EEL3834 - Programming for Electrical Engineers
Fall 2025
Programming Assignment 7: Shape Catalog
Due: 11/21/2025 @ 11:59PM
To be done individually

Objective:
This assignment introduces object-oriented hierarchies, inheritance, and dynamic polymorphism in both C++ and Python.

Problem Description:
You will build a Shape Catalog: a menu-driven program that stores geometric shapes and performs polymorphic operations on them. The emphasis is on class design and virtual dispatch. Both your C++ and Python versions should meet the same behavioral requirements.

Menu:
Below is the menu you will implement for this assignment:
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit

Validation Expectations

All numeric input must be validated before use.

Your program must re-prompt for invalid entries instead of crashing or producing undefined behavior.

1. Menu Selection
    a. Must be an integer
    b. If the input is not between 1-5, print: Invalid choice.
    c. Then redisplay the main menu
2. Shape Selection
    a. When adding a shape, prompt for the shape type: Enter shape type (1=Circle, 2=Rectangle, 3=Triangle):
    b. If a input that is not 1,2, or 3 is inputted print: Invalid choice.
    c. Then reprompt until valid
3. Dimensions
    a. All dimensions (radius, width, height, base) must be positive.

    b. If a non-numeric or non-positive value is entered: Error: Dimension must be greater than 0.
    c. Reprompt for dimensions
4. Scale factor
    a. Must be numeric and greater than zero.
    b. If invalid print: Error: Scale factor must be greater than 0.
    c. Reprompt
5. General rule
    a. Program should not crash from invalid input
    b. If there are no shapes stored, List All Shapes should print No shapes. and then a blank line.

Code Requirements

C++

Files: (.h/.cpp means both a header and a cpp file is expected

- Shape.h (abstract base class)
- Circle.h/.cpp, Rectangle.h/.cpp, Triangle.h/.cpp (derived classes)
- ShapeManager.h/.cpp (manages all shapes)
- main.cpp (menu and user interaction)

Clase Shape (in Shape.h):

- Shape is an abstract base class that represents “any geometric shape” in the program.
- It defines the common interface that all shapes must support (like the ability to compute area, perimeter, scale themselves, and describe themselves as text).
- It is not instantiated directly; only concrete subclasses like Circle, Rectangle, and Triangle are created.
- It exists so the rest of the program (like ShapeManager and main) can work with pointers/references to Shape and use polymorphism to call the correct behavior for each specific shape.
- It ensures that every derived shape agrees to implement the same core operations, so they can all be stored and used uniformly in one collection.
- Because Shape is an abstract base class with only pure virtual functions, no Shape.cpp file is required.

Derived Classes

- Each shape must:
- Privately store its dimensions (like radius, width/height, base/height).
- Override all virtual methods from Shape.
- Provide a constructor that validates dimension values.
- Implement describe() returning a formatted string
    - Example: Circle: radius=3.0 area=28.27 perimeter=18.85

ShapeManager

- Manages a collection of shapes via base-class pointers.
- Required methods:
    - addShape(Shape* s)
    - viewAll()
    - scaleAll(double factor)
    - describeAll()
    - Destructor that deletes all stored shapes.
- All shape operations must occur through virtual functions called via base-class pointers or references.

main.cpp

- Implements the main user interface:
  Shape Catalog
  1. Add Shape
  2. List All Shapes
  3. Scale All Shapes
  4. Describe All
  5. Exit

  Enter choice:
- All numeric and text prompts must match this format exactly.
- Use clean, consistent output formatting for all values

Python

Create a python file named: shape_catalog.py

Requirements

- Implement classes Circle, Rectangle, and Triangle, each with:
    - area(self)
    - perimeter(self)
    - scale(self, factor)
    - describe(self)

- Implement a ShapeManager class that contains a list of shapes and provides:
    - add_shape(obj)
    - list_all()
    - scale_all(factor)
    - describe_all()

- Rely on duck typing. The manager must not check class names or use isinstance().
- Any object that defines the required methods should work.

- All Python classes and the ShapeManager should be defined within a single file named shape_catalog.py.

Sample runs ***(please note you want your assignment to have the exact wording and layout or you will lose points)***

Run 1: (red is user input, blue is where you should have a new line (an empty line))

Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 2
No shapes.
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 9
Invalid choice.
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 1
Enter shape type (1=Circle, 2=Rectangle, 3=Triangle): 4
Invalid choice.
Enter shape type (1=Circle, 2=Rectangle, 3=Triangle): 1
Enter radius: -5
Error: Dimension must be greater than 0.
Enter radius: abc
Invalid input.

Enter radius: 3.5
Shape added.
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 1
Enter shape type (1=Circle, 2=Rectangle, 3=Triangle): 2
Enter width: 0
Error: Dimension must be greater than 0.
Enter width: 4
Enter height: xyz
Invalid input.
Enter height: -1
Error: Dimension must be greater than 0.
Enter height: 2
Shape added.
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 1
Enter shape type (1=Circle, 2=Rectangle, 3=Triangle): 3
Enter base: 5
Enter height: 6
Shape added.
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All

5. Exit
Enter choice: 2
Circle: radius=3.50 area=38.48 perimeter=21.99
Rectangle: width=4.00 height=2.00 area=8.00 perimeter=12.00
Triangle: base=5.00 height=6.00 area=15.00 perimeter=18.81
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 3
Enter scale factor: hello
Invalid input.
Enter scale factor: -2
Error: Scale factor must be greater than 0.
Enter scale factor: 2
All shapes scaled.
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 2
Circle: radius=7.00 area=153.94 perimeter=43.98
Rectangle: width=8.00 height=4.00 area=32.00 perimeter=24.00
Triangle: base=10.00 height=12.00 area=60.00 perimeter=37.62
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 4

Circle: radius=7.00 area=153.94 perimeter=43.98
Rectangle: width=8.00 height=4.00 area=32.00 perimeter=24.00
Triangle: base=10.00 height=12.00 area=60.00 perimeter=37.62
(new line here)
Shape Catalog
1. Add Shape
2. List All Shapes
3. Scale All Shapes
4. Describe All
5. Exit
Enter choice: 5
Goodbye!

Expectations

Your grade will be subject to the following condition(s):

- Submission:
  The submission deadline is **11:59PM** on **11/21/25**. You will be penalized in increments of 25% per day late (regardless of the time).
    - For this assignment submit all your code files in one zipped file named **firstname_lastname_assignment7.zip**
    - Your .zip file should contain the following files:
        - shape_catalog.py
        - Shape.h
        - Circle.h
        - Circle.cpp
        - Rectangle.h
        - Rectangle.cpp
        - Triangle.h
        - Triangle.cpp
        - ShapeManager.cpp
        - ShapeManager.h
    - If any of your files are misnamed there will be reduction in points
- Correct Commenting
    - There should be comments on key areas in your code explaining why you chose certain aspects and what they are doing, a good rule of thumb is you should have a couple of comments every 10 lines of code
- Correct Variable Naming
    - Naming variables in relation to their purpose.
- Code working correctly/compiling

    - Please note: Although your can use whatever compiler you want, your assignment will be graded in the class compiler (VS code and Pycharm) so it must compile with these compilers, or you may lose points
- Code print format is as shown in sample runs

*See rubric on canvas assignment for more details.