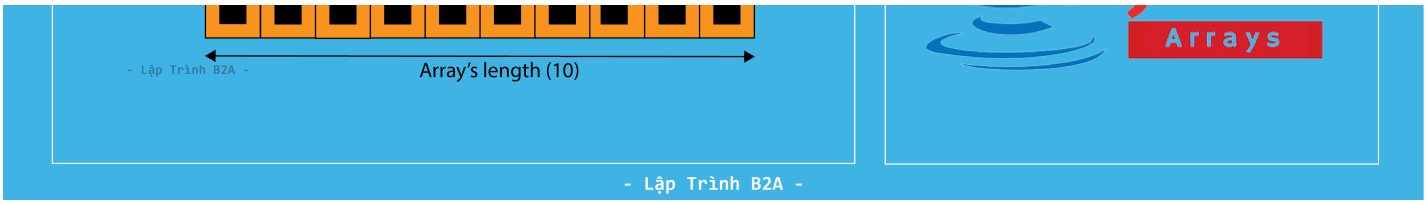

Arrays

Page

- Cada variável só pode armazenar um valor
- Mas, por vezes, era conveniente armazenar vários valores numa única variável
- Quando for necessário armazenar vários valores numa única variável deve-se usar um array (ou Array)

> Um array é um conjunto de elementos consecutivos, todos do mesmo tipo, e que podem ser acedidos individualmente através de um único nome (ou identificador)

- A sintaxe para declaração de um array é:

```java
tipo nomeVariavel[] = new tipo[nElems]; || tipo []nomeVariavel = new tipo[nElems];
```

- O `tipo` indica o tipo de dados de TODOS os elementos do array
- o `nomeVariavel` é o nome pelo qual se vai poder aceder ao array
- `nElems` indica quantos elementos tem o array

```java
int nCarrosVendidos[] = new int[12];
float []lucroMensal = new float[12];
int numTrabalhadoresFalta[] = new int[12];
double []milValores = new double[1000];
```

- Cada valor dentro de um array é identificado por um número - o índice
- Os índices começam sempre em zero
  - O último indíce válido é o n-1 (n = nº de elementos)

- Para aceder ao valor de um dado índice basta usar esse índice dentro de `[ ]`
- Par alterar o programa dos carros é só:

```java
int nCarrosVendidos = new int[12];

//...
```

```
for (int i = 0; i < 12; i++) {
    printf("O número de carros vendidos %d foi %d", i+1, nCarrosVendidos[i]);
    // o i+1 é apenas para o número do mês aparecer entre 1 e 12 e não entre 0 e 11
}
```

- O índice a colocar pode ser uma variável, uma constante ou mesmo o resultado de uma expressão, tem é de ser um valor inteiro:

```
Java
umArray[i] = 10          // Altera o elemento de índice i
umArray[2] = 25          // Altera o elemento de índice 2
umArray[i+3*j] = 23      // Altera o elemento de índice i+3*j
```

- Cada array tem associado a si uma dimensão
- Para saber qual a dimensão do array pode-se usar a variável length, usando a sintaxe: `nomeArray.length`

```
Java
int nCarrosVendidos[] = new int[ 12 ];

// ...

for( int i = 0; i < nCarrosVendidos.lenght; i++) {
    printf("O número de carros vendidos no mês %d foi %d", i+1, nCarrosVendidos[ i ] );
// o i+1 é apenas para o número do mês aparecer entre 1 e 12 e não entre 0 e 11
}
```

- Quando se declara um array as suas posições são inicializadas automaticamente com os valores por defeito
- Mas podem-se inicializar Arrays na sua declaração usando a sintaxe: `tipo nome[] = {valor0, valor1, …, valorn}`

```
Java
int numDiasMes[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

char vogais[] = {'a', 'e', 'i', 'o', 'u' };
```

⚠ Quando se inicializa automaticamente um array não é necessário indicar a sua dimensão, assume-se que é do tamanho do array de inicialização

```
Java
int nDiasMes[ ] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};   // array com 13 elementos

char vogais[ ] = {'a', 'e', 'i', 'o', 'u' };                              // array com 5 elementos
```

⚠ É necessário chamar a atenção que a atribuição de vários valores a um array SÓ É POSSÍVEL NA INICIALIZAÇÃO. Depois disso, terá de ser atribuído um valor a cada elemento INDIVIDUALMENTE

```
Java
int umArray[] = { 13, 45, 6, 8, 15, 10 };

umArray[ 0 ] = 2000; // correcto, alterar o elemento no índice 0

umArray = {12, 24, 45, 67, 98, 12}; // errado, NÃO se pode fazer
```

❌ Quando se tenta aceder a um índice inválido a aplicação "estoura" dando uma indicação de erro. Essa indicação mostra a linha em que foi o erro e qual o índice responsável

```
Java
public class ExplodeArray {
    public static void main( String []args ){
```

```
        int umarray[ ] = { 13, 45, 6, 8, 15, 10 };

        umarray[ 0 ] = 2000;     // correcto, alterar o elemento no índice 0

        umarray[ 10 ] = 5;       // errado, o array só tem 6 posições!
                                 // o último índice válido é o 5.
    }                            // a aplicação estoura quando isto acontece

}

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
at p1.ExplodeArray.main(ExplodeArray.java:7)
```

- Ler e escrever Arrays:
  - Não há qualquer forma de ler ou escrever um array no seu todo
    - Cada elemento tem de ser lido/escrito INDIVIDUALMENTE
  - O programador é que terá de elaborar código para o fazer

```java
int nCarrosVendidos[ ] = new int[ 12 ];

for( int i=0; i < nCarrosvendidos.length; i++) {
    System.out.printf("Quantos carros se venderam no mês %d? ", i+1);
    nCarrosVendidos[i] = input.nextInt( );
}

// ...

for( int i=0; i< nCarrosvendidos.length; i++)
    System.out.printf("O número de carros vendidos no mês %d foi %d", i+1, nCarrosVendidos[ i ] );
```

Arrays Multidimensionais

- Um array pode ter mais de uma dimensão. Neste caso usa-se a sintaxe: `tipo nomeArray[]…[] = new tipo [dim1]…[dimN];`

```java
int matriz3x3[][] = new int[3][3]; // array de 3x3

int plano3D[][][] = new int[2][3][4]; // array de 2x3x4
```

- Na prática um array multidimensional é um array de Arrays
- Para aceder a um elemento de um array multidimensional tem-se de colocar todos os índices

```java
int matriz3x3[][] = new int[3][3]; // array de 3x3

System.out.printf("O valor de (1,2) é %d", matriz3x3[1][2]);
```

- Para inicializar um array multidimensional, convém não esquecer que se trata de uma array de Arrays

```java
int matriz3x3[][] = { {3, 1, 2}, {3, 2, 3}, {7, 6, 2} };

int matriz3x3[][] = { {3, 1, 2},
                      {3, 2, 3},
                      {7, 6, 2}

                    };
```

Strings