COMPENG 2SH4 Project – Peer Evaluation [25 Marks]

Your Team Members	Angelina Yang	Anika Dugh
Team Members Evaluated	Leah Sharma	Wendy Zeng

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 and mail program loop is consistent, and the comments are placed very effectively throughout the main program. The draw function could had been improved by not using that many nested lists and maybe using arrays. The functions look hard to understand and inefficient.
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.
   C++ here is better for larger projects and managing various functions while c is better for small scale projects like in ppa3 .It easier to add more features in c than C++ but in C++ you can be more organized in your approach.

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 provides comments throughout the rest of the project especially in objArrayList file. I think more comments could had been added to GameMechs File. It doesn't clearly convey the code's objective. The errors are handled exceptionally well too. Also, there is not many printf statements for the user while playing the game.

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 spacing and indentation is consistent throughout the code and makes it very easy for the observer to read. I would change deploying new lines after big loops to make it easier to differentiate between various loops and in nested loops.

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)
   No there wasn't any bug in the code and it ran bug free throughout the game. I think maybe using different delay values and seeing which fits the best.
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.
   Throughout the code the heap variables were free with delete [] thus no memory leakage was found.

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 compound object design of objPos class is not sensible. objPos is a simple class the holds x, y, and symbol, having a dynamic memory allocation is over complexing the program. The use of new and delete in the class will lead to future memory management, adding more to the workload. The class can work the same if we just use stack-based member variables.

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 can try to avoid dynamic memory allocation for a simpler design. So, instead of dynamically allocating memory for the Pos struct, just store the position inside the objPos class as member variables. The delete and new is no longer needed, as well as copy/copy assignment constructor.