COMPENG 2SH4 Project – Peer Evaluation

Your Team Members ___Nerjes Al-hollai Rameez Siddiqi_________

Team Members Evaluated ___Haocheng Li Joonseo Lee_____________

Provide your genuine and engineeringly verifiable feedback. Ungrounded claims will lead to deductions.

Part I: OOD Quality

1. **[6 marks]** OOD is about sensible code modularization. Looking at the header files of each object, can you easily interpret the possible behaviours of the objects involved in the program, and how they would interact with each other in the program? Comment on what you have observed, both positive and negative features.

a. **After looking at the header file for each object, it is evident that all the behaviours are easily interpreted. GameMechs and Player objects contain all proper mutators and accessors. The last two member names in objPosArrayList could be named in a better way as more context might need to be there for interpretation.**

b. **Overall, the header files are well constructed and easy to read.**

2. **[6 marks]** Examine the main logic in the main program loop. Can you easily interpret how the objects interact with each other in the program logic through the code? Comment on what you have observed, both positive and negative features.

A. **Overall, the main program loop seems to be well organized, however; the main issue is presented from the draw screen function. It seems to be heavily unintuitive as the logic used is quite confusing. I can hardly find a part in the draw screen logic code that seems well organized and easy to interpret.**

B. **Better construction is needed for the if statements as the conditionals are unintuitive.**

C. **The initialization of the variables and objects are well done as they are simple and all are used within the code which makes it easier to read;**

3. **[5 marks]** Quickly summarize in point form the pros and cons of the C++ OOD approach in the project versus the C procedural design approach in PPA3.

   a. To summarize, the C++ OOD approach comes with a lot of advantages as debugging the code becomes a lot easier. Also, the OOD approach makes the code much more intuitive and easy to read. The C procedural design approach in PPA3 also comes with an advantage because simply designing the code without wondering where certain classes were declared saves a lot of time.

Part II: Code Quality

1. **[5 marks]** Does the code offer sufficient comments, or deploys sufficient self-documenting coding style, to help you understand the code functionality more efficiently? If any shortcoming is observed, discuss how you would improve it.

- **Insufficient comments are deployed in certain areas of the code, especially in the main project file. More comments should have been added to the draw screen function code since none**

were added which hinders the understanding. Almost nothing could be understood by looking at the code, which is a disadvantage. There also seems to be outdated comments within the code player.h file which tells me that the updates might not have taken place. This also indicates a lack of organization. To improve, it seems like both team members had to give the code a look to see where comments could have been added to improve readability and where to remove outdated comments because they quite confusing.

2. **[4 marks]** Does the code follow good indentation, add sensible white spaces, and deploys newline formatting for better readability? If any shortcoming is observed, discuss how you would improve it.
   a. **The code does follow good indentation, especially for the main project file and GameMech file. The indentations were well implemented throughout the code and nothing seemed out of place.**
   b. **The curly brackets are well placed so nothing could be improved on here.**
   c. **There seems to be a lack of white spaces in the code which does reduce the readability of the code. It also makes the code look disorganized. To improve, more spaces should be added throughout the code and between if statements. The lack of white spaces in the code goes to show that no thought and effort was put into making the code look more readable.**

Part III: Quick Functional Evaluation

1. **[8 marks]** Does the Snake Game offer smooth, bug-free playing experience? Document any buggy features and use your COMPENG 2SH4 programming knowledge to propose the possible root cause and the potential debugging approaches you'd recommend the other team to deploy. (NOT a debugging report, just a technical user feedback)
   a. **The snake game does appear to offer a bug-free playing experience**
      i. **All features are implemented correctly**
      ii. **Proper exit message is displayed after self suicide is deployed**
      iii. **Collision detection works accordingly**
   b. **Bonus feature is implemented correctly and no errors found there either**

2. **[6 marks]** Does the Snake Game cause memory leak? If yes, provide a digest of the memory profiling report and identify the possible root cause of the memory leakage.

Dream memory indicates that there is 0 bytes of leak.

Part IV: Your Own Collaboration Experience (Ungraded)

1. Tell us about your experience in your first collaborated software development through this project – what was working and what wasn't. If you are a one-person team, tell us what you think may work better if you had a second collaborator working with you.

- Both team members followed the recommended iterative workflow