

Exercícios Preparação AIP

Exercício E1

¹ Os ficheiros `JogaJogoDoGalo.java` e `jogos/JogoDoGalo.java` definem um programa e um módulo para implementar um “jogo-do-galo” mas onde, propositadamente, foram inseridos vários erros.

- a. Corrija o módulo `JogoDoGalo.java` de forma a eliminar os erros sintácticos (de compilação) do programa.

- Pode compilar com o comando: `javac JogaJogoDoGalo.java`

- b. O programa principal `JogaJogoDoGalo.java` contém também um erro semântico. Detecte-o e corrija-o.

- Pode executar o seu programa com: `java -ea JogaJogoDoGalo`
- Pode executar uma versão correcta com: `java -ea -jar JogaJogoDoGalo.jar`

	1	2	3
1	X		0
	----	+	----
2			X
	----	+	----
3			X

Jogador X ganhou!

- c. Torne o programa principal robusto na utilização do módulo (não é necessário usar Excepções).
- d. Altere o programa `JogaJogoDoGalo.java` de forma a realizar campeonatos de até 10 jogos, terminando quando um dos jogadores atinja 3 vitórias. No fim de cada jogo deve indicar a pontuação de cada jogador.

Exercício E2

² O programa `P1.java` pretende implementar uma versão simples do jogo de memória de descoberta de pares. Neste jogo existe um tabuleiro rectangular onde estão escondidos um número par de símbolos (no caso, letras). O objectivo é descobrir todos os pares de símbolos. Sempre que se tenta descobrir dois símbolos que não sejam iguais, estes apenas ficam visíveis num curto período de tempo. O jogo termina quando todos os pares forem

¹Problema de avaliação AIP em 2009-2010.

²Problema de avaliação AIP em 2011-2012.

descobertos, sendo a pontuação final determinada pelo número de jogadas (quantas mais forem, pior a pontuação)³.

Pode jogar uma versão correcta deste jogo com o comando:

```
java -ea -jar P1.jar <arg> ...
```

O programa principal **P1.java** e o módulo **Board** tentam implementar esse jogo, mas contêm diversos erros. Pretende-se que:

- a) Corrija o módulo de forma a eliminar todos os seus erros sintácticos e opções erradas de construção do módulo.

(Pode compilar com: `javac P1.java`)

- b) Detecte e corrija o erro semântico no programa principal.

(Teste o seu programa com o comando: `java -ea P1` e compare com a versão correcta.)

- c) Torne o programa principal robusto na utilização do módulo e na interacção com o utilizador.

- d) Acrescente ao módulo **Board** um método `pairsFound` que indique o número de pares já descobertos.

(Para testar o método, pode invocá-lo do programa principal sempre que houver uma jogada.)

- e) Altere o programa principal de forma a que dois, ou mais, jogadores possam jogar vários jogos simultaneamente (note que cada jogador terá um jogo diferente dos restantes). O objectivo é competirem entre si, alternando entre todos em cada jogada, para ver quem termina o jogo em menos jogadas (os nomes dos jogadores podem ser passados como argumentos do programa). Quando todos os jogadores terminarem os seus jogos, o programa termina indicando o ranking de todos os jogadores (a ordem da tabela não interessa).

Exercício E3

⁴ Crie um programa **P2.java** que, dado um directório como argumento, liste o número de ficheiros contidos nesse directório e em cada um dos seus sub-directórios sucessivamente. Por número de ficheiros entende-se o número de ficheiros e directórios. Por exemplo, se existir um directório com o conteúdo representado abaixo,

³Sugere-se que teste o programa para tabuleiros pequenos tipo 2×2 para gerir o tempo disponível.

⁴Problema de avaliação AIP em 2009-2010.

```

d1
|-- d2
|   |-- d4
|   |   '-- f4
|   '-- f3
|-- d3
|-- f1
'-- f2

```

então o programa:

```
java -ea P2 d1
```

deverá produzir um resultado idêntico a este:

```

d1: 4 files
d1/d2: 2 files
d1/d2/d4: 1 file
d1/d3: 0 files

```

Pode executar uma versão correcta do programa com: `java -ea -jar P2.jar d1`

Exercício E4

Implemente uma função recursiva **invert** que recebendo uma **String** como argumento a devolve invertida.

Por exemplo, a invocação do programa:

```
java -ea InvertString abc 1234 5 321
```

deve ter como resultado:

```

abc -> cba
1234 -> 4321
5 -> 5
321 -> 123

```

Exercício E5

Implemente uma função recursiva **factors** que recebendo um número inteiro como argumento devolve uma **String** com o produto dos seus factores.

Por exemplo, a invocação do programa:

```
java -ea Factors 0 1 10 4 10002
```

deve ter como resultado:

```

0 = 0
1 = 1
10 = 2 * 5
4 = 2 * 2
10002 = 2 * 3 * 1667

```

