Respondent: **Hannes Karppila** Submitted on: Tuesday, 19 December 2017, 9:20 PM

Mid-term review of C++ project

Name of the project group evaluated

sim-city-1

This is a project self-evaluation

☐ Yes

C1.1: The implementation corresponds to the selected topic and scope. The extent of project is large enough to accommodate work for everyone (2 p)

The implementation corresponds to the topic and scope. From Git commits and GitLab issues we can see that everyone has contributed major amounts to the project, so the project was large enough. (2p)

C1.2: The class structure, information hiding and modularization is appropriate, and it is explained and justified in documentation. The file structure corresponds to the class structure (2 p)

The class structure and modularization are appropriate, and the file structure corresponds well to the class structure. Information hiding is done properly. There is still some unnecessarily duplicated code, for example the input parsing in the main function should probably be moved to a separate function. (2p)

C1.3: Use of at least one external library (in addition to C++ standard library). Comment the appropriateness of libraries and their use. (2 p)

SFML has been used for the graphical side of the project, and it has been used properly. (2p)

C2.1: Git is used appropriately (e.g., commits are logical and frequent enough, commit logs are descriptive) (2 p)

Git has been used regularly. Commits are logical and their messages are informative for the most part. Some inconsistencies in commit message styles. (1.5p)

C2.2: Make or Cmake (recommended) is used appropriately. The software should build easily using these tools without additional tricks. Nevertheless, instructions for building the project should be provided (1 p)

Make was used. Software building can be done easily, and the building instructions are present both in the documentation and in the README.md file. Running make worked without any problems. (1p)

C2.3: Work is distributed and organised well, everyone has a relevant role that matches his/her skills and contributes project (the distribution of roles needs to be described) (1 p)

Work distribution looks good. Everybody has had work to do. (1p)

C2.4: Issue tracker is used appropriately to assign new features and bug fixes (1 p)

Issue tracker was used for both bugs and features. Work was properly assigned. However, issue labels have not be used, which could have been nice for separating new features, modification and bugs (for example). (1p)

C2.5: Testing and quality assurance is appropriately done and documented. There should be a systematic method to ensure functionality (unit tests, valgrind for memory safety, separate test software and/or something else.) (1 p)

A method called "test" has been created, testing nearly everything in the program in small units. It would seem to catch at least most mistakes, but more complicated mistakes might be missed. (1p)

C3.1: C++ containers are used appropriately (including appropriate use of iterators), and justified (e.g., why certain type of container over another) (2 p)

std::vector and std::list have been used appropriately to store lists of items. The map/graph saves vertices in a std::vector<std::vector<Vertex>> which is logical.

Iterators have been widely used and in a correct manner.
(2p)

C3.2: Smart pointers are used in memory management, describe how (1 p)

Smart pointers are used for vehicles, to make their deletion as simple as removing them from a list, as well as in edges to store their vertices. (1p)

C3.3: C++ exception handling is used appropriately, describe how (1 p)

Exceptions are used for bounds checking in main method. This seems very forced, and using exceptions for goto-like control flow is usually considered an antipattern. Even using goto would have been a saner choice. On the bright, the side catch block actually contains the clean up code. (0.5p)

C3.4: Rule of three / rule of five is followed, describe how (1 p)

Proper destructors, copy-constructors, etc. have been implemented for the classes requiring them. (1p)

C3.5: Dynamic binding and virtual classes/functions are used, describe how (1 p)

Virtual class members and dynamic binding have both been used properly. (1p)

Other comments and feedback to the evaluated project group.

Some todo-comments were left over.

All in all, a very nice project. The visuals were especially nice.

If you did this review together with (some of) your group members, list the names of the group members here. Everyone needs to turn in a review, either separately or as a group.

We all did this together. Hannes, Iiro, Aleksanteri and Petteri.