DIN Departamento de Informática

UEM Universidade Estadual de Maringá

Arrays (parte 2)

Lilian Passos Scatalon
lpscatalon2@uem.br

Passando arrays para métodos

Arrays podem ser passados como argumentos para métodos

```
double[] temperaturasDoDia = new double[24];
modificaArray(temperaturasDoDia);
```

A lista de parâmetros do método deve incluir um parâmetro do tipo array

```
void modificaArray(double[] b)
```

```java
public class PassaArray {

	public static void main(String[] args) {
		int[] array = {1, 2, 3, 4, 5};

		System.out.printf( "Os valores originais do vetor são:%n");
		for (int valor : array) {
			System.out.printf(" %d", valor);
		}
		modificaArray(array); // passa a referência do array
		System.out.printf("%n%nOs valores modificados do array são:%n");
		for (int valor : array) {
			System.out.printf(" %d", valor);
		}

		System.out.printf("array[3] antes de modificaElemento: %d%n", array[3]);
		modificaElemento(array[3]); // tentativa de modificar array[3]
		System.out.printf("array[3] depois de modificaElemento: %d%n", array[3]);
	}


	// multiplica cada elemento do array por 2
	public static void modificaArray(int[] array2) {
		for (int i = 0; i < array2.length; i++) {
			array2[i] *= 2;
		}
	}

	// multiplica o valor do parâmetro por 2
	public static void modificaElemento(int elemento) {
		elemento *= 2;
	}

}
```

Passagem de argumentos

Passagem por valor

- uma cópia do valor do argumento é passada ao método chamado
- o método exclusivamente com a cópia
- mudanças no parâmetro (que recebeu a cópia) não afetam o argumento passado

Passagem de argumentos

Passagem por referência

- a referência (endereço) do argumento é passada ao parâmetro
- o método chamado pode acessar e modificar o valor argumento diretamente
- elimina a necessidade de copiar grandes quantidades de dados

Passagem de argumentos

Em Java todos os argumentos são passados por valor

- Cópias de valores de tipo primitivo (por ex., valores do tipo int)
- Cópias de referências a objetos (por ex., arrays)

Mesmo que a referência a um objeto seja passada por valor, o método ainda pode interagir com o objeto referenciado

Passagem de argumentos

O parâmetro e o argumento possuem ambos a referência ao mesmo objeto em memória

```
public static void modificaArray(int[] array2){
...
}

modificaArray(array);
```

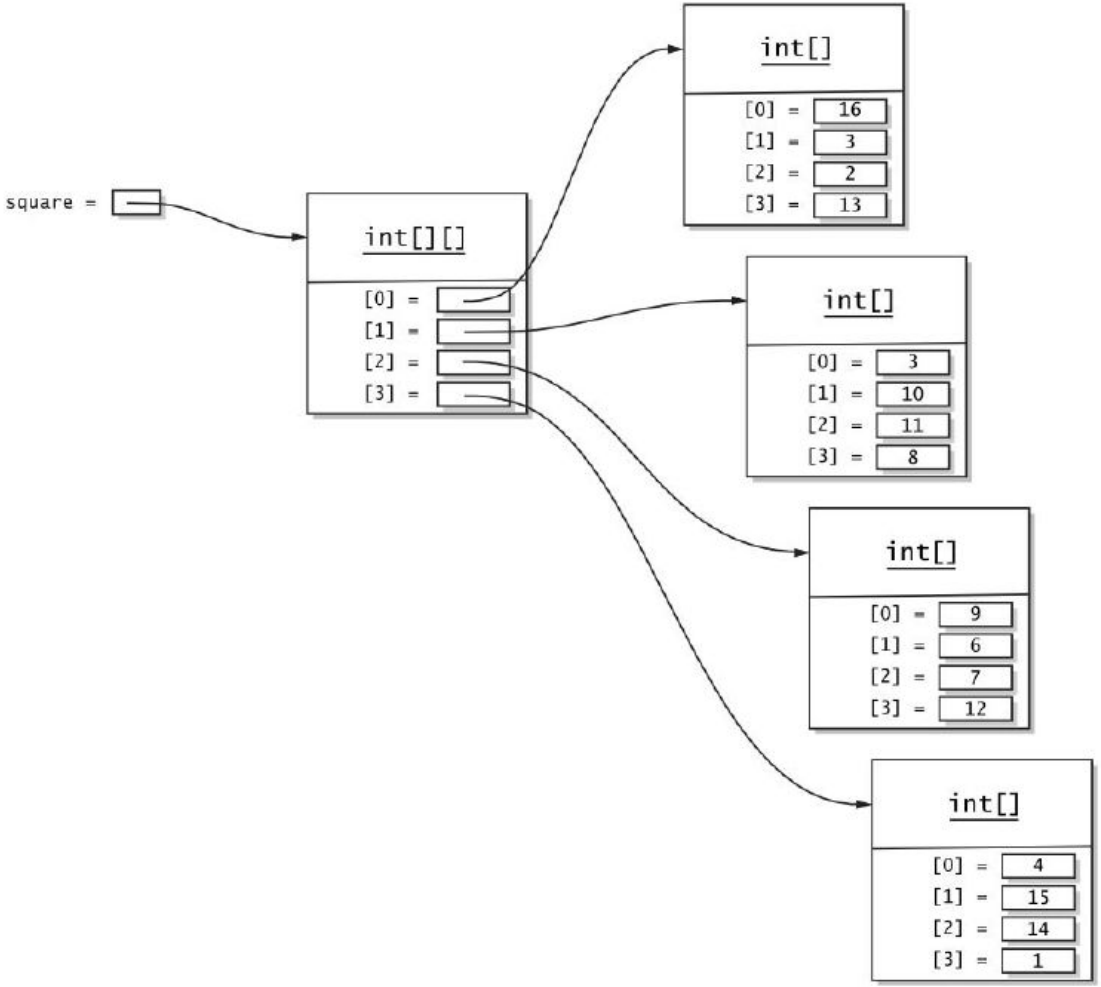
Arrays multidimensionais

Java não tem arrays multidimensionais verdadeiros

Eles são implementados como arrays de arrays

```
int[][] square = {
        { 16, 3, 2, 13 },
        { 5, 10, 11, 8 },
        { 9, 6, 7, 12 },
        { 4, 15, 14, 1}
};
```

square é um array unidimensional de arrays int[]


square =
int[][]
[0] =
[1] =
[2] =
[3] =
int[]
[0] = 16
[1] = 3
[2] = 2
[3] = 13
int[]
[0] = 3
[1] = 10
[2] = 11
[3] = 8
int[]
[0] = 9
[1] = 6
[2] = 7
[3] = 12
int[]
[0] = 4
[1] = 15
[2] = 14
[3] = 1

Arrays multidimensionais

Não há restrição para que os tamanhos das linhas sejam iguais

Por exemplo, é possível armazenar o triângulo de Pascal

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
...
```

Arrays multidimensionais

Construir um array de n linhas

```
int[][] triangulo = new int[n][];
```

Em seguida, construir cada linha com um loop e preencher

```
for (int i = 0; i < n; i++) {
        triangulo[i] = new int[i + 1];
        triangulo[i][0] = 1;
        triangulo[i][i] = 1;
        for (int j = 1; j < i; j++) {
                triangulo[i][j] = triangulo[i - 1][j - 1] + triangulo[i - 1][j];
        }
}
```

Arrays multidimensionais

Para percorrer o array bidimensional, é preciso ter dois loops aninhados

```
for (int l = 0; l < triangulo.length; l++) {
        for (int c = 0; c < triangulo[r].length; c++) {
                System.out.printf("%4d", triangulo[l][c]);
        }
        System.out.println();
}
```

Que podem ser dois loops de enhanced for

```
for (int[] linha : triangulo) {
        for (int elemento : linha) {
                System.out.printf("%4d", elemento);
        }
        System.out.println();
}
```

Argumentos de linha de comando

É possível passar argumentos para uma aplicação por meio do parâmetro args do método main

- Exemplos de uso: para passar opções, nomes de arquivos, etc

args é um array de String

```
public static void main(String[] args) {
        ...
}
```

```java
public class InitArray {
	public static void main(String[] args) {

		if (args.length != 3) {
			System.out.printf("Erro: os argumentos devem incluir %n" +
			" tamanho do array, valor inicial e incremento.%n");
		}
		else {

			int tamanhoArray = Integer.parseInt(args[0]);
			int[] array = new int[tamanhoArray];

			int valorInicial = Integer.parseInt(args[1]);
			int incremento = Integer.parseInt(args[2]);

			for (int i = 0; i < array.length; i++) {
				array[i] = valorInicial + incremento * i;
			}

			System.out.print("[ ");
			for (int i = 0; i < array.length; i++) {
				System.out.printf("%3d ", array[i]);
			}
			System.out.print("]");
		}
	}
}
```

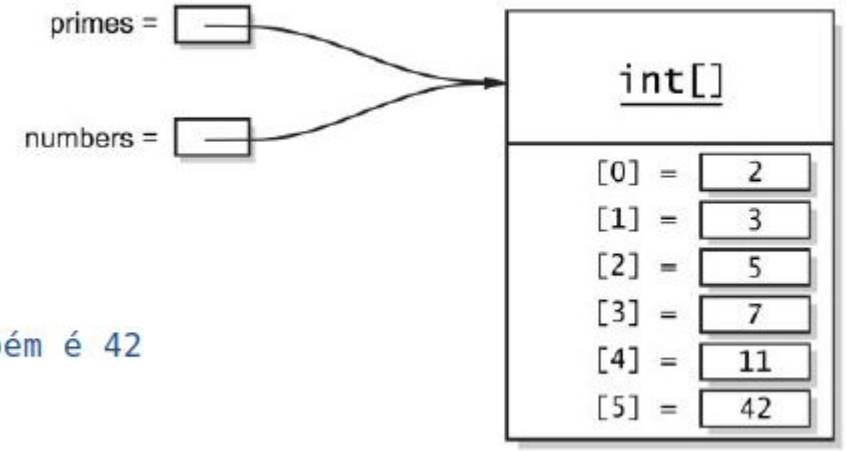
Classe Arrays

Inclui métodos estáticos para manipulações comuns de arrays

- sort para ordenar um array
- binarySearch para buscar em um array ordenado
- equals para comparar arrays
- fill para preencher os valores dos elementos
- copyOf para copiar arrays

Classe Arrays

É possível atribuir uma variável de array para outra, mas ambas vão apontar para o mesmo array




```
int[] numeros = primos;
numeros[5] = 42; // Agora primos[5] também é 42
```

Classe Arrays

Para evitar essa situação, usa-se o método copyOf

```
int[] numeros = Arrays.copyOf(primos, primos.length);
```

Bibliografia

Deitel, P.J. e Deitel, H.M. **Java How to Program: Late Objects**. 11th edition. Pearson, 2020.

C. S. Horstmann. **Core Java SE 9 for the Impatient**. Addison-Wesley, 2nd Edition, 2018.