I2P(II) Mini Project 2 Questions

Here are some questions to help you verify your understanding about the source code and the concept of OOP.

If the explanation of your implementation is ambiguous or not clear enough during the demo, TA might ask you the questions below to validate your understanding about this project. Moreover, if you cannot answer the questions correctly, the scores are graded subjectively by the TA in terms of your performance.

Easy Questions

1. Explain the concept of "Encapsulation". Show a short code snippet to illustrate why "Encapsulation" is useful and what might happen if we do not use it.
2. Explain the concept of "Inheritance". Show a short code snippet to illustrate why "Inheritance" is useful and what might happen if we do not use it.
3. Explain the concept of "Polymorphism". Show a short code snippet to illustrate why "Polymorphism" is useful, and what might happen if we do not use it.
4. What is the difference between Procedural Programming and Object-Oriented Programming? Name at least three advantages of OOP.
5. The game crashes when changing to win scene. What tools and techniques did you try to find the bug? Why does the bug crash the game? What have you learned through this debugging experience?
6. How does the game parse the file containing the enemy waves? What is the meaning of the three values in each line?
7. When will the function main return the exit value zero?

ADVANCE QUESTIONS

1. What did the template do to achieve different speed multipliers? When the speed multiplier is zero, can the player place new turrets? Why or why not?

2. In `PlayScene`, multiple `Engine::Groups` are used for storing different objects. What will happen if we remove the groups and add all new objects directly into the scene?

3. In `Engine::Group`, there are two linked-lists, storing objects and controls respectively. Can we simplify this into a single linked-list? What are the advantages and disadvantages of storing them in different linked-lists?

4. Try-Catch statements can intercept the error thrown by the inner code. Why are Try-Catches preferred instead of error codes in modern programming? List out two advantages of the Try-Catch statement.

5. Garbage Collection is a clever concept introduced in Lisp and become widely popular in higher-level programming languages such as Java, C#, Python. What does the code in `Engine::Resources` do? What are the benefits and drawbacks of automatically managing the resources?

6. What is Memory Leak? Can the invention of Smart Pointers reduce this problem? How are Smart Pointers implemented and what are the benefits of using them? Elaborate on the difference between `std::unique_ptr`, `std::shared_ptr`, and `std::weak_ptr`.

7. The public change scene function of `Engine::GameEngine` does not change the scene immediately but waits until the next update. What might happen if the scene is changed immediately when the public function is called?

8. `Engine::Group` is a class derived from `Engine::IObjects`, while storing a linked list of pointers to many `Engine::IObjects`. Why do we store the pointer to `Engine::IObjects` instead of directly storing the value of `Engine::IObjects`?