COMPENG 2SH4 Project – Peer Evaluation [25 Marks]

Your Team Members Aashish – paula41, Garish - manivang

Team Members Evaluated MACID #1: rakuljk, MACID #2: mcmannp

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.
   Their code is neatly organized and well spaced-out which makes it readable and consequently helps the user understand the code well. The comments are also easy to comprehend and are inserted in areas which the user may find hard to follow, making the program seem straightforward. However, there should be some more comments in sections such as the RunLogic function which may need some comments to explain what occurs within it.

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:

   - Keeping variables separate of each class
   - Composition, for example, the food class was made with object array list, this ensured reducing reptation. And allow access to data of other classes to enforce conditions based on the snake body and have food members to be built on objPos to access coordinates.
   - Allowed for collaboration to be easier, working on it virtually, and not needing to wait for the other developers to implement your section of the code.
   - Keeping class files separate from project allows for better organization, which also allows for member function use outside of the class.

   Cons:

   - Difficult to keep track of multiple source and header files and to recall their various functionalities when linking them together
   - Debugging is significantly more complicated throughout the different classes
   - If there are memory leaks, the existence of many variables and members being initialized on the heap makes tracking said leaks an arduous task

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 in all source files. All comments explaining the workings of member functions are descriptive, yet concise. Thus, the comments add to the users' understanding of their code. Minor shortcomings originate from insufficient commenting employed in the numerous for loops which may make it unclear as to what objects or variables are being looped through. There are many classes within the code, and it becomes challenging to remember the object/variable names, so adding comments for what the for loops loop through would make them easier to read.

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.
   This code has well placed indentations and newline formatting ensuring it has excellent readability. Furthermore, there are also adequate spaces for comments which are also well separated from the code body, so that it is not cluttered or disorderly in appearance.

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)
   With thorough testing, there are no bugs to be seen in the code. Wrap around, collision with food and self, along with generation of food all seem to be working correctly and there are no issues when playing. Additionally, an exit code of 0 was returned indicating no segmentation faults.

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.
   After running a diagnostic with Dr. Memory, there are no memory leaks in this code.

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?
   We don't believe that the compound object design is sensible, the reason for this is because a struct itself is a complex data type and a class is also complex data type. With the ObjPos class, it uses members of the struct to record the x and y coordinates, however, it would be more efficient to make the x and y position a public part of the class. This would reduce the members that are the heap and remove the struct overall.

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)
   As explained above, it would be better to put position x and position y as public data members of objPos along with symbol. The UML diagram below demonstrates this: