

# Ficha Prática 3a

António Nestor Ribeiro, António Fernandes, Mário Martins  
{anr,af,mmm}@di.uminho.pt  
PIIV (LESI/LMCC)

2004/05

## Objectivos

1. Classes e instâncias.Métodos e variáveis de instância
2. Construtores.Usos de `this`.
3. Classes de Teste e BlueJ
4. Exercícios

## Exercícios

1. Considere que pretende implementar uma mini agenda pessoal de endereços de e-mail.

Para armazenar a informação de um contacto crie uma classe `Contacto` com duas variáveis de instância: `nome` e `e-mail`;

Implemente os seguintes métodos nesta classe:

```
Contacto()  
Contacto(String nome, String e-mail)  
Contacto(Contacto outro)
```

```
void setNome(String nome);
```

```
/* só altere o email se o parâmetro contiver uma e só uma vez o símbolo "@"  
void setEmail(String email);
```

```

String getNome();
String getEmail();

/* métodos que verificam se o parâmetro é igual à respectiva variável de in
boolean isMyName(String nome);
boolean isMyEmail(String email);

/* Compare as suas implementações para os seguintes métodos */
boolean equals(Object); /* esta é a versão standard */
boolean equals(Contacto);

/* utilize a classe StringBuffer internamente para criar o valor a retornar
String toString();

```

2. Analise a classe `StringBuffer`, que permite a implementação de uma `String` de forma eficiente.

Permite o mesmo tipo de operações mas sem criar novas strings para cada alteração. É utilizada internamente pelo compilador ao executar operações de concatenação de Strings do tipo:

```

x = "Viva o " + language + " " + version

/* tudo na mesma linha. A evitar ;) */
x = new StringBuffer().append("Viva o ").append(language).append(" ").append

```

Aguns métodos úteis:

- `append(...)`
- `toString`
- `int indexOf(String a);` // devolve o índice do início da String a na instância.
- `int indexOf(String a, int fromIndex);` // faz o mesmo que o anterior, mas a partir de um determinado índice
- `length()` // devolve o número de caracteres da String

Exemplo de utilização:

```

StringBuffer a = new StringBuffer("Viva" + " o");
a.append(" java");

```

3. Implemente agora uma `Agenda` num Array de contactos com um limite de 100 entradas. Na agenda podem existir nomes repetidos, mas não endereços de email repetidos. Pretende-se implementar os seguintes métodos:

```
Agenda();

void add(Contacto c); /* adiciona um contacto ao fim da lista */
void add(String nome, String email);

void remove(String email);

/* métodos de procura */
boolean existe(Contacto c);
String getNome(String email);

/* devolve o número de emails associado a um determinado nome */
String numEmails(String nome);

/* altera o nome do contacto com o email passado como parâmetro */
void setNome(String nome, String email);

/* determina se duas agendas são iguais, ou seja quando contém o mesmo
conjunto de contactos, mas não necessariamente pela mesma ordem */
boolean equals(Object c) ;

/* devolve uma agenda ordenada por nome */
Agenda ordena();
```

Nota: para a comparação de nomes utilize o método `compareTo` da classe `String`. Este método compara strings e da seguinte forma:

```
int x = a.compareTo(b)

x negativo => a < b
x = 0 => a = b
x positivo => a > b
```