IP – Semana 1 – Lista de Problemas (alunos em 1ª inscrição)

Miguel Goulão

miguel.goulao@di.fct.unl.pt

Objectivos

- No final da 1ª semana, o aluno deverá ser capaz de:
  - A partir de uma especificação simples, em língua natural, propor uma lista de operações a implementar numa classe
  - Usar o BlueJ na construção de classes simples, definindo:
    - Variáveis de instância (privadas)
    - Construtor de instâncias da classe (público)
    - Operações modificadoras (públicas)
    - Operações de consulta (públicas)

Especificação de interfaces

CRONÓMETRO
MÁQUINA FOTOGRÁFICA

Cronómetro

- Recorde o exemplo do leitor de MP3
- Inspirado nesse exemplo, e usando um estilo semelhante ao dos slides 6 e 7, proponha uma lista de operações (também conhecida como interface) de um cronómetro. Indique claramente:
  - Quais das operações propostas são modificadoras
  - Quais das operações propostas são de consulta
  - Que informação de estado julga ser necessário armazenar

Máquina fotográfica digital

- Recorde o exemplo do leitor de MP3
- Inspirado nesse exemplo, e usando um estilo semelhante ao dos slides 6 e 7, proponha uma lista de operações (também conhecida como interface) de uma máquina fotográfica digital simples, com flash, zoom e écran para ver as fotos. Indique claramente:
  - Quais das operações propostas são modificadoras
  - Quais das operações propostas são de consulta
  - Que informação de estado julga ser necessário armazenar

Introdução ao BlueJ

SENSORES DE MOVIMENTO FAIRPLAY

Sensores de movimento

Sensores de movimento

- Defina em Java uma classe `Sensor`, cujos objectos são sensores de movimento. Estes sensores são usados, por exemplo, em edifícios inteligentes, como forma de poupar energia, abrir automaticamente portas, ou detectar intrusos, em sistemas de entretenimento , etc
- Programe a sua classe no BlueJ
- Teste vários objectos da classe Sensor e verifique que se comportam tal como esperado.



Sensores de movimento

- Operações reconhecidas:

  `public boolean` `isMoving()`
  Consultar se há movimento detectado

  `public void` `move()`
  Detectar movimento

  `public void` `stop()`
  Detectar ausência de movimento

- Assuma que periodicamente são feitas chamadas aos métodos `move` e `stop`, consoante exista, ou não, movimento detectado nesse momento

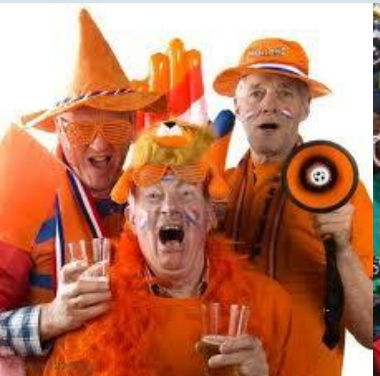
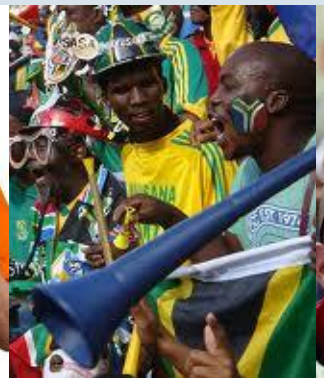
Sensores de movimento

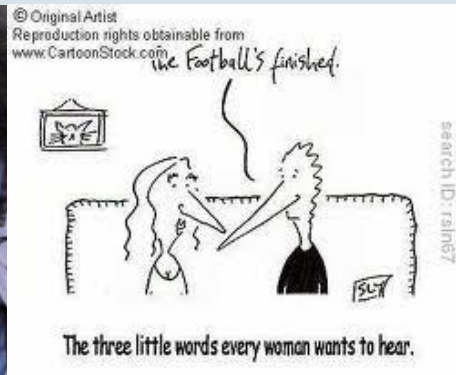
```
Sensor s1 = new Sensor();
Sensor s2 = new Sensor();
s1.isMoving()
false (boolean)
s2.isMoving()
false (boolean)
s1.move();
s1.isMoving()
true        (boolean)
s2.isMoving()
false       (boolean)
s2.stop();
s2.isMoving()
false (boolean)
s2.move();
s2.isMoving()
true        (boolean)
s2.stop();
s2.isMoving()
false (boolean)
```

FairPlay

FairPlay

- Defina em Java uma classe `FairPlay`, cujos objectos são usados para descobrir o estado de espírito de um adepto de futebol que apenas está feliz na sequência da vitória da própria equipa, ou do empate ou derrota de uma equipa rival. Os empates da própria equipa não mudam o estado de espírito do adepto e tudo o resto é um drama para o adepto (e um caso de polícia evidente), deixando o adepto deprimido.
- Programe a sua classe no BlueJ
- Teste vários objectos da classe `FairPlay` e verifique que se comportam tal como esperado.

FairPlay

- Operações reconhecidas (antes do primeiro jogo, o adepto está contente):

`public boolean` `happy()`
  Devolve `true` se o adepto está contente, false se está triste.

`public void` `weWon()`
  A equipa do adepto venceu, ele fica contente.

`public void` `weDrew()`
  A equipa do adepto empatou, o adepto fica triste.

`public void` `weLost()`
  Trágica derrota, o adepto fica triste.

`public void` `theyWon()`
  Lamentável vitória de um rival, o adepto fica triste.

`public void` `theyDrew()`
  Finalmente o rival perdeu pontos, o adepto fica contente.

`public void` `theyLost()`
  Glorioso momento, o rival está de rastos e o adepto fica contente.

FairPlay

```
FairPlay b = new FairPlay();
b.happy()
true      (boolean)
b.weLost();
b.happy()
false     (boolean)
b.theyLost();
b.happy()
true      (boolean)
b.weLost();
b.happy()
false     (boolean)
b.weDrew();
b.happy()
false     (boolean)
b.theyDrew();
b.happy()
true      (boolean)
```