COMPENG 2SH4 Project – Peer Evaluation [30 Marks]

Your Team Members: group (chadwe1 & farshads)

Team Members Evaluated: societytomymanagement (wangj840 & dogara1)

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 **30 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.

Interpreting how objects interact with one another in the main program loop is very simple as the code is polished to have as little code as possible in the main project file. It uses the object's functions to maximize efficiency and use as little code as possible in the main program loop. The commenting they did is also very beneficial since it makes understanding their code much simpler. Additionally, instead of making the board in a 2d array like we did, they simply printed it out directly. Their main program loop prints out the board directly instead of storing it in a 2d array. This does simplify the code as an array is not needed however an array could be useful when building on top of the project. Still, it is an efficient main program.

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.

Object-oriented development makes it easier to build upon and interpret code, especially when its code that you didn't write. Additionally, it helps encourage code organization. Having separate files for different classes makes it very easy to follow along with the function of the code and makes it very easy to debug for errors since you can narrow down the possible classes that are causing the error. When starting a project from scratch, the main con of and OOD implementation is the development time. Had we simply expanded on our PPA3 code, implementing the snake would've been most likely much faster.

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.

The code offers sufficient commenting to help the viewer understand the function and purpose of each line of code. Additionally, as previously mentioned, the code is very efficient and simple, which also helps the viewer gain a better understanding of the code's functionality. Variable names are also very accurate to what the variable is storing, which additionally makes understanding the codes functionality easier.

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 code follows standard indentation practices, making the code very legible and neat. White space is also added where needed to improve the appearance of the code. All print statements are printed with ample space around them – they are not squished together with the other statements. This makes the playing the game very enjoyable since it makes everything look neater.

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)

The snake game offers smooth and a bug-free playing experience, making it an enjoyable to play. There are spaces between each character, technically making the board 60x30, making the game very visually appealing and easy to follow. All the instructions are also clearly listed at the bottom, allowing the player to quickly and easily understand the game's rules.

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.

The snake game does not cause any memory leakage as everything instantiated on the heap was also deleted from the heap.

Project Reflection

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

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

The compound object design of the objPos class and the struct inside it are not sensible, since they're redundant. This is because it is not necessary to instantiate the objPos struct when it is already inside of a class. All the variables are already specific to that one instance of the class, making the struct redundant.

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. <u>You are expected to facilitate the discussion with UML diagram.</u>

To improve the object design, I'd remove the struct and simply implement all the variables in the struct just as data members that are specific to the object class objPos.

New UML diagram: