conjunto de dados do mesmo tipo que possui um nome comum.
- ❑ Elementos deste conjunto são referenciados através de índices.
- ❑ Forma geral para declarar uma matriz unidimensional:
tipo nome [tamanho];

Declaração de Vetores – Matriz Unidimensional (1)

- A forma geral da declaração de um vetor é:

`tipo nome [tamanho];`

- O primeiro elemento tem índice 0.
- O espaço de memória, em bytes, ocupado por um vetor é igual a:
total em bytes = tamanho vetor * (número de bytes ocupado pelo tipo)

- Exemplos:

```
int numero[1000];
```

```
float nota[65];
```

```
char nome[40]; /* STRING de 40 caracteres */
```

Declaração de Vetores – Matriz Unidimensional (2)

□ Exemplo de declaração:

```
float exemplo[20];
```

□ Acessando um elemento individual

```
exemplo[0] // acessa o primeiro elemento
```

```
exemplo[1] // acessa o segundo elemento
```

```
.
```

```
.
```

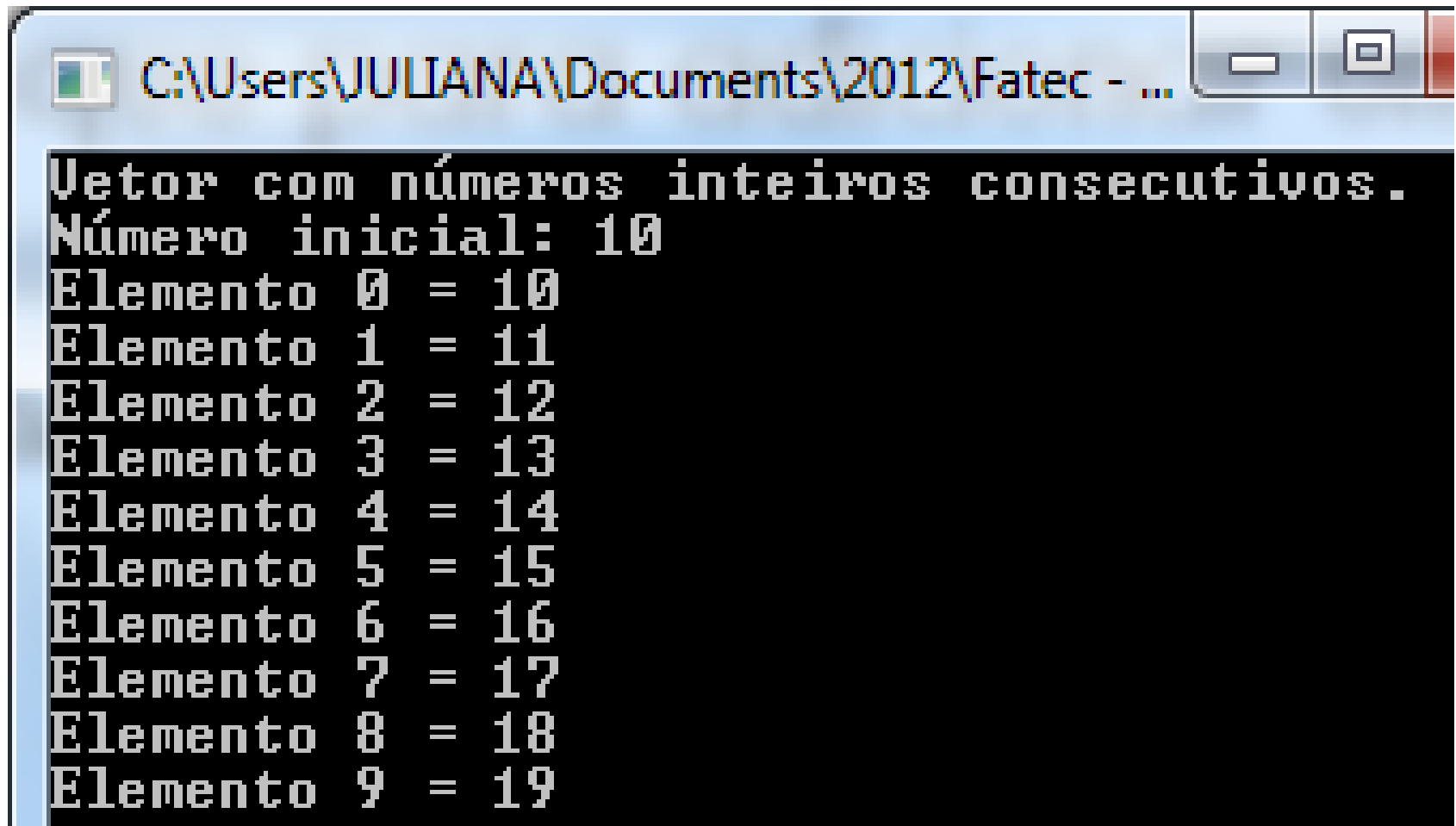
```
.
```

```
exemplo[19]
```

Carregando e Acessando

```
main() {  
    int vetor[10], i, num;  
    printf ("Vetor com números inteiros  
consecutivos.\n");  
    printf("Número inicial: ");  
    scanf("%d", &num);  
  
    for (i=0 ; i<10; i++)  
        vetor[i] = num++; /* Populando */  
  
    for (i=0; i <10; i++)  
        printf("Elemento %d = %d\n", i, vetor[i]);  
    getch();  
}
```

Carregando e Acessando



A screenshot of a Windows command prompt window. The title bar shows the file path "C:\Users\JULIANA\Documents\2012\Fatec - ...". The window contains the following text output from a C program:

```
Vetor com números inteiros consecutivos.  
Número inicial: 10  
Elemento 0 = 10  
Elemento 1 = 11  
Elemento 2 = 12  
Elemento 3 = 13  
Elemento 4 = 14  
Elemento 5 = 15  
Elemento 6 = 16  
Elemento 7 = 17  
Elemento 8 = 18  
Elemento 9 = 19
```

Carregando e Acessando(Vetor/Função)

```
#include<stdio.h>
#include<conio.h>

int popula_vetor (int vetor[10]);
void exibe_vetor (int vetor[10]);

main(){
    int vetor[10];
    popula_vetor (vetor);
    exibe_vetor (vetor);
    getch();
}
```

```
popula_vetor (int vetor[10])
{
    codificar...
}

void exibe_vetor (int vetor[10])
{
    codificar...
}
```

Carregando e Acessando(Vetor/Função)

<pre>#include<stdio.h> #include<conio.h> int popula_vetor (int vetor[10]); void exhibe_vetor (int vetor[10]); main(){ int vetor[10]; popula_vetor (vetor); exhibe_vetor (vetor); getch(); }</pre>	<pre>popula_vetor (int vetor[10]) { int num,i; printf ("Vetor com números inteiros consecutivos.\n"); printf("Número inicial: "); scanf("%d", &num); for (i=0 ; i<10; i++) vetor[i] = num++; /* Populando */ return vetor[10]; } void exhibe_vetor (int vetor[10]) { int i; for (i=0; i <10; i++) printf("Elemento %d = %d\n", i, vetor[i]); }</pre>
---	--

Acessando Vetores – Matriz Unidimensional

- O que acontece se escrevermos:

```
printf ("Elemento %d\n",vetor[11]);  
printf ("Elemento %d\n",vetor[12]);
```

- **OBS.:** O C não verifica se o índice que você usou está dentro dos limites válidos. Sendo assim fica a cargo do programador tomar os devidos cuidados

Vetores – Exemplo

```
main( )  
{  
    int x[100];  
    int t;  
    for (t=0; t< 100; ++t)  
        x[t]=t;  
}
```

Vetores – Exemplo

```
#include <stdio.h>
main (){
    int num[100]; /* Declara um vetor de inteiros de
                  100 posicoes */
    int count=0;
    int totalnums;
    do{
        printf ("\nEntre com um numero (-999 p/terminar): ");
        scanf ("%d",&num[count]);
        count++;
    } while (num[count-1]!=-999);

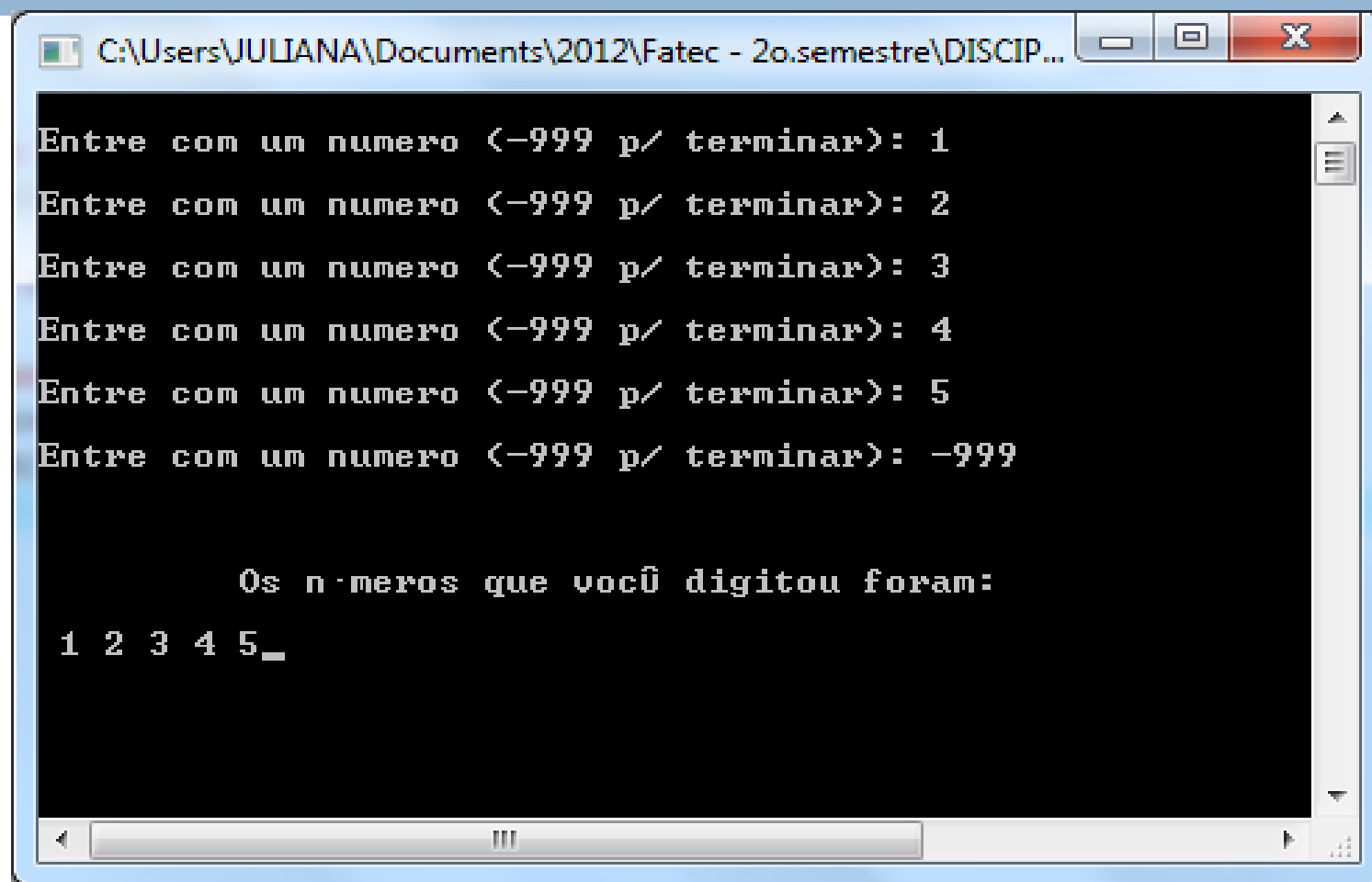
    totalnums=count-1;
    printf ("\n\n\n\t Os números que você digitou foram:\n\n");
    (codificar – exibir o vetor)
}
```

Vetores – Exemplo

```
#include <stdio.h>
main (){
    int num[100]; /* Declara um vetor de inteiros de
                  100 posicoes */
    int count=0;
    int totalnums;
    do{
        printf ("\nEntre com um numero (-999 p/terminar): ");
        scanf ("%d",&num[count]);
        count++;
    } while (num[count-1]!=-999);

    totalnums=count-1;
    printf ("\n\n\n\t Os números que você digitou foram:\n\n");
    for (count=0;count<totalnums;count++)
        printf (" %d",num[count]);
}
```

Vetores – Exemplo



```
C:\Users\JULIANA\Documents\2012\Fatec - 2o.semestre\DISCIP...  
Entre com um numero <-999 p/ terminar>: 1  
Entre com um numero <-999 p/ terminar>: 2  
Entre com um numero <-999 p/ terminar>: 3  
Entre com um numero <-999 p/ terminar>: 4  
Entre com um numero <-999 p/ terminar>: 5  
Entre com um numero <-999 p/ terminar>: -999  
  
    Os n-meros que vocô digitou foram:  
1 2 3 4 5_
```

Strings (1)

- O que são?
 - ▣ São vetores de caracteres (matriz unidimensional).

- Como declarar?

```
char nome_da_string [tamanho];
```

- Toda string deve terminar com o caracter '\0'.

```
main ()
{
    int count;
    char str1[30];
    char str2[30];
    printf ("Digite uma string:");
    gets (str1); // scanf ("%s",&str1);

    for (count=0;str1[count] !='\0';count++)
        str2[count]=str1[count];
    str2[count]='\0';

    printf ("Str2: ");
    for (count=0;str2[count] !='\0';count++)
        printf ("%c",str2[count]); //printf ("%s",str2);
}
```

Funções para manipulação de Strings: **Paramos aqui**

- gets
- strcpy
- strcat
- strlen
- strcmp

string.h

Função gets

- O que faz?
 - ▣ Lê uma string do teclado.

- Sua forma geral é:

```
gets (nome_da_string);
```

- Exemplo:

```
#include <stdio.h>
main ()
{
    char string[100];
    printf ("Digite o seu nome: ");
    gets (string);
    printf ("\n\n Ola %s", string);
}
```

Função strcpy

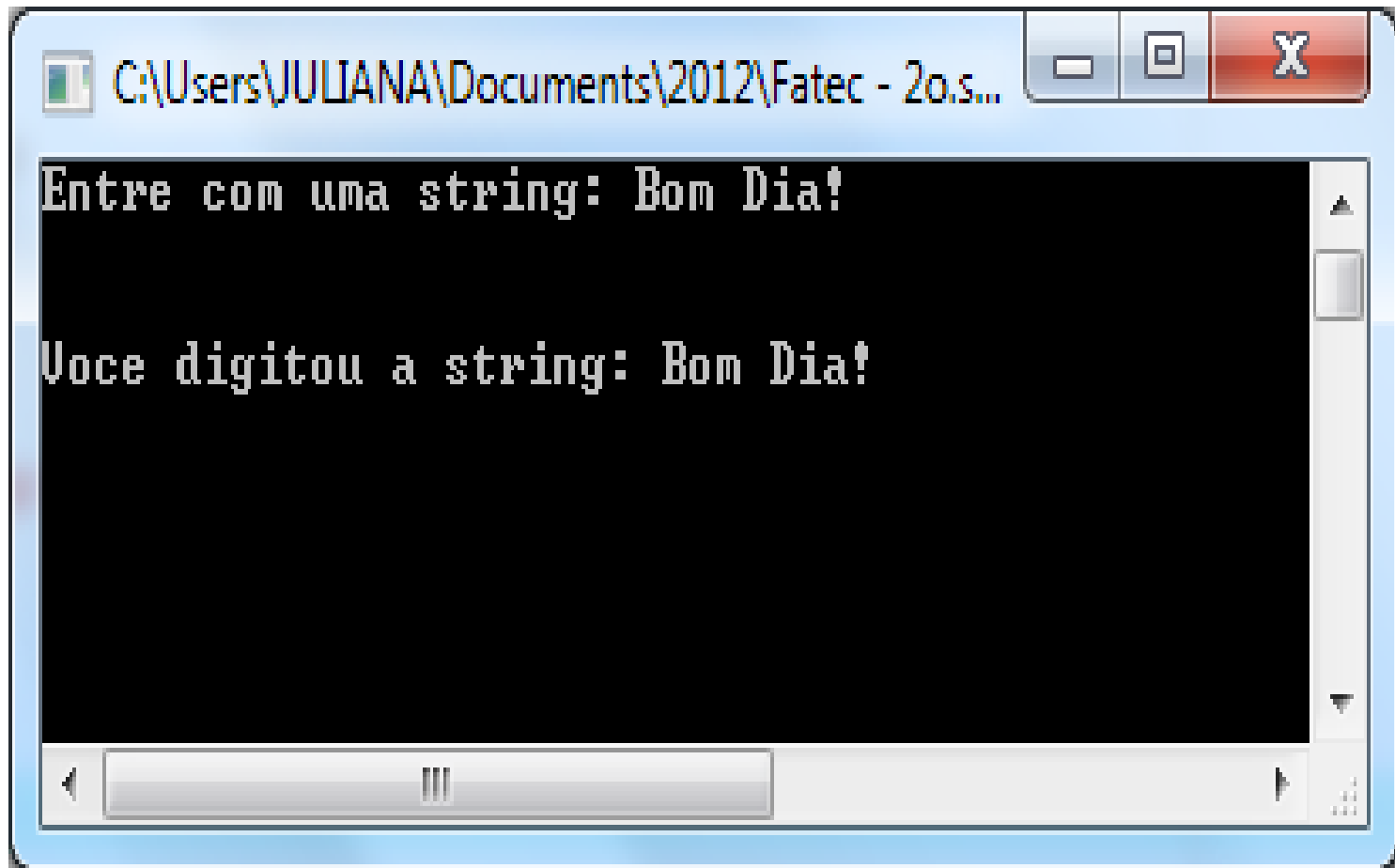
- O que faz?
 - ▣ Copia uma string para outra.
- Sua forma geral é:
`strcpy (string_destino, string_origem);`

Função strcpy

Exemplo:

```
#include <stdio.h>
#include <string.h>
main ()
{
    char str1[100],str2[100],str3[100];
    printf ("Entre com uma string: ");
    gets (str1);
    strcpy (str2,str1);
    strcpy (str3,"Voce digitou a string ");
    printf ("\n\n%s%s",str3,str2);
}
```

Função strcpy



Função strcat

- O que faz?
 - ▣ Concatena duas strings.
- Sua forma geral:
strcat (string_destino,string_origem);
- **Obs.:** A string de origem permanecerá inalterada e será anexada ao fim da string de destino.

Função strcat

Um exemplo:

```
#include <stdio.h>
#include <string.h>
main ()
{
    char str1[100],str2[100];
    printf ("Entre com uma string 1: ");
    gets (str1);
    printf ("Entre com uma string 2: ");
    gets (str2);
    strcat (str2,str1);
    printf ("\n\n%s",str2);
}
```

Função strlen

- O que faz?
 - ▣ Retorna o comprimento de uma string sem levar em conta o terminador nulo. '\0'
- Sua forma geral é:
strlen (string);

Função strlen

Exemplo:

```
#include <stdio.h>
#include <string.h>
main ()
{
    int size;
    char str[100];
    printf ("Entre com uma string: ");
    gets (str);
    size=strlen (str);
    printf ("\n\nA string que voce digitou tem
tamanho %d",size);
}
```


Função strlen

```
Entre com uma string: Bom Dia
```

```
A string que voce digitou tem tamanho 7.
```

Função strcmp

- O que faz?
 - ▣ Compara duas strings lexicograficamente e:
 - Retorna 0 se elas forem iguais
 - Um valor menor que 0 se a primeira for menor
 - Um valor maior que 0 se a primeira for maior
- Sua forma geral é:
strcmp (string1,string2);

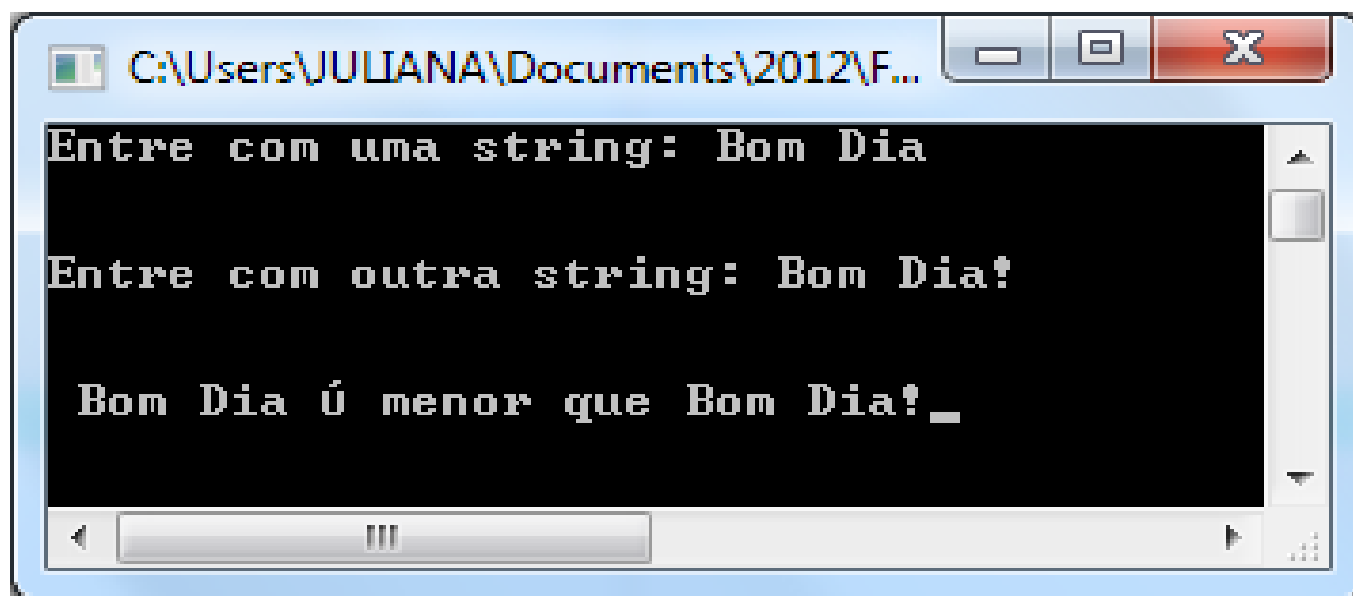
Obs: lexicograficamente – Baseia-se na ordenação dos caracteres estabelecida na tabela ISO8859-1. Os primeiros 128 caracteres constituem a "tabela ASCII" (American Standard Code for Information Interchange).

Função strcmp

Exemplo:

```
main ()
{
    char str1[100],str2[100];
    printf ("Entre com uma string: ");
    gets (str1);
    printf ("\n\nEntre com outra string: ");
    gets (str2);
    if ((strcmp(str1,str2)) < 0 )
        printf ("\n\n %s é menor que %s",str1,str2);
    else if ((strcmp(str1,str2)) > 0)
        printf ("\n\n%s é maior que %s",str1,str2);
    else
        printf ("\n\n%s é igual a %s “,str1,str2);
}
```

Função strcmp



A screenshot of a Windows command prompt window. The title bar shows the file path "C:\Users\JULIANA\Documents\2012\F...". The window contains three lines of text in a monospaced font: "Entre com uma string: Bom Dia", "Entre com outra string: Bom Dia?", and "Bom Dia é menor que Bom Dia?_". The third line shows the result of a strcmp comparison, where the first string is considered less than the second.

```
C:\Users\JULIANA\Documents\2012\F...  
Entre com uma string: Bom Dia  
Entre com outra string: Bom Dia?  
Bom Dia é menor que Bom Dia?_
```

Matrizes

- Matrizes bidimensionais
- Matrizes de strings
- Matrizes multidimensionais
- Inicialização de Matrizes

Matrizes bidimensionais

- Como declara?

tipo nome [linha][coluna];

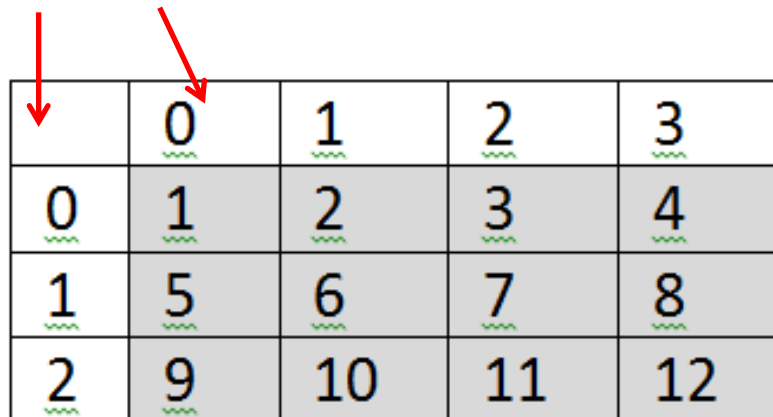
- Exemplo:

```
int matriz[3][4]
```

Carregando Matriz Bidimensionais

```
int const LINHA = 3, COLUNA = 4;  
int i, j, matriz[LINHA][COLUNA];  
  
for (i=0; i<LINHA; i++) {  
    printf("\n%dª Linha:\n", i+1);  
    for (j=0; j<COLUNA; j++)  
        scanf("%d", &matriz[i][j]);  
}
```

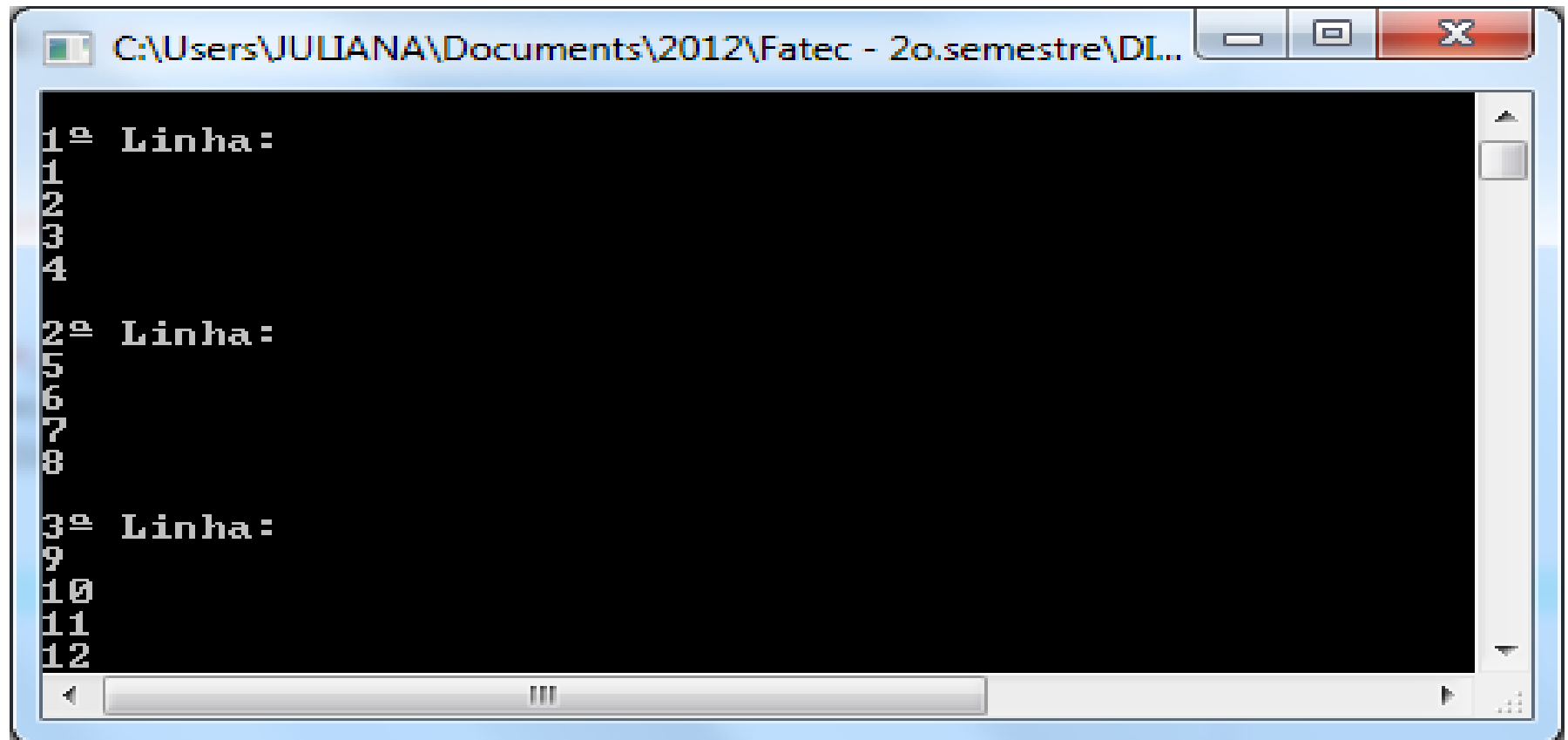
Matriz [linha] [coluna]



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

matriz [0,0]=1
matriz [0,1]=2
matriz[0,2]=3
..
matriz [2,3]=12

Carregando Matriz Bidimensionais



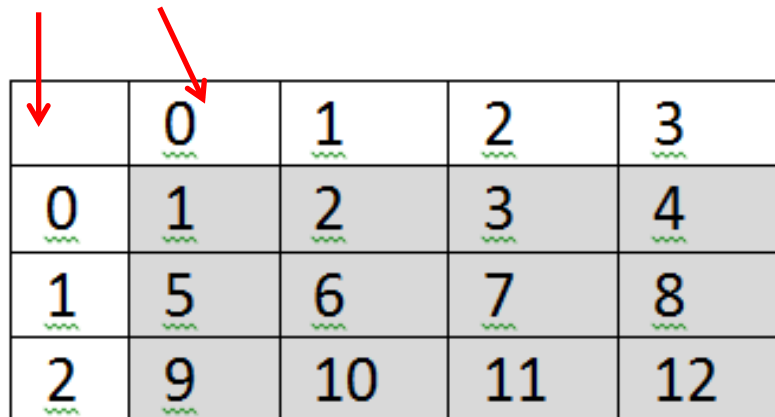
A screenshot of a Windows command prompt window. The title bar shows the path "C:\Users\JULIANA\Documents\2012\Fatec - 2o.semestre\DI...". The window contains the following text:

```
1ª Linha:  
1  
2  
3  
4  
  
2ª Linha:  
5  
6  
7  
8  
  
3ª Linha:  
9  
10  
11  
12
```


Acessando Matriz Bidimensionais

```
for (i=0; i<LINHA; i++) {  
    for (j=0; j<COLUNA; j++)  
        printf("%d", matriz[i][j]);  
}
```

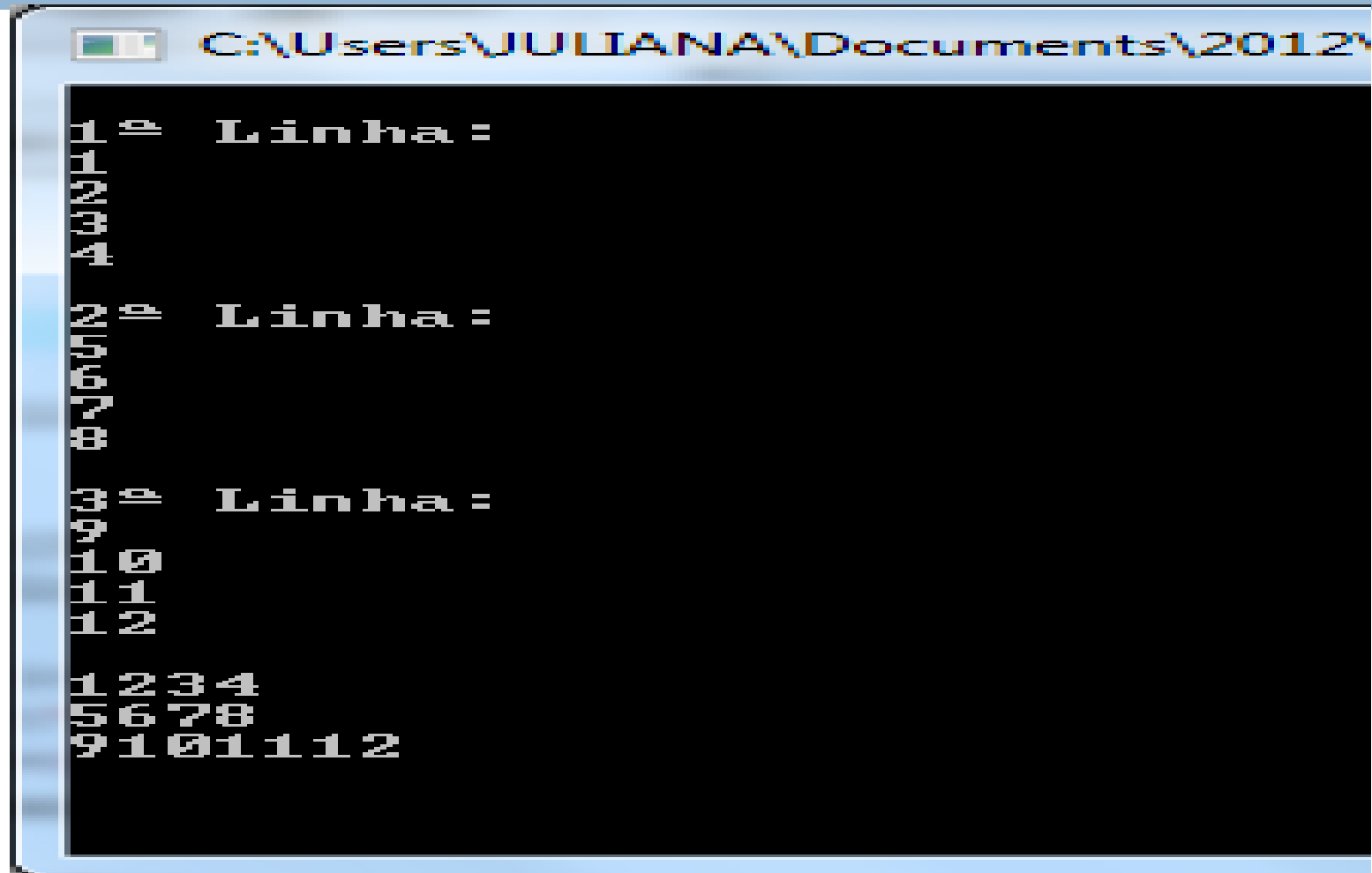
Matriz [linha] [coluna]



	<u>0</u>	<u>1</u>	<u>2</u>	<u>3</u>
<u>0</u>	1	2	3	4
<u>1</u>	5	6	7	8
<u>2</u>	9	10	11	12

matriz [0,0]=1
matriz [0,1]=2
matriz[0,2]=3
..
matriz [2,3]=12

Acessando Matriz Bidimensionais



```
C:\Users\JULIANA\Documents\2012\>
1ª Linha:
1
2
3
4

2ª Linha:
5
6
7
8

3ª Linha:
9
10
11
12

1234
5678
9101112
```

Matrizes de strings

- O que são?
 - ▣ Matrizes de strings são matrizes bidimensionais de caracteres.

- *Como declara*

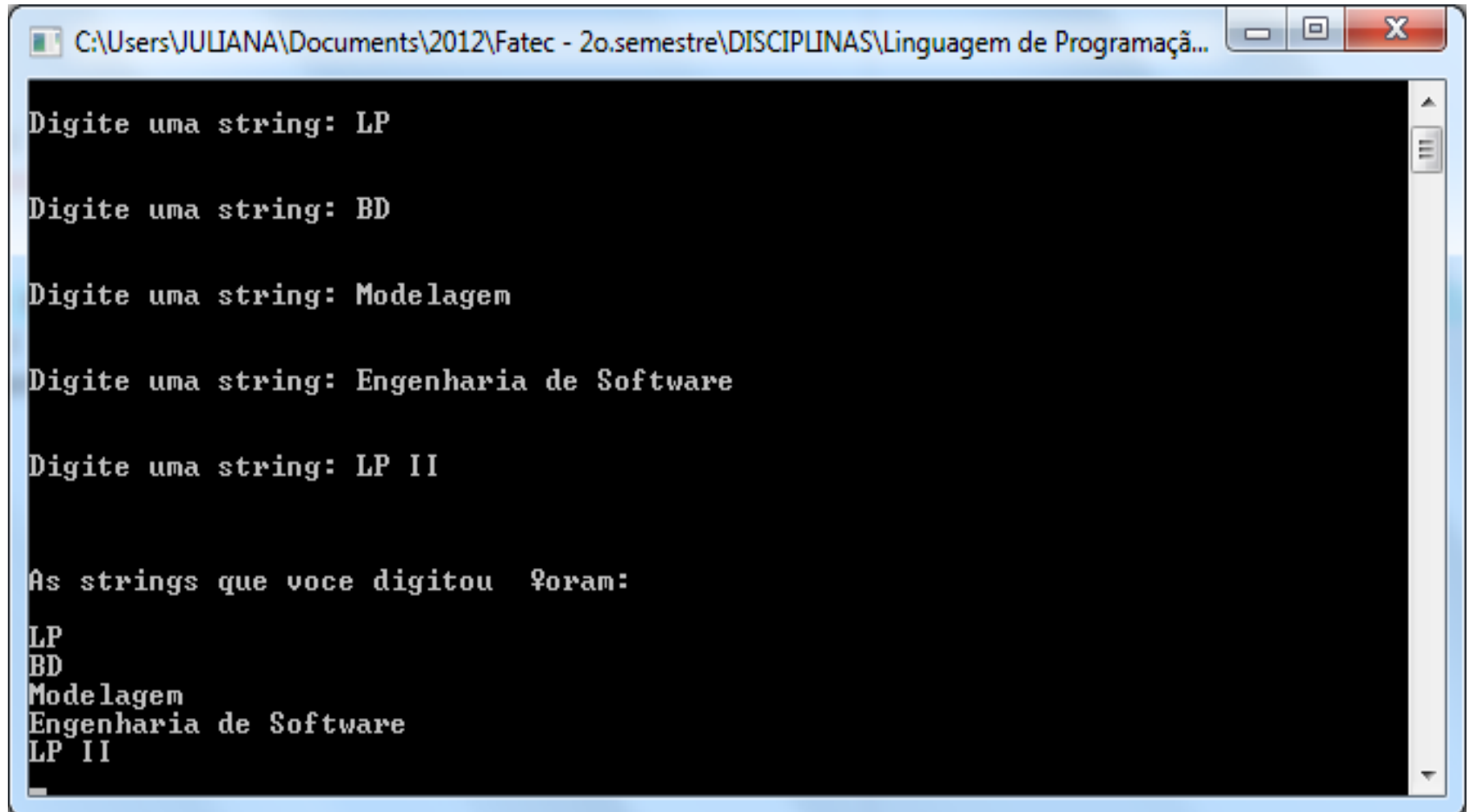
```
char nome[num_de_strings][compr_das_strings];
```

Matrizes de strings

Exemplo:

```
#include <stdio.h>
main ()
{
char strings [5][100];
int count;
    for (count=0;count<5;count++){
        printf ("\n\nDigite uma string: ");
        gets (strings[count]);
    }
    printf ("\n\n\nAs strings que voce digitou \n\n\n");
    for (count=0;count<5;count++){
        printf ("%s\n",strings[count]);
    }
}
```

Matrizes de strings



A screenshot of a Windows command prompt window. The title bar shows the path: C:\Users\JULIANA\Documents\2012\Fatec - 2o.semestre\DISCIPLINAS\Linguagem de Programaçã... The window contains the following text:

```
Digite uma string: LP
Digite uma string: BD
Digite uma string: Modelagem
Digite uma string: Engenharia de Software
Digite uma string: LP II

As strings que voce digitou foram:
LP
BD
Modelagem
Engenharia de Software
LP II
```

Matrizes multidimensionais

□ Como declarar?

Tipo nome [tam1][tam2] ... [tamN];

□ Exemplo:

```
int matriz [10] [20] [5];
```

- Uma matriz N-dimensional funciona basicamente como outros tipos de matrizes. Basta lembrar que o índice que varia mais rapidamente é o índice mais à direita.

Inicialização de Matrizes

- Como inicializar:

```
tipo nome [tam1][tam2] ... [tamN] = {lista_de_valores};
```

- A lista de valores é composta por valores (do mesmo tipo da variável) separados por vírgula.
- Os valores devem ser dados na ordem em que serão colocados na matriz.

Inicialização de Matrizes

Exemplos:

```
float vect [5] = {1.3,4.5,2.7,4.1,0.0}
```

```
int matrix [3][4] = { 1,2,3,4,5,6,7,8,9,10,11,12};
```

```
char str [10] = { 'J','o','a','o','\0' }
```

```
char str [10] = "Joao";
```

```
char str_vect [3][10] = {"Joao","Maria","Jose"};
```


Inicialização de Matrizes

```
int const DIM = 5;
int vetor[DIM];
int vetor1[] = {10, 20, 30, 40, 50, 60, -1};
int vetor2[] = {3, 6, 9, 12, 15, 18, 21, 24};
int i, tam;

printf("\nVetor com tamanho predefinido\n");
for (i=0; i < DIM; i++)
    printf("Elemento %d = %d\n", i, vetor[i]);

tam = sizeof (vetor2) / sizeof (int);

for (i=0; i < tam ; i++)
    printf("Elemento %d = %d\n", i, vetor2[i]);
```