**COMPENG 2SH4 Project – Peer Evaluation [25 Marks]**

Your Team Members \_\_\_\_\_\_lopeze5\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_loho\_\_\_\_\_

Team Members Evaluated \_\_\_\_\_\_\_wonge84\_\_\_\_\_\_\_\_ \_\_\_\_\_\_sifarf\_\_\_\_\_

Provide your genuine and engineeringly verifiable feedback. Ungrounded claims will lead to deductions. Completing the peer code evaluation on time will earn your team a total of **25 marks**. Do not exceed 2 paragraphs per question.

**Peer Code Review: OOD Quality**

1. **[3 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.

I can easily interpret how the objects with each other in the program logic, I don’t have much of an issue reading and understanding what is happening in their code. One negative I found is the unnecessary use of local variables in the draw function. Writing the chain of function names throughout makes the code more readable, and shows the reader exactly where data is coming from. This is also a long function, and as I read through I find that I forget which local variable is which and I have to scroll through and find where it was defined and what it is, which is an issue that could easily be avoided by just never making them. Other than that the code is very clear and easy to follow.

1. **[3 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.

I think the C++ OOD approach has almost all of the pros. The OOD approach breaks apart code into smaller and easier to find / read sections. For example, when a function called from an instance of food is used, it is intuitive to go to the food file and see the function definition. It is also easier to keep track of where data is being changed through the use of getters and setters as opposed to just changing global variables like the C approach does. I think readability is the biggest C++ OOD pro overall, as the main game file ends up just being mostly function calls. With descriptive names, it is often easy to tell what a function does, but if you are not sure, you can just open the file of the class the function is called from to find it. You also know that a function called from a class is related to that class. The main pro of C is since all functions and variables are contained in one file, you don’t have to worry about scope, or trying to figure out how to get certain data. I think the pros of C++ are the cons of C, and vice versa.

**Peer Code Review: Code Quality**

1. **[3 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 are not many comments throughout, so I would maybe add a few more in more complicated areas for less experienced programmers. However, I think their code is self-documenting, and I never really had any issue understanding any of their code.

1. **[3 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.

I see no formatting issues, the code is clean, well organized, and easy to follow.

**Peer Code Review: Quick Functional Evaluation**

1. **[3 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 technical user feedback)

I did not observe any bugs, the code worked perfectly.

1. **[3 marks]** Does the Snake Game cause memory leak? If yes, provide a digest of the memory profiling report and identify the possible root cause(s) of the memory leakage.

The game does not cause memory leak.

**Project Reflection**

Recall the unusual objPos class design with the additional Pos struct. After reviewing the other team’s implementation in addition to your own, reflect on the following questions:

1. **[3 marks]** Do you think the compound object design of objPos class is sensible? Why or why not?

I think the compound object desion of objPos is not sensible. objPoss is a class which is a collection of data and functions that are related, while a struct is a collection of just related data. Since the class is already a collection of related data, there is no need to further collect data, it is intuitive for a position to have an x and y attribute, so there is no point including a collection of coordinates inside the collection of coordinates.

1. **[4 marks]** If yes, discuss about an alternative objPos class design that you believe is relatively counterintuitive than the one in this project. If not, explain how you’d improve the object design. You are expected to facilitate the discussion with UML diagram(s).

To improve the design I would remove the struct and make the x and y positions fields of the class. Here is the UML of how I would make this change: A screenshot of a computer program

Description automatically generated