COMPENG 2SH4 Project – Peer Evaluation

Your Team Members Danyal Fairoz, Hamza Nadeem

Team Members Evaluated Ali, Maaz

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.

   Looking at the header files of their program, I easily see how the function works and what it is meant to do, this is because their functions names (which were given) are easy for the user to interpret. This, however, could've been further supported if they had added comments that helped explain the functions a little more in depth. For example, in their "Food.h" file, they have 2 integers that they made called X_range and Y_range, however, the user may not get an idea of what the ranges are exactly for, because it is not commented. I, as someone who also coded this game, can infer that the variables are used for the bounds of generation for the food, but there is no way for me to be sure unless I were to look at their Food.cpp file. Apart from commenting, it is clear what the functions in the header file are made to do.

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.

   Their code is extremely well modularized and easy to interpret in the main loop. I can easily tell what function does what and from where the function is from. It is easy to see how the objects interact with each other because of the names that were given to each function and class pointer. There are no concerns that jump out at me either. They did well.

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.

   Pros of C++ OOD:

   - Makes code easier to understand.
   - Helps the user keep their main file clean and neat.
   - Bugs are a little easier to locate
   - Greater readability due to the numerous classes and member function names

   Cons of C++ OOD:

- Slightly difficult to start off if you have no experience.
- Slightly more tedious to code because of the amount of files
- Greater overall length of code (many more lines)

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.
   The code offers commenting for nearly every step of the process. This manifests in explaining the variables, the process of how a code block works, showing values of constants, etc. The objPosArrayList and Food files were noteworthy in this aspect, with detailed explanations for almost every line.

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 was neatly organized throughout. Generally, a good amount of space was implemented for ease of readability, and the comments were placed in an unobtrusive manner. However, there were a few instances, such as in the movePlayer() function, where the code was slightly hard to read due to the lack of spaces for long stretches of code. In addition, there was only one line of text displayed in the terminal, so we can't comment on the newline formatting.

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)

   We did not notice any bugs in their game, it ran smoothly and worked as intended. There were no visual/gameplay glitches.

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.

   The snake game does not cause any memory leakage.

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.

   Danyal:
   I think this collaborative experience went very well overall!
   - Communication was the most important factor for this project, especially because we haven't had much experience programming in groups. It was imperative to convey the thought process behind your work so your teammate can have a good understanding of how to implement it. Commenting on your code can help in this aspect, but talking one on one was the most efficient method.
   - I also think it's important to both ask for and offer your help. This is especially true in a group setting, as creating the best final product is beneficial for both team members.
   - Hamza was a great partner in all regards, offering help when I needed it and always stepping up to the plate to tackle any issues/setbacks.

   Hamza

   Although this isn't my first collaboration on a software development team for a game, I can say that my thoughts on this matter have remained the same.

   - When working within a group, especially for something like coding, it is important that every group member comments on their respective parts. I can't stress this enough because everyone has different levels of experience and coding style. Leaving without commenting, or without properly conveying to your team what your code does, can instantly.
   - Danyal also did a great job communicating any issues he came across so I could help solve them immediately, this help tremendously in terms of the time it took to complete the project overall.