COMPENG 2SH4 Project – Peer Evaluation [30 Marks]

Your Team Members wonge84 sifarf

Team Members Evaluated erik and owen lopeze5 loho

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.
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.

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.
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.

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)
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.

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?
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.

OOD Quality

1. From looking at the main program code, I can tell that the Food and Player classes interact with the GameMechs class to generate Food positions and to display the game board. It is easy to look at the DrawScreen function and understand how the game board gets each symbol on the game board from the other classes.

2. Pros:

- Reusability of code

- Inheritance

- Can easily tell how objects interact with each other

Cons:

- Takes more time to create new object classes
- Higher learning curve

Code Quality

1. Most of the code is self-documented with good variable names. The code is written in an order that logically makes sense. Parts of the code that are less immediately intuitive are explained well with comments.

2. The code has good formatting overall. Indentation is used properly within functions, for loops, and if statements. White spaces are used in a sensible manner, making the code easier to read quickly. Newline formatting is also good, with for loops and if statements having the code body on separate lines.

Quick Functional Evaluation

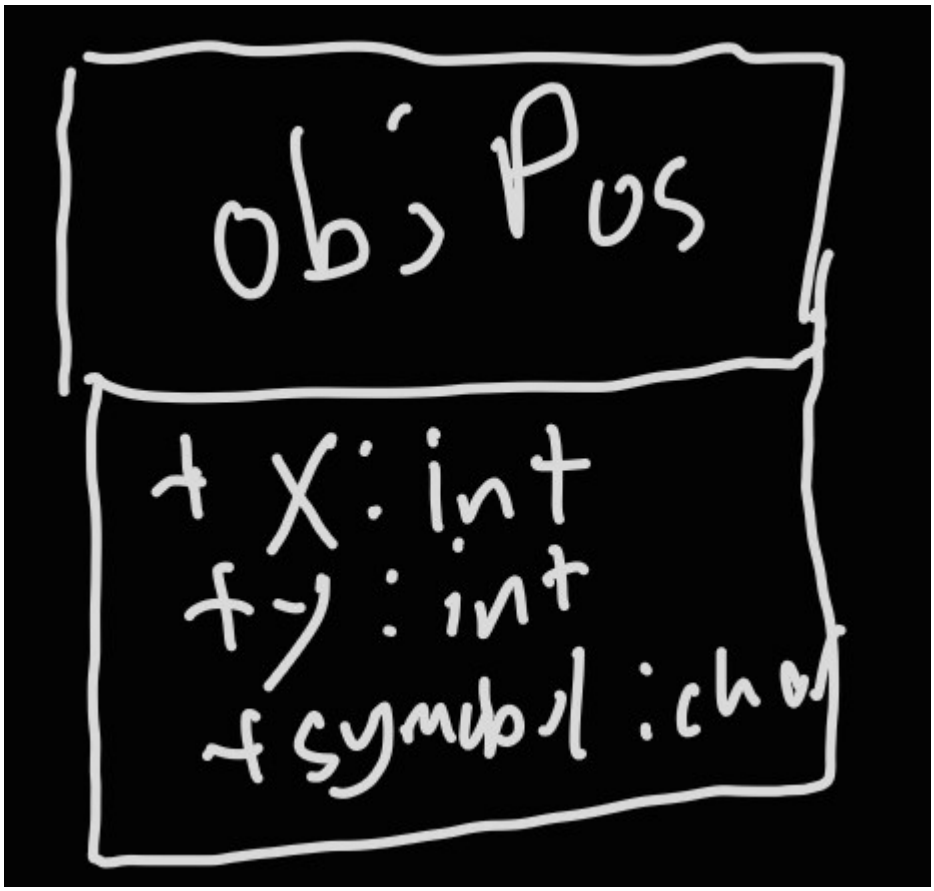
1. The game runs smoothly, with proper wrap-around logic and snake growth upon food consumption. Food is generated after being eaten properly as well. The only issue is the score starts at 1 instead of 0, but this can be easily fixed by subtracting 1 from getScore for display purposes.

2. After checking with DrMemory, the program did not demonstrate any evidence of memory leak.

Project Reflection

1. No, the compound object design is not sensible for objPos. All the struct Pos does is give us the x and y coordinate for the object, but this can be easily implemented as part of the private attribute within the class. With such a simple purpose, it makes more sense to get rid of the struct altogether. objPos is also meant to be the base class with which other classes like objPosArrayList, Food and Player inherit from. Calling Pos->x or Pos->y is also annoying when having to type it out so often in the code.

2. To improve the object design, we believe that simply implementing the x and y coordinates to the ObjPos class would improve the object design. They would go in the private attribute of the class, removing additional repeated syntax in the code, making the code more readable and simpler to both write and review.



(x and y become private attributes, no need for pos struct)