COMPENG 2SH4 Project – Peer Evaluation [25 Marks]

Your Team Members shawb10fall24 pyer4

Team Members Evaluated grenia4 switzmab

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.

Object oriented code is supposed to simplify the logic by keeping it in member functions of a class. However, there are many if-else and switch cases within the RunLogic function, instead of calling objects and member functions that could do these checks. This makes it slightly difficult to understand at first glance and overcomplicates the logic. For example, they check self-collision and set the lose flag within the main logic when this could have just been done all within the checkselfcollison function itself. RunLogic should be more high-level delegating tasks to member functions.

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

- Draw Screen function was easy to follow as the boarders were initialized inside the default and additional constructor in GameMechs function.
- It offers a more organized approach when initially creating code.
- Easier readability through each file as it uses a more logical approach when writing new code. Each class follows the same steps, that being each has constructors, getters, and setters.

Cons

- Difficult to find functions as it runs across multiple files. It requires multiple files to be open and to constantly switch between tabs to create new code and to understand the main file.
- An OOD approach becomes redundant as every class inside each file need to use the rule of 6 and therefore has 6 default functions on top of the getters, setters, and additional constructors need when implementing various parts of the code.

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 some comments in some files but very little in others. For example, GameMechs has very little comments while objPos has many. In the main, there is also a lot of lengthy comments for code segments that are just checking conditions. This could be improved by generally adding more comments, while keeping them concise.

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.

There are appropriately placed indentation and whitespace as well as new line formatting while printing the game screen. One thing I would suggest is to name the instances of the objects more intuitively, like “p” could instead be myPlayer and “board” could be Ga. Also, when the player loses the game by colliding with itself there is a large “GG” that prints within the game board out of X’s. This is a little confusing and would be more legible and readable if this message was placed below or above the game board or after clearing the screen.

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 function incorrectly randomizes food characters, by only printing x’s and o’s. It also only randomizes the x and y positions on the board.

- Code does not follow OOD principals in the RunLogic function. The exit key ‘ ‘ should be implemented as a function inside of GameMechs as well as checking for collision, food eaten and setting the lose flag.

- After the set loose flag is true “GG” prints out of X’s within the game board leading to confusing screen logic as it overwrites the snake and current randomized food positions. Using the debugger to check if there is anything printed inside the boarders is a suggestion on how to debug.

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.

There was no memory leak in their program, and everything was correctly deallocated form the heap.

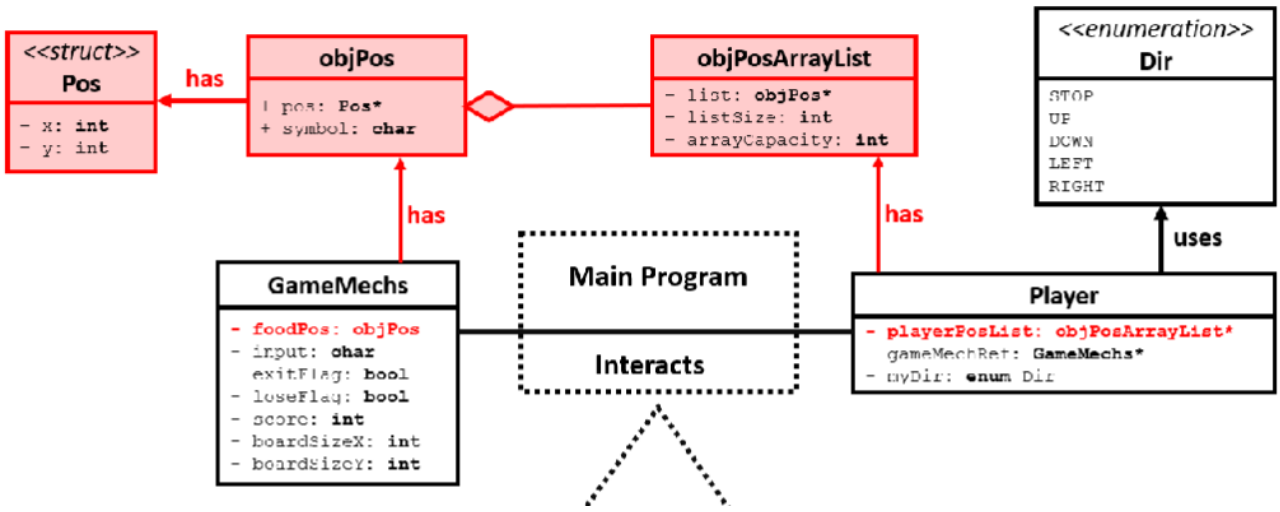
Project Reflection

Recall the unusual objPos class design with the additional Pos struct. After reviewing the other team's implementation in addition to 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?

No, I do not believe that the objPos class is sensible as it creates a confusing a lengthily statement when trying to reference the x,y, and symbol. When referencing any getter inside of objPosArrayList, this contains a list of instances inside of objPos. ObjPos was only used for creating inital points inside of the player class and then used to find the position of the randomized food.

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(s).</u>



The best way to improve this program is to combine both the objPos class and the objPosArrayList class as objPosArrayList creates a pointer of objPos privately. If the two were combined, less referencing to objPos would be needed as objPosArrayList is the primary way of accessing the x,y position as well as the symbol of the snake and food class. The above UML diagram would then change so that objPosArrayList has <<struct>> while GameMechs has objPosArrayList.