Imperial College
London

Software Engineering 2: Object Oriented Programming

Programming test

Max Cattafi (`m.cattafi@imperial.ac.uk`)

It's in your interest to read the full specification at least once before you start writing any code.

A (sci-fi) robot factory makes various kinds of `Humanoid`s. For instance an `Android` is a `Humanoid`, a `Cyborg` is a `Humanoid` too etc.

It is important to be able to `test` the `vision` of `Humanoid`s, although how this is actually implemented depends on the specific type of `Humanoid`.

Both `Android`s and `Cyborg`s have a `BinocularVisionSystem`. In the case of `Android`s the `BinocularVisionSystem` is composed of two `Camera`s, in the case of `Cyborg`s it is composed of one `Camera` and one `Eye`.

Among `Humanoid`s there are also `CyclopicBot`s which do not have a `BinocularVisionSystem` but only one `Camera`.

The `Eye`s and the `Camera`s used in this factory have a `test` function, in our example let's assume that the test always succeeds and that it prints what kind of 'device' is being tested (e.g. 'camera' or 'eye') and that it is working. The whole `BinocularVisionSystem` has a `test` function too, which is performed through the `test`ing of both of its components.

Model the domain described in this text (following the specification and the hints) using classes (you will need also abstract and template ones), in suitable relationships of (for instance) inheritance and composition and with the required member functions.

Organize your code separating (if possible) header and implementation files (but you can have more than one class in the same header or implementation file).

Write a `main` to test your classes. In the `main` a few `Humanoid`s of different types should be created and the polymorphic behaviour of the test on their vision should be evidenced.

See also the following example (yours doesn't need to look identical) of execution log:

```
how many humanoids?
3
enter humanoid 0 ([c]yclopic[b]ot/[cy]borg/[an]droid):
cy
enter humanoid 1 ([c]yclopic[b]ot/[cy]borg/[an]droid):
an
enter humanoid 2 ([c]yclopic[b]ot/[cy]borg/[an]droid):
cb
testing vision of humanoid 0:
eye working
camera working
testing vision of humanoid 1:
camera working
camera working
testing vision of humanoid 2:
camera working
```