COMPENG 2SH4 Project – Peer Evaluation

Your Team Members	Samer Gebirrebbi, Manvendra Jani
Team Members Evaluated	Majid and Ali

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.

Pros	Cons
- Intuitive naming - Members of objects are allocated appropriately (private/public) - Adequate commenting	

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.

Pros	Cons
- Logic is easy to interpret and flows well - All elements seem to be in the correct place	- Initializing too many variables on global can cause memory issues, would be more beneficial to initialize variables only in the functions they are used in

3.
4. **[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.

Pros	Cons
The OOD approach helps us breakdown the code into individual parts allowing us to reduce code duplication (i.e you are not writing the same code again and again) and improve readability. It also increases code reusability and gives a modularity to your code	It can be quite tedious and has a slight learning curve

Another pro is that OOD allows for parallel development, as multiple people can work on different parts without great interference	

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.

   There is an adequate level of commenting that allows for clear understanding of the code, with perhaps even some over-commenting for certain sections that are self explanatory

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.

   There is no glaring issues with format, the code includes sensible amount of whitespace, enough to make it readable, but not too much. New lines are made as they should and there are no issues with readability

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)

   There are no noticeable bugs and the game is a smooth experience. The snake's size updates correctly, as does the score, and the snake moves through the boundaries and overlaps as expected.

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.

   There is no memory leak in the game

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.

   Working with another person had its challenges, but overall, it was a good experience. I think it helped us appreciate one of the major benefits of OOD, which is the fact that it allows for parallel work and smoother teamwork. Sometimes it could be a little tedious when we would try to edit the same portions of code, as one of us would have to delete or stash our code in order for the other person to git push and pull respectively, but this was rarely an inconvenience as we would usually be working on separate parts, so overall it was a fine experience and good practice for future projects.