Module 1 Day 12

Polymorphism and Interfaces

Polymorphism

- “Many forms”
- Two distinct aspects:
  - If B is a subclass of A and a function can accept A as a parameter, then it can also accept B. If we have a collection of A, we can also store B in the collection
  - Subclasses can override methods defined on the superclass, and the appropriate method gets invoked based on the Type of the target object
- You cannot talk Polymorphism without talking Inheritance
  - (or Interfaces, which we will talk about later)

New Feature Request

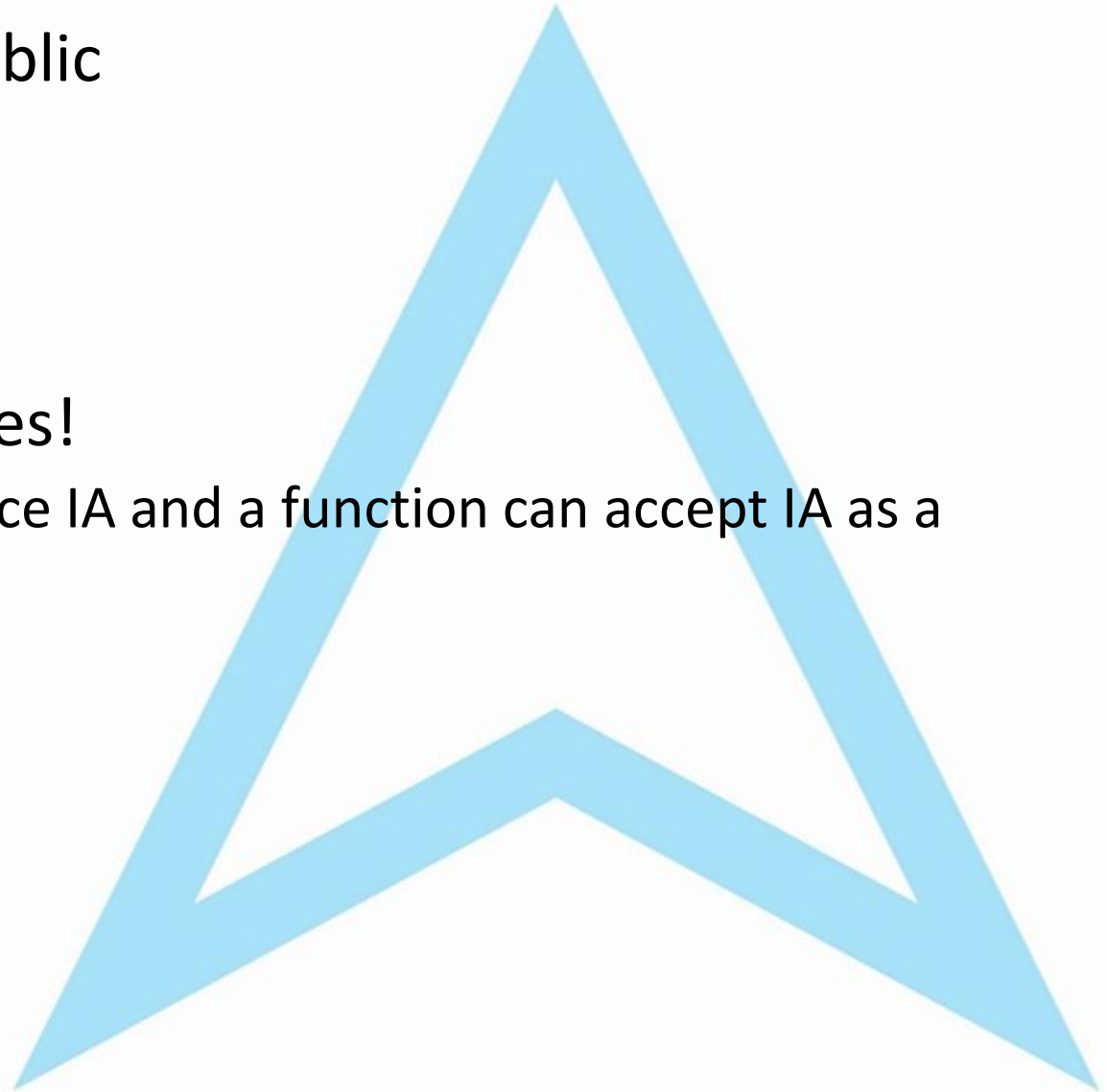
- In addition to shapes, we need to capture, store and print text labels on our drawing.
  - How should we implement this? Are these shapes (is-a)?

Interfaces

- Defines a contract between a class and its user
- Defines the **public** properties and methods, but NEVER the implementation
  - No need for access modifiers because all members are *public* by definition
- Classes which *implement* the interface MUST provide the implementation
  - For **ALL** its methods / properties
- Class inheritance ➔ “is a”; Interface inheritance ➔ “Can do”

Interfaces

- All members of an interface are public
- A class can
  - derive from ONE class
  - Implement MANY interfaces
- Polymorphism works with Interfaces!
  - If B is a class that *implements* interface IA and a function can accept IA as a parameter, then it can also accept B



New Feature Request

- We are also asked to determine the total area of all the shapes. Can we do that?

Interfaces in the .NET Framework

- IComparable
- IEnumerable
- IDisposable
- IDbConnection, IDbCommand, IDataReader