Q learning 9

1. Main design (6 p)

•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)

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)

2p

•C1.3: Use of at least one external library (in addition to C++ standard library) (2 p)
- •The library should be either readily installed on Aalto Linux machines, or included as part of the project deliverable such that the program can be easily compiled as pulled directly from git, without additional steps.

2p, used Gtest (and box2d)

2. Working methods and tools (6 p)

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

2p, one branch but frequent and descriptive commit messages

•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)

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)

1p,

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

0.5p, 1 issue during project, also closing of the issue can be done by adding "Closes #numberofissue" to the commit message that fixes the issue.

•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)

1p, Gtest, no valgrind, from what I went through the source code the destructors seem to free the allocated heap appropriately

3. Use of C++ features (6 p)

•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)

2p, very sophisiticated use of iterators

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

0.5p, not needed

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

0.5p, sometimes the exception is thrown from the try block itself, this is sort of bad practice (create a class to handle the operation and throw inside it instead)

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

0p,

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

0.5p,

15 pts total

overall the code itself is very well done. One thing to remember is the rule of three (later rule of five);

“If a class defines one of the following it should probably define all three:

- destructor
- copy constructor
- copy assignment”

peer review: 3pts (max 3pts)