COMPLEX NUMBER COMPUTATIONAL TOOLKIT: INTEGRATION AND VISUALIZATION WITH PYTHON

Objective:

Create a Python program that functions as a comprehensive complex number calculator but also graph these numbers in their respective forms. This program should accept input in various complex number formats (rectangular, polar, exponential) and perform a wide range of operations and conversions on these numbers.

Requirements:

1. Input Handling: The program must accept complex numbers in rectangular ($a + bj$), polar ($r\angle\theta$), and exponential $re^{j\theta}$ forms. Ensure proper parsing and error handling for each format.

2. Graphing Capability: Implement a feature to graph complex numbers on a 2D plane. You may use libraries such as Matplotlib for plotting. The graph should clearly distinguish between different complex numbers, indicating their forms and results of operations.

3. Operations: Implement the following operations for complex numbers:

    *   Arithmetic operations: addition, subtraction, multiplication, division

    *   Power and root operations: nth power, nth root, complex raised to complex

    *   Logarithmic functions: natural logarithm

    *   Trigonometric functions: sine, cosine, tangent

    *   Inverse trigonometric functions: arcsine, arccosine, arctangent

4. Conversions: Allow conversion between all three forms of complex numbers (rectangular to polar/exponential, polar to rectangular/exponential, exponential to rectangular/polar).

5. User Interface: Develop a user-friendly console-based interface where users can select operations, input values, and view results. Include clear instructions and error messages for invalid inputs.

6. Documentation: Provide comments and documentation within the code to explain the functionality and usage of functions and modules.

Submission Guidelines:

- Submit a single .py file containing your program.

- Ensure your code is well-commented and follows the Python PEP 8 style guide.

- Include at the top of your file a brief comment explaining the program's purpose, your name, and submission date.

Evaluation Criteria:

- Correctness and efficiency of implemented operations and conversions.

- Code readability and adherence to coding standards.

- Robustness in handling various input formats and invalid inputs

- Completeness and usability of the user interface