COMPENG 2SH4 Project – Peer Evaluation

Your Team Members Aashvi Patel Jhanvi Shah

Team Members Evaluated trepanv _____

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.**

   Positive aspects evident in the team's codebase include the use of clear and indicative class names. This clarity facilitates a straightforward understanding of each variable's meaning, making it easy to navigate the code and comprehend the interactions between different components. Additionally, each class is designed with a singular and well-defined purpose, ensuring that its responsibility is focused on executing a specific piece of code. This deliberate structuring enhances the code's readability and comprehensibility, allowing for seamless tracking of the program's flow. Some cons about the code are making a couple variables (like boardY and boardX) global and constant to better the readability of the code, but this was because the template of the code was given to us like this and we cannot say it was the p.

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 clarity of the main program loop's logic is facilitated by the team's commendable practice of providing thorough comments and using descriptive variable names. This thoughtful approach ensures that the code is easily comprehensible, allowing for a seamless understanding of its intended execution. All parts of runLogic and DrawScreen are easily readable and well-spaced out. Therefore, we do not see any negative features for this part of 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.

C++ OOD

Pros

- indicate all class names and responsibilities, increase code readability
- clearly defined classes improve understanding and comprehension of code

- object methods are utilized effectively

Cons

- space for memory leakage, which requires careful memory management
- tight object coupling between objects can reduce flexibility in making changes to future code changes

C Procedural Design Approach (PPA3):

Pros

- simple procedural design and more straightforward in its approach
- more efficient in terms of resource utilization

Cons

- lack of modularization can potentially make it harder to maintain the code overtime
- not complex, so less suitable for larger projects
- the procedural approach may make it harder to reuse code

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 well-placed header comments explaining each segment of the code very briefly. However, a little bit more description in the comments would've helped to follow the code somewhat better. The variable and method names are clear and concise which reflect the purpose of the

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 naming of the variables is very good. It is clear what the variables store and where it was used. There are comments explaining either the method or line of code. There could have been a couple more comments in the individual files and more in project.cpp. In the code, there was a lot of whitespaces making the handling a little difficult as we had to scroll down and up a lot to look over the code.

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)

   The code does run seamless on one team members laptop as they have a windows laptop, and the other team member is not able to run it as there is a compilation error because of a Mac OS. To better the code the programmers should keep in mind to use certain convention. The Mac user must change how an objPos was assigned the x, y,and symbol values using the dot operator instead of the curly brackets. Other than that, the game ran seamlessly. The only thing we would have hoped for a better end message rather than just printing out the score at the end of 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.

There are no memory leaks as all the variables on the heap are deallocated for in the files.

```
(base) jhanvi@Jhanvis-MacBook-Pro 2sh4-project-trepanv % leaks --atExit --
  ./Project
Project(79580) MallocStackLogging: could not tag MSL-related memory as no_
footprint, so those pages will be included in process footprint - (null)
Project(79580) MallocStackLogging: recording malloc and VM allocation stac
ks using lite mode
Process 79580 is not debuggable. Due to security restrictions, leaks can o
nly show or save contents of readonly memory of restricted processes.

Process:         Project [79580]
Path:            /Users/USER/Downloads/*/Project
Load Address:    0x100150000
Identifier:      Project
Version:         0
Code Type:       ARM64
Platform:        macOS
Parent Process:  leaks [79579]

Date/Time:       2023-12-06 10:51:00.750 -0500
Launch Time:     2023-12-06 10:49:31.462 -0500
OS Version:      macOS 13.5.2 (22G91)
Report Version:  7
Analysis Tool:   /usr/bin/leaks

Physical footprint:         3345K
Physical footprint (peak):  3345K
Idle exit:                  untracked
----

leaks Report Version: 4.0, multi-line stacks
Process 79580: 335 nodes malloced for 201 KB
Process 79580: 0 leaks for 0 total leaked bytes.
```

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.