COMPENG 2SH4 Project – Peer Evaluation [25 Marks]

Your Team Members	Ethan Su, 400521000 Dylan Manamendra 400509187
Team Members Evaluated	Marc Cosma (cosmam), Zaiyou He (he151)

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.

The code is easy to interpret, containing instantly understandable variable names. Interaction between objects is easily interpretable. Nothing in the main logic stands out as being a negative feature. Their design is not difficult to wrap your head around.

2. **[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.

Pros:

- The code becomes better organized
- Allows more freedom and diversification in what you can implement
- Allows the main program file to be easier to understand and read through (more organized)
- Much better for finely tuning and optimizing functions since they are separated

Cons:

- Code is spread out over multiple files, meaning not all data is readily available
- More complex to implement, requiring multiple files and more code overall
- Debugging is far more difficult to locate and diagnose.

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.

Code provides sufficient comments in most difficult to understand areas. The movePlayer() method is the only one which lacks this. Its wraparound logic could use some more comments to explain what is happening for each step.

2. **[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.

The if statement in GetInput(), Food destructor, movePlayer(), default GameMechs() and GameMechs() constructor, and the first half of objPosArrayList() are indented incorrectly. Although functional, it does take away from code organization.

Newline formatting could have been used in the GameMechs() constructor to better organize the code. Most of the variable assignment occurs on the same line, even though newline formatting was used in the default constructor.

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)

Gameplay experience was smooth with no issues, program did everything it was required to do up until iteration 3. The above and beyond features were not attempted.

2. **[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.

Zero bytes of memory leak were recorded after running DrMemory and the code appears to delete all memory on the heap during cleanup.

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?

After reviewing the other team's implementation and our own, we believe the compound object design to not be sensible. The compound object design of objPos class adds another layer of complexity to the OOP by introducing a separate method of retrieving the x and y coordinates of the object in the main program through the struct rather than a method for retrieving the values.

2. **[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).

We would improve the design of the objPos class by removing the struct and relocating the x and y coordinates to the public class in objPos as visualized in the UML diagram below.

objPos
+ pos: Pos*

+ symbol: char + x: int + y: int
+ objPos() + objPOs(x:int,y:int, sym:char) +setobjPOs(o:objPos) : void +setObjPos(x:int, y:int, sym:char) : void +getObjPos() : objPos const +getSymbol(): char const +isPosEqual (refPos:const objPos*) : bool const +getSymbolIfPosEqual(refPos:const objPos*): char const

This allows the main program to consistently use methods to retrieve data instead of needing to use the struct occasionally for accessing coordinates. This organizes the main program as retrieval methods are the only mode of data access being used, keeping the code consistent.