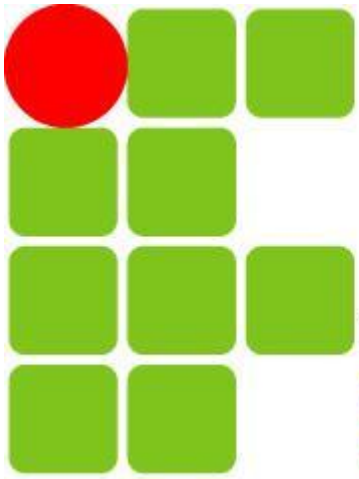


INSTITUTO FEDERAL
ESPÍRITO SANTO
Campus Cachoeiro de Itapemirim

Matrizes

Programação 1

Rafael Vargas Mesquita

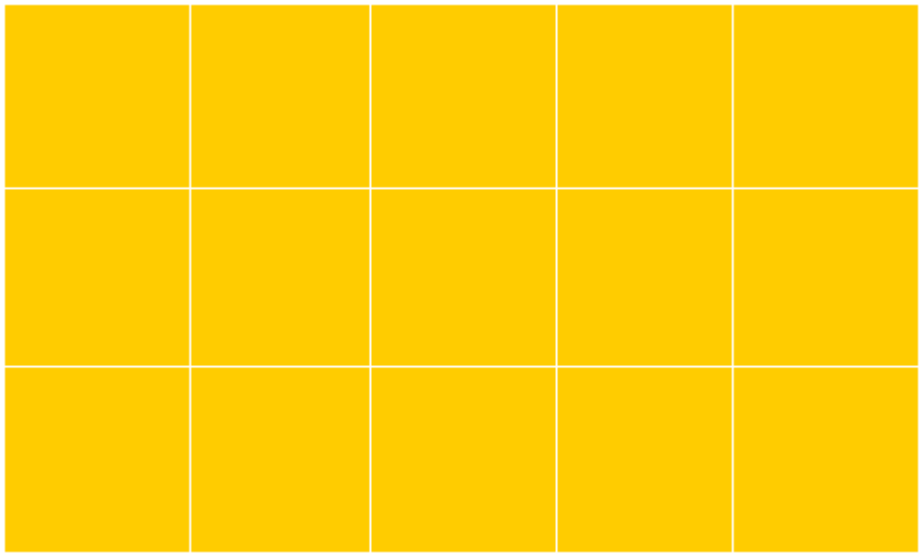
http://www.ci.ifes.edu.br
ftp://ftp.ci.ifes.edu.br/informatica/rafael/

Definição de Matriz

- Definição: é um conjunto de espaços de memória referenciados por um mesmo nome e que necessitam de um ou mais índices para que seus elementos sejam endereçados.

Representação Gráfica de Matriz

- Graficamente uma matriz bidimensional pode ser representada da seguinte forma:



Representação Gráfica de Matriz Exemplo

- A seguinte matriz pode representar os preços das passagens de uma empresa de transportes – comercial, executivo e leito (eixo x) das cidades (eixo y) até Cachoeiro do Itapemirim:

	Comercial	Executivo	Leito
São Paulo	43.30	60.33	97.26
Vitória	17.23	33.14	46.23
Rio de Janeiro	24.10	37.24	64.29

Declaração de Matrizes

- A declaração de matrizes bi-dimensionais pode ser feita da seguinte forma:

```
tipo_da_variável nome_da_variável [altura][largura];
```

- Exemplo:

```
int mtrx [20][10];
```

Manipulação de Matrizes

- A manipulação de matrizes é bastante semelhante a manipulação de vetores.
- Em vetores para acessar a uma posição utilizamos um índice, em matrizes um ou mais índices (vetores também são matrizes), uma matriz bi-dimensional por exemplo pode ser acessada da seguinte forma (declaração no slide anterior):

```
mtrx[0][0] = 10; //Escreve o valor 10 na posição [0][0]
printf("O valor da posição [0][0] é %d", mtrx[0][0]);
```

Manipulação de Matrizes
Exemplo

```
#include <stdio.h>
   int main (){
   int mtrx [20][10];
   int i,j,count;
   count=1;
   for (i=0; i<20; i++){
      for (j=0; j<10; j++){
         mtrx[i][j]=count;
         count++;
      }
   }
   return(0);
}
```

```
  1   2   3   4   5   6   7   8   9  10
 11  12  13  14  15  16  17  18  19  20
 21  22  23  24  25  26  27  28  29  30
 31  32  33  34  35  36  37  38  39  40
 41  42  43  44  45  46  47  48  49  50
 51  52  53  54  55  56  57  58  59  60
 61  62  63  64  65  66  67  68  69  70
 71  72  73  74  75  76  77  78  79  80
 81  82  83  84  85  86  87  88  89  90
 91  92  93  94  95  96  97  98  99 100
101 102 103 104 105 106 107 108 109 110
111 112 113 114 115 116 117 118 119 120
121 122 123 124 125 126 127 128 129 130
131 132 133 134 135 136 137 138 139 140
141 142 143 144 145 146 147 148 149 150
151 152 153 154 155 156 157 158 159 160
161 162 163 164 165 166 167 168 169 170
171 172 173 174 175 176 177 178 179 180
181 182 183 184 185 186 187 188 189 190
191 192 193 194 195 196 197 198 199 200
```

Matrizes de Strings

- Matrizes de strings são matrizes bidimensionais. Imagine uma string. Ela é um vetor. Se fizermos um vetor de strings estaremos fazendo uma lista de vetores. Esta estrutura é uma matriz bidimensional de chars. Podemos ver a forma geral de uma matriz de strings como sendo:

```
char nome_matriz [num_strings][compr_strings];
```

- Aí surge a pergunta: como acessar uma string individual? Fácil. É só usar apenas o primeiro índice. Então, para acessar uma determinada string faça:

```
nome_matriz [índice]
```

Matrizes de Strings
Exemplo

Aqui está um exemplo de um programa que lê 5 strings e as exibe na tela:

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

Matrizes Multidimensionais

- O uso de matrizes multidimensionais na linguagem C é simples. Sua forma geral é:

```
tipo nome_da_variável [tam1][tam2] ... [tamN];
```

- Exemplo:

```
int cubo_numerico[10][10][10];
```

Inicialização de Matrizes

- Podemos inicializar matrizes, assim como podemos inicializar variáveis. A forma geral de uma matriz como inicialização é:

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

Inicialização de Matrizes Exemplo

- Exemplos:

```
float vect [6] = { 1.3, 4.5, 2.7, 4.1, 0.0, 100.1 };
int matrx [2][3] = { 1, 2, 3, 4, 5, 6 };
char str [10] = { 'J', 'o', 'a', 'o', '\0' };
char str [10] = "Joao";
char str_vect [3][10] = { "Joao", "Maria", "Jose" };
```

Inicialização de Matrizes

- Sem especificar o tamanho:
  - Podemos, em alguns casos, inicializar matrizes das quais não sabemos o tamanho a priori. O compilador C vai, neste caso verificar o tamanho do que você declarou e considerar como sendo o tamanho da matriz.
  - Isto ocorre na hora da compilação e não poderá mais ser mudado durante o programa, sendo muito útil, por exemplo, quando vamos inicializar uma string e não queremos contar quantos caracteres serão necessários.

```
char mess [] = "IFES Cachoeiro de Itapemirim";
int matrx [][2] = { 1,2,2,4,3,6,4,8,5,10 };
```

Passando Matrizes como Parâmetros

- Exemplo:
  - Quando vamos passar uma matriz bidimensional como argumento de uma função, não será necessário especificar o número de linha na matriz, mas, sim, especificar o número de colunas. Seja a matriz:

```
int mat[10][20];
```

  - E que queiramos passá-la como argumento de uma função `func()`. Podemos declarar `func()`:

```
void func (int mat[10][20]);
void func (int mat[][20]);
```

Bibliografia

- SANTOS, Henrique José. Curso de Linguagem C da UFMG, apostila.
- FORBELLONE, André Luiz. Lógica de Programação – A Construção de Algoritmos e Estruturas de Dados. São Paulo: MAKRON, 1993.