COMPENG 2SH4 Project – Peer Evaluation

Your Team Members: Carson Crysler, Luca namestnik

Team Members Evaluated: Loretta Lau, Ashviya Jeyaseelan

Provide your genuine and engineeringly verifiable feedback. Ungrounded claims will lead to deductions.

Part I: OOD Quality

1. **[6 marks]** OOD is about sensible code modularization. Looking at the header files of each object, can you easily interpret the possible behaviours of the objects involved in the program, and how they would interact with each other in the program? Comment on what you have observed, both positive and negative features.

    - All the header files are neat and organized keeping the same spacing and consistent order of member functions (constructors/destructors then setters/getters). One thing I would recommend to do is stay consistent with declaring the private members before the public members for good coding practice and consistency. In all the header files, the private is declared first except in the player file which the public members are first. Also, no header files have unnecessary members or functions that are not used throughout the code. All the functions and variables have appropriate names that describe what they do/hold. This allows someone reading the code to understand what the function or variables do/hold with out looking through the whole code. There was clear use of objects used in other classes that have relevant names to assist what they do within that class.

2. **[6 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 in the program is very neat and easy to read. All the variables used in the main have names that are describing what they do. You can easily tell how the objects interact with each other and what variables are pointing to/ holding. They used the classes and objects well to minimize the code in the main and just call member functions to execute the appropriate tasks. When looking at the code I noticed they included an object; objpos tempPos in the global scope at the top of the file. tempPos was declared but never used in the whole file, this creates more storage, another task for the complier and bad coding practice. This variable can be deleted from the code as It is not used and unnecessary. I also noticed that food is declared globally which is only used in the drawscreen function. This variable can be declared locally in the drawscreen function prevent bad coding practice and unnecessary use of storage through the code.

3. **[5 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.

- Can classify different members in different scopes with classes whereas you cant make something private in a struct
- Can function overload in c++ when using classes but cant in c when using structs
- Can easily inherit or in our case use composition in c++ classes but not in structs in c
- When using c we executed a lot more in the main file because it takes a lot longer to create all the functions with structs and easier to make a mistake than using classes in c++

Part II: Code Quality

1. **[5 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.

   - Throughout the code there Is several comments outlining what each function does. These comments are all correct and makes it very easy to understand for someone who is trying to interpret the code for the first time without knowing what it is going on. One thing I would add is a few comments at the top explaining what each global variable is storing or what it points to. I liked how they used define at the top of the code to define escape key to 32 which is the ASCII value. This made it easier to understand instead of using a value of 32 in the code.

2. **[4 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 very good indentation and spacing. The indentation and spacing is consistent throughout the entire code keeping It the same within each header file and source file. Nothing looks cluttered and all the brackets are wrapping around each code block correctly along with the code breaking at necessary lines. This code overall is very clean looking and readable.

Part III: Quick Functional Evaluation

1. **[8 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 a technical user feedback)

   - After playing the game a few times, the code runs great with no bugs. They implemented the wrap around and snake movement logic correctly along with the randomized food and the collision detection. All the food is spawning in appropriate locations and when the snake eats the food it grows, simultaneously increasing the score. This makes their game run smoothly and have no errors when running into yourself or exiting the game.

2. **[6 marks]** Does the Snake Game cause memory leak? If yes, provide a digest of the memory profiling report and identify the possible root cause of the memory leakage.

   - After running drmemory for the project executable file there was no leakage found in the code. They did a good job deleting the memory that they allocated on the heap for the pointers in main and creating the right destructors in the classes.

Part IV: Your Own Collaboration Experience (Ungraded)

1. Tell us about your experience in your first collaborated software development through this project – what was working and what wasn't. If you are a one-person team, tell us what you think may work better if you had a second collaborator working with you.

   - This was a very good experience, and the collaboration went well. We split the work up equally and helped each other when one of us was stuck. Having two people to trouble shoot and approach the code where on of us was stuck helped a lot. This provided another perspective on some way to approach a task and if one way wasn't working, we talked it out and tried again. The team project was a lot more realistic, as it prepares us to be working as a team in the real world. Overall, this was a great experience/project and we both enjoyed it.