COMPENG 2SH4 Project – Peer Evaluation [25 Marks]

Your Team Members siddir15 | mazham8

Team Members Evaluated bohme | mcmilj10

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 **25 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.**
   Contributed: siddir15
   After going through .h and .cpp associated files, I can confidently say that logic was interpreted, minimum of four implemented despite not being utilized further reenforcing standard and interpretability.

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.**
   Contributed: siddir15
   I would for C++ OOD approach the pros are as discussed in lecture encapsulation, modularity and more importantly modularity. Where all together creates a sort of library esque functionality. I would say cons are sometimes you may not need to necessarily implement a certain ability but must be due to minimum of four (actually six) so it could increase complexity.
   For C procedural I would say the pro is that it is quite linear in fashion and access of elements/memory. Which can have its pros and cons however requires more attention for memory activities. A con could be found from the contrast to C++ OOD where, there is less reusability so application in larger scope could potentially yield to inefficient development.

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.**
   Contributed: siddir15

   Throughout the whole .h files didn't have much comments, however investigating in the .cpp yielded sufficient comments that produced easy of understanding and readability.
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.**
   Contributed: siddir15

Yes good indentation was followed, everything is spaced nicely didn't feel like there was a part that I struggled to read.

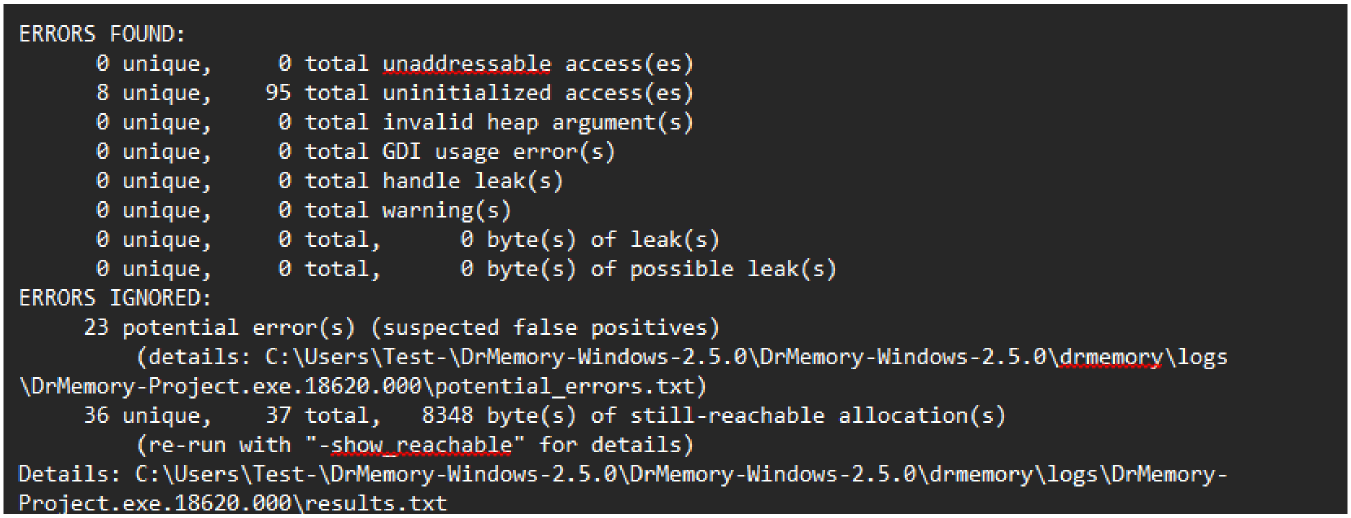
Peer Code Review: Quick Functional Evaluation

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)**

   Contributed: siddir15

   After multiple iteration I would say gameplay is concise and up to standard, the only complaint is that if you were to Exit out of game early it says you won.
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.**
   Contributed: siddir15

   Zero leak was found in the drmemory analysis:



```
ERRORS FOUND:
      0 unique,     0 total unaddressable access(es)
      8 unique,    95 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)
ERRORS IGNORED:
     23 potential error(s) (suspected false positives)
         (details: C:\Users\Test-\DrMemory-Windows-2.5.0\DrMemory-Windows-2.5.0\drmemory\logs
\DrMemory-Project.exe.18620.000\potential_errors.txt)
     36 unique,    37 total,   8348 byte(s) of still-reachable allocation(s)
         (re-run with "-show_reachable" for details)
Details: C:\Users\Test-\DrMemory-Windows-2.5.0\DrMemory-Windows-2.5.0\drmemory\logs\DrMemory-
Project.exe.18620.000\results.txt
```

Project Reflection

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

1. **[3 marks] Do you think the compound object design of objPos class is sensible? Why or why not?**
   Contributed: siddir15

   No, I think that it is sensible because instances that are primarily utilized based on object position should be grouped together so compounding object design based on relative function in my opinion is efficient for my limited understanding.
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. <u>You are expected to facilitate the discussion with UML diagram(s).</u>**
   Contributed: siddir15

N/A