COMPENG 2SH4 Project – Peer Evaluation

Your Team Members __Zhenghan Li__ ____Yifan Zhang_____

Team Members Evaluated _____Leopold______ _____Jake__________

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

Part I: OOD Quality

1. **[5 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.

   We can easily interpret the possible behaviors of the objects and understand the interaction with each other by reading their object header files. Their header files and corresponding .cpp files are professionally written and quite easy to follow. We believe that other users can integrate the objects written by this team and use for their projects with ease. The only concern is caused by lacking the dedicated object for SnakeFood, since food has a different behavior than GameMechs object and it can be derived into diverse types of foods. Snake Traps and Awards in our program are derived from SnakeFood object. Therefore, we believe that from maintainability and expandability point of view, it is necessary to create a SnakeFood object instead of combining that with the GameMechs object.

2. **[5 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 main program Project.cpp are simple and well structured. All function and necessary features can be identified with ease. They initialize including gameMechs and player objects inside the function called void Initialize() once the game is started. Other features like collision detection, play behaviors are inserted into void drawScreen(), which ensures that when the player's behavior is updated, it can be draw as quickly as possible. There are no negative features because their main function is high quality and easy to understand.

3. **[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.

   C++ OOD Approach Pros:

   - Enhanced code reused ability.
   - Code modular design and make the code easier to maintain and modify

- Scalability, it is better for large-scale applications due to its ability to manage complex data and functionality.

C++ OOD Approach Cons:

- Complexity code might need even for a simple task, will also bring a harder learning curve.
- The code could be inefficient.

C Approach Pros:

- Straightforward and easy to understand, better for smaller projects or specific tasks.
- Could be more efficient compare to OOD approach since it can offers more control over the computer's hardware resources.
- Easier for new beginner to learn.

C Approach Cons:

- Low code reused ability compared to OOD.
- Scalability, it is not efficient and modular enough for large-scale applications.

Part II: Code Quality

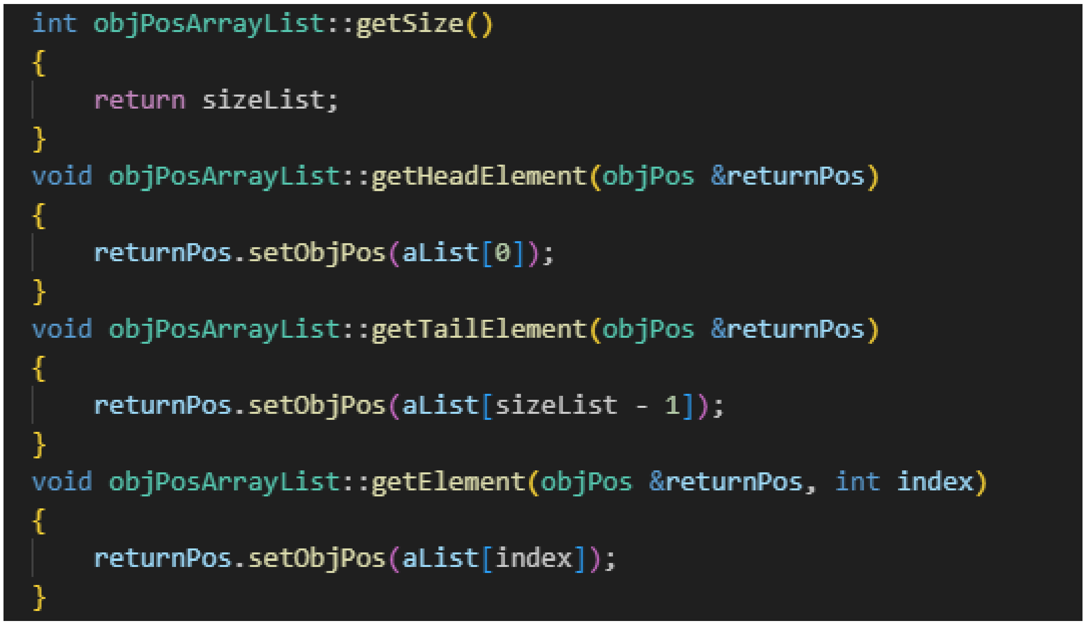
1. **[4 marks]** Does the code offer sufficient comments, or deploys sufficient self-documenting coding style, to help you understand the code functionality more efficiently? If any shortcoming is observed, discuss how you would improve it.

- The code has basic some commenting and self-explain method names which is good, also doing well in the comments for the code inside the function, but the comments lack detailed explanations of methods and data members, and the complex logic within methods is not well commented, making it a little bit difficult to understand.
- Also, a brief comment should be included at the beginning parts of each function.

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 overall readability if the code is good. The code in their program is well spaced with reasonable indentation and white spaces in between. Newline formatting is effectively use to print game information. But the format is not perfect. In .cpp files for object function implementations one or two rows of white space can be added between functions to further improve the overall readability.

The screenshot below is one of the example from their objPosArrayList.cpp, readability can be improved significantly by adding a row of white space between each functions.



```
int objPosArrayList::getSize()
{
    return sizeList;
}
void objPosArrayList::getHeadElement(objPos &returnPos)
{
    returnPos.setObjPos(aList[0]);
}
void objPosArrayList::getTailElement(objPos &returnPos)
{
    returnPos.setObjPos(aList[sizeList - 1]);
}
void objPosArrayList::getElement(objPos &returnPos, int index)
{
    returnPos.setObjPos(aList[index]);
}
```

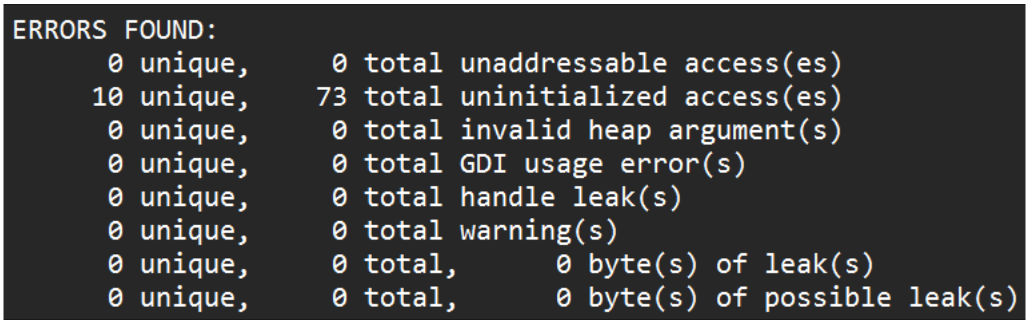
Part III: Quick Functional Evaluation

1. **[6 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)

   This program presents a bug-free experience and all basic functions like snake growth, collision detection between food and player itself are implemented in a proper way.

2. **[4 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.

   Their Snake Game has no memory leak. They did a very good job with memory management.



```
ERRORS FOUND:
      0 unique,     0 total unaddressable access(es)
     10 unique,    73 total uninitialized access(es)
      0 unique,     0 total invalid heap argument(s)
      0 unique,     0 total GDI usage error(s)
      0 unique,     0 total handle leak(s)
      0 unique,     0 total warning(s)
      0 unique,     0 total,      0 byte(s) of leak(s)
      0 unique,     0 total,      0 byte(s) of possible leak(s)
```

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.