BMI Calculator Console App

Classes:

1. **BMI Calculator (`BMICalculator`)**:
    - Responsible for calculating BMI based on height and weight inputs.
    - Utilizes generics to handle different numeric types for height and weight.
    - Uses a stack to store previous BMI calculations. (bonus)

Input and Output:

- **Input**:
    - Height (in meters or feet/inches)
    - Weight (in kilograms or pounds)
- **Output**:
    - BMI value (numeric)
    - BMI category (e.g., Underweight, Normal weight, Overweight, Obesity)
    - Display previous BMI calculations using a stack.

Example Workflow:

1. **User Interaction**:
    - Console prompts for height and weight inputs.
    - User enters height and weight values.
2. **BMI Calculation**:
    - The `BMICalculator` class calculates BMI based on the formula:

$$\boldsymbol{BMI = \frac{Weight}{Height^2}}$$

    - Adjust calculations for different units (metric or imperial).
3. **Output Display**:
    - Display calculated BMI.
    - Display BMI category based on predefined ranges (e.g., underweight, normal weight, etc.).
    - Store the calculated BMI in a stack.
    - Provide an option to view previous BMI calculations stored in the stack.

BMI Categories and Ranges

- **Underweight**: BMI less than 18.5
- **Normal weight**: BMI from 18.5 to 24.9
- **Overweight**: BMI from 25 to 29.9
- **Obesity**: BMI 30 or greater

GenericArithmeticCalculator

Create a generic class `Calculator<T>` that supports basic arithmetic operations (`Add`, `Subtract`, `Multiply`, `Divide`) using delegates. Implement the class to work with numeric types (`int`, `double`, etc.).

Requirements:

1. **Generic Class `Calculator<T>`:**
   - Use a generic type parameter `T` constrained to numeric types (`where T : struct, IComparable, IFormattable, IConvertible, IComparable<T>, IEquatable<T>`).
   - Include delegate types for operations (`Func<T, T, T>` for addition, subtraction, multiplication, and division).
2. **Methods:**
   - Implement methods `Add`, `Subtract`, `Multiply`, and `Divide` using delegates.
   - Each method should take two parameters of type `T` and return the result of the respective operation.