COMPENG 2SH4 Project – Peer Evaluation [25 Marks]

Your Team Members Cameron Lin Andrew Guo

Team Members Evaluated Veronica Hafsa

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 main logic in the main program loop shows how the three objects of “Player”, “Gamemechs” and “Food” interact with each other. It is easy to tell what each object does due to thorough comments and proper variable and function declarations, making the code easy to read and interpret. Some negative aspects of the code are the way functions and pointers are used to interact with each other, causing long lines of syntax and making it hard to know what certain variables do.

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 of C++:

- The code is modular and reusable and has high abstraction which simplifies a lot of the code writing, whereas C is less versatile and harder to follow when implementing large amounts of functions and logic
- It makes it easier to add new features compared to C

Cons of C++

- The syntax and knowledge required for larger programs becomes more difficult, where C is simpler to understand

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 comments adequately addressing the meaning of specific lines of code.

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.

Yes, the code provides good indentation, white spaces, and new lines between lines of code for better readability.

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 effectively provides smooth bug free playing experience of its intended design. One small error was the loss message not properly appearing. A debugging approach would be to check the program logic and see if the print message would be overridden by something like the clear screen function.

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.

No, the snake game does not cause any memory leak.

Project Reflection

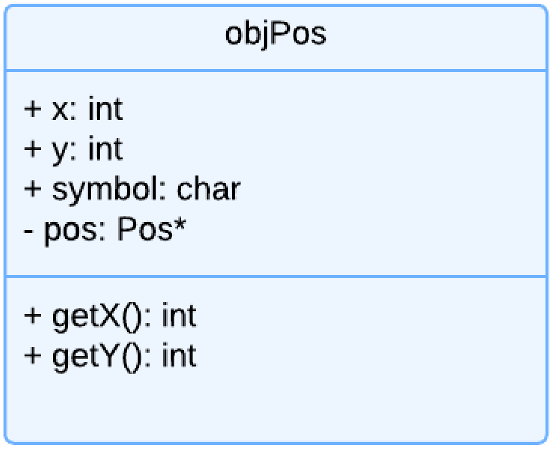
Recall the unusual objPos class design with the additional Pos struct. After reviewing the other team's implementation in addition to your own, reflect on the following questions:

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

The object design is not a sensible decision. The implementation of the design is unnecessary as the code needed for this implementation does not really need to be used to perform the function of the program as other more simple methods are available.

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).

Instead, we could set x, y and storing them as private members in the objPos class. Furthermore, we can access the values of x, y by creating a getter function which returns the value for each respective variable.



Above is a simplified version of the objPos class, which shows the new changes of adding the x and y integers and removing the pos structure.