Mini Project - Generic Numeric Data Type in C++

Project Overview:

To create a base class Numeric that can represent all numeric data types (e.g., int, short, long, double, float, std::complex, char, and unsigned types) and design its derived classes to enable operations like storing these types in a std::vector<Numeric>.

Key Requirements:

1. **Base Class (Numeric):**
    - Create an abstract base class Numeric to serve as the parent of all numeric types.
    - Include pure virtual functions for basic arithmetic operations (+, -, *, /) and comparison operators (<, >, ==).
2. **Derived Classes:**
    - Implement derived classes for int, double, float, std::complex<double>, char, and other numeric types as required.
    - Override the functions for arithmetic and comparison.
3. **Polymorphism:**
    - Use pointers or references to Numeric for runtime polymorphism.
4. **Memory Management:**
    - Ensure proper handling of dynamic memory (e.g., using smart pointers).
    - Avoid memory leaks in the implementation.
5. **Additional Features:**
    - Implement a method to convert Numeric objects to strings (e.g., std::string toString()).
    - Provide functionality to read and write numeric values through std::istream and std::ostream.
6. **Main Function:**
    - Demonstrate the usage of std::vector<Numeric> by:
        - Pushing different numeric types into the vector.
        - Performing operations (e.g., addition, sorting based on comparison).

Constraints:

- Use only features compatible with the course syllabus.
- Avoid templates for the Numeric base class but allow templates in derived classes if required.

Thank You
Edges For Training Team