

## ESQUEMA AULA PRÁTICA 6

- **Listas Dinâmicas**

A principal limitação dos arrays resulta do seu carácter estático. É necessário estabelecer a dimensão do array aquando da sua definição e não é possível exceder este limite máximo.

A classe `ArrayList` disponível no pacote `java.util` distingue-se dos arrays pelas seguintes características:

- Uma `ArrayList` pode crescer ou decrescer de tamanho;

Uma `ArrayList` armazena objectos (os tipos primitivos são “embrulhados” em objectos... Lembram-se das classes `Integer`, `Double`,...?).

- Uma `ArrayList` pode conter objectos de diferentes tipos.

Em conclusão, a classe `ArrayList` implementa uma abstracção de dados que representa uma estrutura linear indexada a partir do índice 0 (deste ponto de vista, análoga ao array) sem limite de dimensão.

No final desta ficha tem alguns dos métodos mais usados da classe `ArrayList`.

1 - Estude o programa abaixo e tente prever o seu output.

```
public static void main (String [] args) {  
  
    ArrayList lista = new ArrayList();  
    lista.add( "Maria");  
    lista.add( "João");  
    String s = (String) lista.get(0);  
    System.out.println (lista.toString() + " , "+ s);  
}
```

2 – Implemente o programa e verifique a sua resposta.

A `ArrayList` pode conter objectos de qualquer tipo, não havendo verificação de tipos. A partir da versão 5 do Java, a verificação de tipos pode ser feita durante a compilação, usando **tipos genéricos**:

Um tipo genérico é um tipo referenciado que usa na sua definição um ou mais tipos de dados como parâmetros.

Por exemplo, o tipo `ArrayList <E>` em que **E** pode ser qualquer classe (ou interface!!). A instanciação de um tipo genérico para um valor concreto de E, dá origem a um tipo parametrizado.

Por exemplo, o código :

```
ArrayList <String> lista1;  
lista1 = new ArrayList <String> ();  
lista1.add ("Joana");  
lista1.add ("Manuel");  
  
String ss = lista1.get(0);  
System.out.println (lista1.toString() + " ," + ss);
```

Usa o tipo Parametrizado `ArrayList <String>`, com verificação estática de tipos (isto é, verificação de tipos em tempo de compilação).

3 - Construa a classe Biblioteca que tem como atributos o nome da biblioteca, e uma lista com o nome dos livros de que dispõe (objeto do tipo `ArrayList<String>`).

- Defina o construtor sem parâmetros.
- Defina o construtor que receba como parâmetro o nome da biblioteca.
- Defina os getters e setters.
- Redefina o método `toString`.
- Redefina o método "boolean equals (Object o)".
- Redefina o método "Object clone()".
- Defina um método que dado o nome de um livro verifique se este faz parte da lista.
- Defina um método que adicione um novo livro à biblioteca caso não exista ainda.
- Defina um método que permita remover um livro da biblioteca.

4 - Construa uma classe de teste para a sua Biblioteca.

5 – Supondo a classe Telefone definida na aula teórica, considere a classe Exemplo:

```
public class Exemplo{  
    private int n  
    private String s;  
    private ArrayList<Integer> listaX ;  
    private ArrayList<Telefone> listaT;
```

- a) Defina o construtor de omissão;
- b) Construa os getters e setters;
- c) Construa o método equals;
- d) Construa o método clone;
- e) Construa o método toString.

### API da classe `java.util.ArrayList`:

```
ArrayList() // construtor vazio, dimensão inicial zero.

boolean add(Object element)
// adiciona o elemento especificado ao final da lista

void add( int index, Object obj)
//insere o elemento especificado na posição index

Object remove(int index )//remove o elemento da posiçã index

boolean remove( Object o)
//remove a primeira ocorrência do objecto dado como parâmetro

Object set (int position, Object obj )
// substitui o elemento da posição index pelo elemento dado

Object get (int position)//devolve o elemento da posição index

void clear() // remove todos os elementos da lista

Object clone() // devolve uma cópia da lista

boolean contains(Object element)
// devolve true se a lista contém o elemento especificado

boolean equals ( Object obj)
// permite comparar duas listas

int indexOf(Object element)
// procura o índice da 1ª ocorrência de elemento

boolean isEmpty() // verifica se a lista não tem componentes

int size() // devolve a dimensão actual

String toString ()
```