

Estruturas de Dados
Ficha Laboratorial Nº 3
Engenharia Informática – Instituto Superior de Engenharia de Coimbra

1 - Construa um método que recebe como parâmetro um *array* genérico, bem como um elemento genérico a procurar, e devolve como resultado a indicação se o elemento indicado se encontra mais do que uma vez no *array*. O elemento deve ser identificado através de comparação por referência.

2- Construa um método que recebe como parâmetro um *array* genérico, bem como um elemento genérico a procurar, comparável com os elementos do *array*, e devolve como resultado a indicação se o elemento indicado se encontra mais do que uma vez no *array*. O elemento deve ser identificado através do método *compareTo*.

3 – Construa uma hierarquia *Figura / Rectangulo*, em que *Figura* implementa o interface *Comparable<Figura>*. A comparação entre duas figuras é feita pela sua área.

- a) Construa um método que recebe duas figuras e devolve o resultado da comparação.
- b) Construa um método que recebe um *Rectangulo* e um segundo objecto com o qual *rectangulo* é comparável e devolve o resultado da comparação. Verifique se o método funciona correctamente quando o segundo objecto é de qualquer um dos seguintes tipos: *Figura*, *Rectangulo*, uma classe *X* que seja *Comparable<Rectangulo>*, uma classe *Y* que seja *Comparable<Figura>*, e uma classe *Z* que seja *Comparable<Object>* (as classes *X*, *Y* e *Z* não são nem estão relacionadas com a classe *Figura* e *Rectangulo*)
- c) Construa um método que recebe um primeiro objecto qualquer e um segundo com o qual o primeiro é comparável.

4 - Construa um método estático que recebe dois parâmetros, sendo o primeiro um *array*. O método deve indicar se nesse *array* se encontra um elemento maior do que o segundo valor recebido por parâmetro.

Exemplo:

```
Integer m[]={3,2,6,3};
String n[]{"Ada", "Albino"};
System.out.println(search(m,2));    // true
System.out.println(search(n,"Francisco"));    // false
```

5 - Construa uma classe *Ponto* que armazena duas coordenadas numéricas, usando genéricos de forma adequada. O código seguinte ilustra o tipo de operações que devem ser possíveis:

```
Ponto<Integer, Integer> p=new Ponto<>(3,4);
Ponto<Number, Number> x=new Ponto<>(0,0);
System.out.println(p);    // imprime (3,4)
System.out.println(x);    // imprime (0,0)
x.copy(p);
System.out.println(x);    // imprime (3,4)
Ponto<String, Integer> erro=
    new Ponto<String,Integer>("olá",3);
// erro de compilação
```