Refactoring the Bad Boids

Name: Stephen Morrell

Student number rmapjmo

Code of the module MPHYG001

## Smells

Lists by name the code smells identified refactorings used, making reference

to the git commit log [2 marks]

|  |  |  |
| --- | --- | --- |
| **Smell** | **Refactor** | **Commit message** |
| Raw numbers appear in your code | Replace magic numbers with constants  Replace constants with a configuration file | parameters to config file, added non-standard initiation tests first refactor of match\_speed |
| Fragments of repeated code appear | Replace repeated code with a function | refactor velocity update, first refactor of match\_speed |
| Code needs a comment to explain what it is for | Change of variable/function/class name | refactor flee neighbours, refactor fly\_to\_middle, + many others |
| An expression becomes long |  | refactor flee neighbours |
| Loop variable is an integer from 1 to something | Replace loop with iterator |  |
| It feels like surely someone else must have done this at some point | Replace hand-written code with library code | used np.random and added x min max test. works ok |
| A function needs to work corresponding indices of several arrays: | Replace set of arrays with array of structures |  |
| You need to change your code file to explore different research scenarios |  | parameters to config file, added non-standard initiation tests |
| A global variable is assigned and then used inside a called function: | Replace global variables with function arguments | use global variables |
| Two neighbouring loops have the same for statement | Break a large function into smaller units | refactor flee neighbours |
| A function or subroutine no longer fits on a page in your editor | Separate code concepts into files or modules | moved functions to boid\_utilities |
| A line of code is indented more than three levels | Separate code concepts into files or modules | moved functions to boid\_utilities |
| A piece of code interacts with the surrounding code through just a few variables | Break a large function into smaller units | refactor flee neighbours  refactor velocity update  first refactor of match\_speed |
| You find it hard to locate a piece of code | Break a large function into smaller units | refactor flee neighbours  refactor velocity update  first refactor of match\_speed |
| You get a lot of version control conflicts |  |  |

## UML

Includes a UML diagram of the final class structure [1 mark]

## Refactoring

Discusses in your own words the advantages of a refactoring approach to improving code [1 marks]

*Refactoring makes code easier to read, understand and maintain. The many small changes add up to much more user-friendly code which is much easier to use. Debugging becomes simpler. It’s often difficult and sometimes impossible to write unit tests for code which hasn’t been well refactorer because the code isn’t broken down into discrete pieces each with its own function.*

## Problems

Discusses problems encountered during the project [1 mark]

There wasn’t always a 1:1 relationship between the smell / refactor and the commit. I.e. many commits dealt with multiple issues.

Absolute referencing to import the .yml files broke when improving the directory structure. Fortunately the os.path.join function allows ‘..’ as part of the path.