Third Programming Task

Advanced Object-oriented Programming

LVA 185.211
2023 S
TU Wien

Catching Mice in an Eiffel Game

Learn Eiffel and implement a game in Eiffel. The game shall resemble that of the second programming task, this is, the game need not be network-based, and only one player plays as a cat. It is recommended to use EiffelStudio, an IDE based on Eiffel. Be careful to download the open-source version, not the commercial one. All team members shall work on this task to get their own Eiffel experiences.

There is support for windows with graphical content, but it may be difficult to find appropriate libraries and documentation. As an alternative you can write the game such that it runs on a (virtual) terminal with text output only: Repeatedly clear the screen and write out the whole playing field at an appropriate frequency.

Please try out the specific features of Eiffel to answer the following questions (in the Abgabegespräch):

- How much work is it to specify useful assertions in Eiffel?
- How important is the run-time penalty of assertion checking?
- How is it possible to specify pre-conditions that are, in some sense, in the subtype stronger than in the supertype although Eiffel does not allow us to redefine pre-conditions to become stronger? How is it possible to specify post-conditions that are, in some sense, in a subtype weaker than in the supertype although Eiffel does not allow us to redefine post-conditions to become weaker? (Yes, it is possible, although not obvious, and with an appropriate interpretation it is possible to do so without violating Design by Contract.)
- Eiffel supports co-variant input parameters. What are the advantages and disadvantages of this feature in practical programming?
- Which features of Eiffel would you like to see also in your favorite programming language? Which features of Eiffel would you rather avoid to use?