21BCS052
9/5/23

OBJECTS, ATTRIBUTES AND BEHAVIOUR

Objectives: The objective of this lab assignment was to explore the concept of operator overloading in C++ and apply it to a class representing 2D points. This involved overloading arithmetic, relational, and assignment operators, as well as implementing user-defined functionality through operator overloading.

Learnings:

- **Arithmetic Operator Overloading:**
    - Identified and implemented the overloading of the + operator in the **Point** class to perform addition of two points.
    - Learned about the return type (**Point**) and argument type (**const Point &**) for the overloaded + operator.
    - Explored the usage of the **const** keyword for operator overloading, understanding its role in restricting changes to operands.
- **Relational and Assignment Operator Overloading:**
    - Modified the **Point** class to overload the <, >, and == operators based on the Euclidean norm distance measure between points.
    - Implemented functionality that compares points based on their Euclidean distances.
- **User-Defined Functionality through Operator Overloading:**
    - Investigated linking the "TinyPNG" library into the C++ program and utilizing it for operator overloading.
    - Loaded an image representing nightlight intensity and implemented a user-defined < operator based on the intensity at different points.

Challenges:

- Understanding the intricacies of operator overloading, especially when dealing with different types of operators and their functionality, presented initial challenges.
- Linking and integrating the external "TinyPNG" library into the program required familiarity with external dependencies and their usage.

Key Notes:

- Operator overloading provides a means to define custom functionality for operators like arithmetic, relational, and assignment, tailored to specific classes.
- The **const** keyword in operator overloading ensures that the operands are not modified within the operation.