UNIVERSIDADE da MADEIRA

CTeSP CURSOS TÉCNICOS SUPERIORES PROFISSIONAIS

TECNOLOGIAS DE PROGRAMAÇÃO DE SISTEMAS DE INFORMAÇÃO

# Matrizes/Arrays e Strings

PROGRAMAÇÃO ORIENTADA A OBJECTOS| Prof. Doutora Frederica Gonçalves

Cofinanciado por:

MADEIRA 14-20

REGIÃO AUTÓNOMA DA MADEIRA

PORTUGAL 2020

UNIÃO EUROPEIA
Fundo Social Europeu

- **Declaração de *arrays***
  - É uma estrutura de dados, em que estes são de um mesmo tipo - base, agrupados e identificados individualmente por um mesmo **nome** e por um **índice**.

    - **Tipos de *arrays:***

**Unidimensionais** → Tipo_base nome_do_array [n];

**Multidimensionais** → Tipo_base nome_do_array [e0] [e1] … [en];

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Declaração de *arrays***

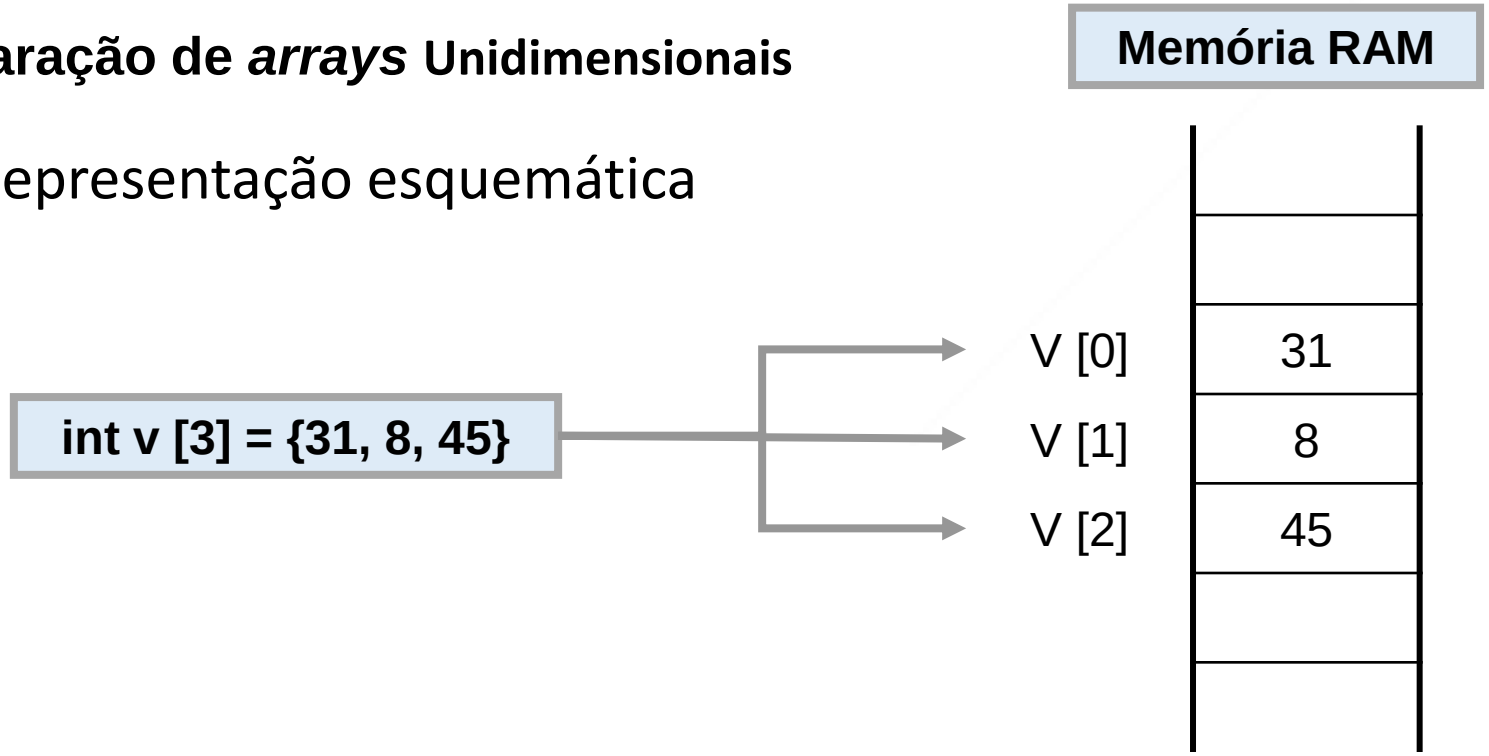
**int v [4];** →

Declara um *array* unidimensional, definindo uma variável **v**, constituída por 4 elementos do tipo **int.**
v [0]; v [1]; v [2]; v [3]

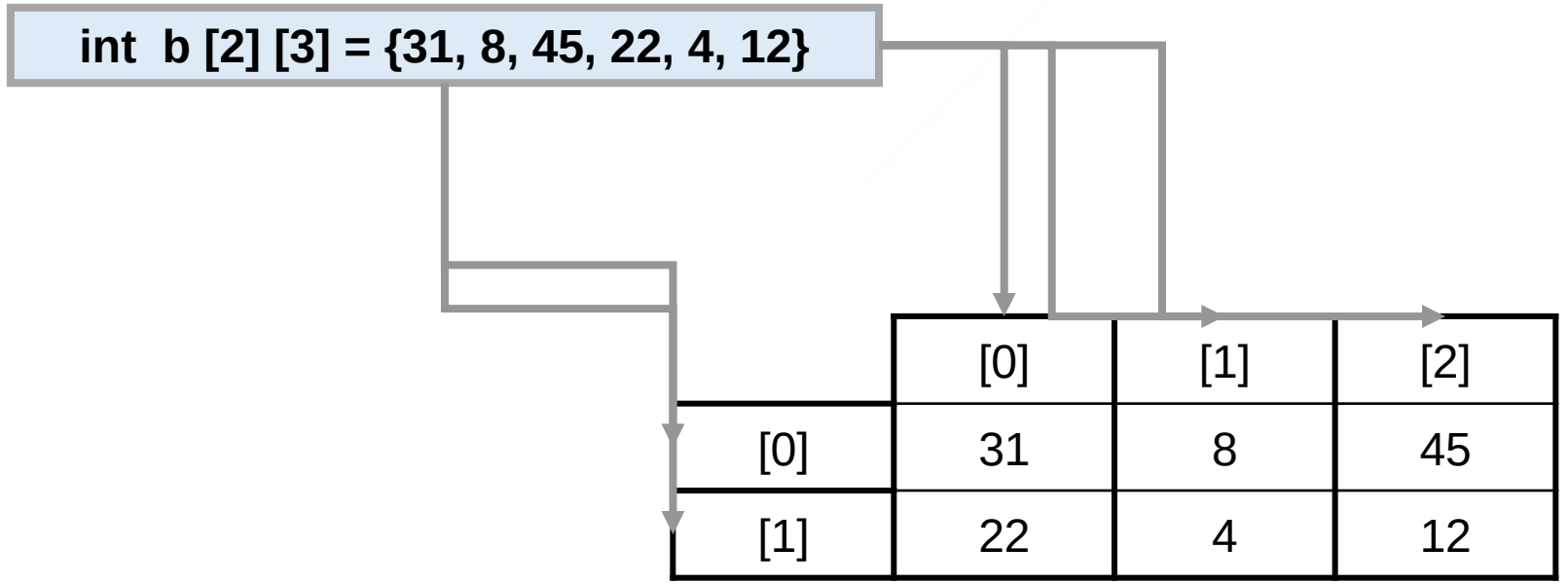
**int v [2] [3];** →

Declara um *array* multidimensional, especificamente bidimensional, definindo uma variável **v**, do tipo **int**, tendo 2 por 3 elementos;
v [0] [0] ; v [0] [1] ; v [0] [2] ; v[1] [0] ; v[1] [1] ; v [1] [2]

- **Declaração de *arrays* Unidimensionais**
  - Representação esquemática



- **Declaração de *arrays* Bidimensionais**
  - Representação esquemática



UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Inicialização de *arrays***

**tipo_base nome do array [n] = { <lista_de_valores>};**

- **Exemplo:**

**int v [3] = {14,15,16};**

**int v [ ] = {14,15,16} ;**

Com esta inicialização, os elementos do *array* **v**, ficam com os seguintes valores:
v [0] = 14 ; v [1] = 15 ; v [2] = 16 ;

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Exemplo Unidimensional**

```
…
int v[4]={12,31,8,45};
int i;

main(){
      for(i=0; i<4;i++){
      cout << "O valor do indice "<<i<<" e: "<<v[i]<<" \n";
…
```

- **Exemplo Bidimensional**

```
…
int v[2][3]={1,2,3,4,5,6};
int i,j;

main(){
      for(i=0; i<2;i++){
            for(j=0;j<3;j++){
      cout << "O valor do indice ("<<i<<","<<j<<") e:
"<<v[i][j]<<" \n";
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Exemplificação de pequenos programas utilizando arrays**
  - **Função rand() –** utilizada para gerar valores aleatórios.

```
…
int v[2];
int i;

main(){
      for(i=0; i<2;i++){
            v[i] = rand() % 100;
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Exemplificação de pequenos programas utilizando arrays**

- **Função srand() –** gerar valores aleatórios sempre diferentes;
- **Biblioteca <ctime> -** permite usar a função *time*, que devolve a hora do sistema.

```
#include <ctime>
…
int v[2][3];
int i,j;

main(){
      srand((unsigned) time (NULL));
      for(i=0; i<2;i++)
            for(j=0;j<3;j++)
                  v[i][j] = rand () % 50;
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **EXERCÍCIO**
- Criar um *array* unidimensional com 5 números inteiros, e preenchê-lo com valores aleatórios de 0 a 30, com a função srand(), e a biblioteca <ctime>. Depois apresentar na consola os valor do *array* separados por uma vírgula.

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **EXERCÍCIO**

```
#include <iostream>
#include <ctime>
using namespace std;
int v[5];
int i;
main(){
	srand((unsigned) time(NULL));
	for(i=0; i<5;i++)
	v[i] = rand() % 30;
	cout << "O valores do array sao: ";
	for(i=0; i<5;i++){
	cout <<v[i]<<“, ";
	}
	system("pause");
}
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- ***Strings* como *arrays* de *char* (caracteres)**
  - Em C ++ não existe um tipo-base de dados que seja *string.*
  - Uma *string*, então é declarada como um *array de char* (caracteres).
  - Utiliza-se a biblioteca **<string>.**

```
char t [5];
```

→ Declara um *array* **t**, do tipo ***char*** (caracteres) com 5 elementos.

```
char t [ ]= "Ana";
char t [ ] = {'A', 'n', 'a'}
```

→ Outras formas de criação de *strings*.

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- ***Exemplo***

**char t [ ] = “Teste”** → Declara um *array* **t**, do tipo ***char*** (caracteres) iniciado com uma *string.*

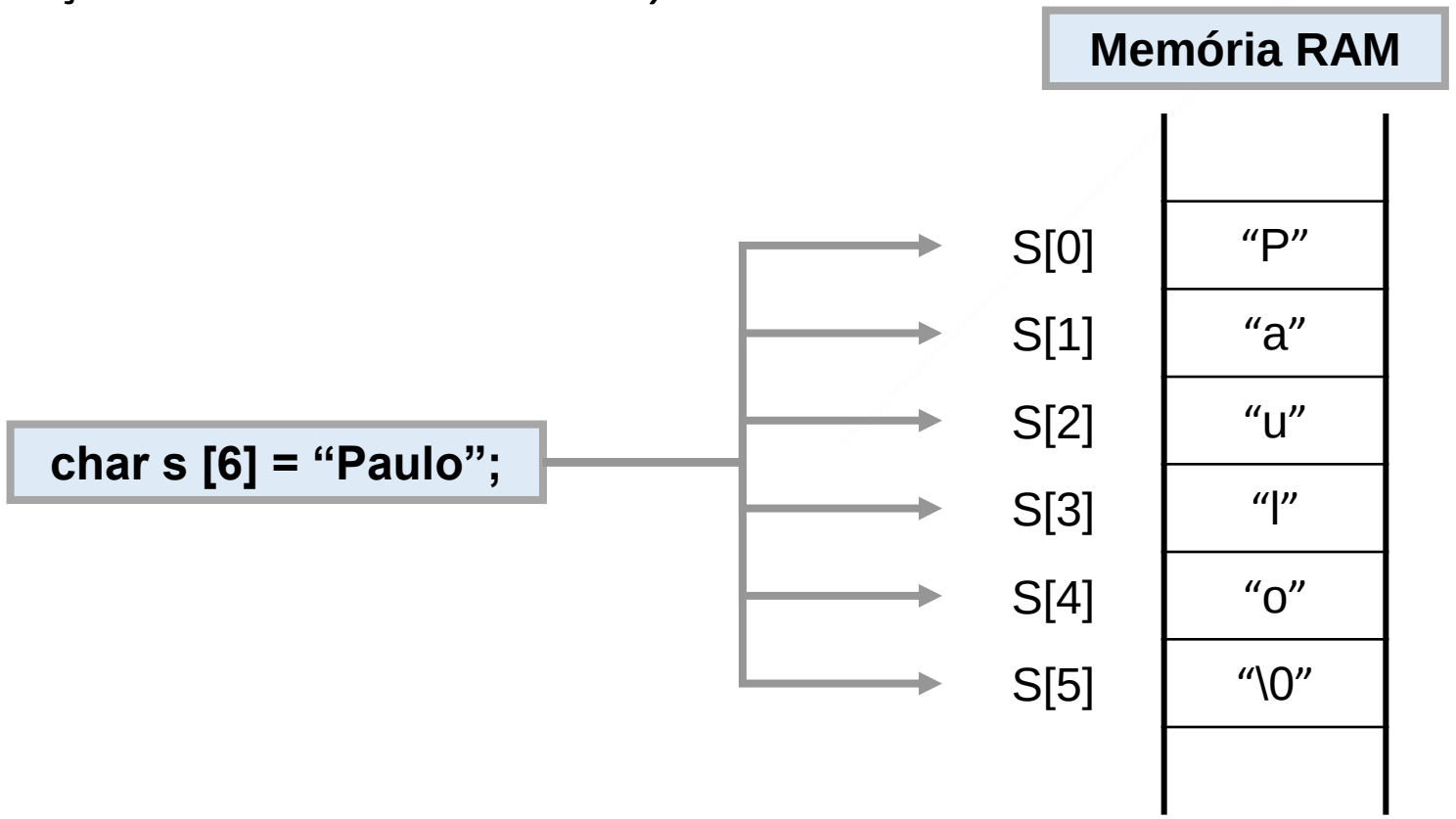
**char t [5]= “Teste”** → Declara um *array* t, do tipo *char* (caracteres) iniciado com uma *string* com 5 elementos.

**char t[4] = “Teste”** → ERRO.

- Este erro ocorre, pois como se trata de uma *string* com 5 caracteres, o compilador necessita de reservar automaticamente um carácter especial “\0” para o final da palavra.
- Assim sendo, o tamanho do *array* deve ser o número de caracteres que precisamos, mais um.

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- Representação na memória de um *array* de 6 elementos *char*.



UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Exemplo**

```
…
char nome[10];
        cout << "Escreva o seu nome: ";
        cin >>nome;
        cout << "Ola, "<<nome <<", bem vindo a aula!\n";
..
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Funções predefinidas para operar com *strings***

  - Bibliotecas

**<stdio.h> ou <cstdio>** → Estas bibliotecas permitem a utilização da função *gets().* Esta função permite que se incluam espaços dentro das *strings.*
*gets (nome)*

```
…
#include <cstdio>
…
      char nome[6];
      cout << "Escreva o seu nome: ";
      gets(nome);
      cout << "Ola, "<<nome <<", bem vindo!\n";
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Funções predefinidas para operar com *strings***

  - Bibliotecas

**<string>** → A biblioteca permite a declaração *string = st*. Com isto, podemos atribui directamente uma *string* à variável *st.*

```
#include <string>
…
      string st;
      string vossa_st;
      frase = "Escreve algo, sobre algo: ";
      cout << st ;
      cin >> vossa_st;
      cout << vossa_st << "!\n";
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Funções predefinidas para operar com *strings***
  - **strcpy()** – copia strings

    - O comando “cls” na linha de comando, serve para limpar o ecrã.

```
…
char nome[20];
      cout << "Escreve o teu nome: ";
      cin >> nome;
      cout << nome <<"\n";
      system("pause");
      system("cls");
      strcpy(nome, "Paulo");
      cout << nome <<"\n";
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Funções predefinidas para operar com *strings***
  - **strcat()** – acrescenta strings
  - **strcmp()** – compara strings
  - **strlen()** – retorna o tamanho da string

```
…
    char teu_nome[20];
    char nome[20] = "Josefina e ";
    cout << "Escreve o teu nome: ";
    cin >> teu_nome;
    strcat(nome, teu_nome);
    cout << nome <<"!\n";
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Funções predefinidas para operar com *strings***

```
…
#include <cstdio>
…
char nome[20] = "paulo";
char teu_nome[20];
cout << "Escreve o teu nome: ";
gets (teu_nome);
if(strcmp (teu_nome, nome))
cout << "O teu nome e diferente do criador!";
else cout << "Tens o mesmo nome que o criador!\n";
…
```

UNIDADE CURRICULAR : PROGRAMAÇÃO ORIENTADA A OBJECTOS

- **Funções predefinidas para operar com *strings***

```
…
#include <cstdio>
…
      char nome[20];
      cout << "Escreve o teu nome: ";
      gets(nome);
      cout << “O teu nome tem "<< strlen(nome) <<" caracteres! ";
…
```

UNIVERSIDADE da MADEIRA

CTeSP CURSOS TÉCNICOS SUPERIORES PROFISSIONAIS