COMPENG 2SH4 Project – Peer Evaluation [30 Marks]

Your Team Members	Jason Baik	Julia Palumbo
Team Members Evaluated	Sivansh	Lhavanjan

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.

Yes, the main program loop logic is very easily interpretable and well commented. There is a very good use of spacing/indentation to convey exactly what is going on in the program. Everything was very thought out and even the bonus was completed by them. One suggestion is to either make all of the delay constants either included inside of the speed control method or to make them all global constants for clarity, but other than this minor thing, I believe that this code is really well written and thought out.

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.

The pros of this object oriented design approach in C++ is the modularity of it. When everything is a method of a class, it can make it easier to debug the code due to the fact that the programmer can see exactly where the problem is happening by using print statements. Additionally, it is easier to see exactly what is happening to the object due to the syntax of the method. For example, calling 'myGM->getInput()' makes it immediately apparent what is happening to your game object. The cons to this approach is the unnecessary complexity that this brings when the implementation does not require multiple objects of the same class. For example, in the basic snake game implementation, it only requires one object of the same class for all the classes except for' objPos'. Another con of this approach is the time consuming process of having to make constructors, copy constructors, assignment operators, destructors as well as header files and the multiple methods required to access the variables inside of the object for classes of which only one would be constructed.

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 evaluated is very well commented. The comments are clear and concise and make the reader understand each component of the code is for and what it does. The comments allow the reader to understand the purpose of each function used in an efficient way. The overall code is very easy to follow and is very organized. The variable names chosen are descriptive and related to what they are used for. If someone were to look at this code for the first time with no instructions beforehand, the programmer or reader would have a very easy time following along and would easily understand what the code is used for and what is expected when the code runs. Based on the comments provided I do not think there are any shortcomings and there is nothing needed to improve the comments.

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.

The code follows perfect indentation. It is very organized and is made very visually pleasing and makes the code very easy to follow. The newline formatting is done very well in this code. Each block of code that is together is separated from other blocks, allowing the reader to clearly see what each block of code does. There is not an overuse of new lines, making the code the perfect length and there are not too many lines of code. The code has very sensible whitespaces and each line of the code is very easy to read from left to right. I think that with how well organized and easy the code is to follow in terms of formatting there is nothing I would change.

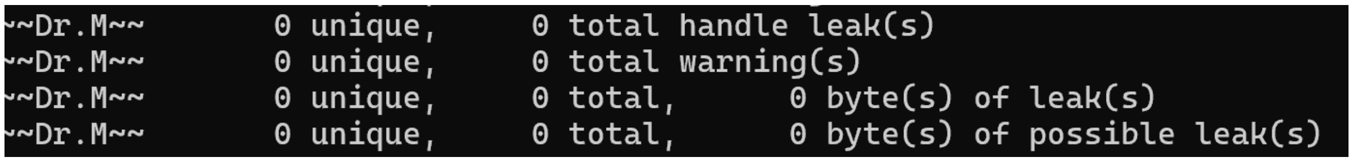
Peer Code Review: Quick Functional Evaluation

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 code provided does provide a smooth and bug-free playing experience. There were no features that seemed buggy. After running the code for a long time, testing the wrap around and analysing the code. The escape key is a $ sign and when clicked the code outputs the correct response. There are no bugs that are shown while the code runs.

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.

The Snake Game memory has no memory leakage.



```
~~Dr.M~~        0 unique,      0 total handle leak(s)
~~Dr.M~~        0 unique,      0 total warning(s)
~~Dr.M~~        0 unique,      0 total,      0 byte(s) of leak(s)
~~Dr.M~~        0 unique,      0 total,      0 byte(s) of possible leak(s)
```

Project Reflection

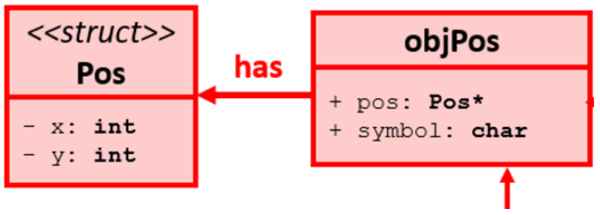
Recall the unusual objPos class design with the additional Pos struct. After reviewing the other team's implementation in addition to yours, reflect on the following questions:

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

The compound object design of objPos class was unnecessary and can be seen as unsensible as the implementation requires an additional step when trying to access the x or y coordinate of the object. For example, when trying to access the x coordinate of the head element, the programmer would have to access it by going into the objPosArrayList and then into the object position class and then the position struct and then they would have access to the x value, 'playerPosList->getHeadElement().pos->x.' This could have simply been reduced to 'playerPosList->getHeadElement().x' if the x and y coordinates were implemented in objPos as x and y instead of in the pos struct. Additionally there are no additional benefits to this implementation in C++, as there is no difference between the x and y coordinates being in the struct in the object as opposed to the coordinates being inside of the object and being accessed and mutated with getter and setter methods.

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.

As previously mentioned, the design implementation could have been simplified from the following:



To this:

objPos
+ symbol: char - x: int - y: int

This would simplify the implementation however some additional getter and setter methods would have to be included. This would include:

Getx()

Setx()

Gety()

Sety()

And this would simplify the implementation and make it more clear to the programmers as it keeps the format of the project consistent. Note that the x and y positions are still declared privately to prevent non methods from accessing the coordinates (accidentally changing it, etc).