- The aim of C++ class concept is to provide the programmer with a tool for creating new types that can be used conveniently as the built-in types (Chapters 7, 13, 14).
- Additionally, inheritance/dynamic binding and templates provide ways of organizing related classes allowing programmer to take advantage of their relationships.

- Chapter 15 covers inheritance and dynamic binding. Along with data abstraction, inheritance and dynamic binding are fundamental to object-oriented programming (OOP).
- Chapter 16 covers function and class templates. Templates let us write generic classes and functions that are independent of type.

Inheritance (繼承)

- 繼承（inheritance）是物件導向語言重要的機制，繼承讓我們在建立類別時，可以不用宣告全新的成員，反之，你可以指定新類別繼承現有類別的成員。
- 我們稱現有類別為父類別（superclass / parent class / **base class**），新延伸類別為子類別（subclass /child class /extended class / **derived class**）。

See Note

Dynamic Binding

- Implicit derived-to-base conversion is the key behind **dynamic binding**.

- Through **dynamic binding**, we can use the same code to process objects of either base or derived class **interchangeably**.

See Note

Pure Virtual Functions and Abstract Base Class

Abstract Base Class (ABC)

- For general properties of different classes, a base class is used for the **common term**.
- Different derived classes are used under their **common term**. Operations defined in the base class are often **overridden.**
- For clean design, we often like to define the base class as an abstract class (pure interfaces). Functions can be declared in the base class without being implemented. What really happens in run-time **must** then be implemented in the respective derived classes.

Pure Virtual Function

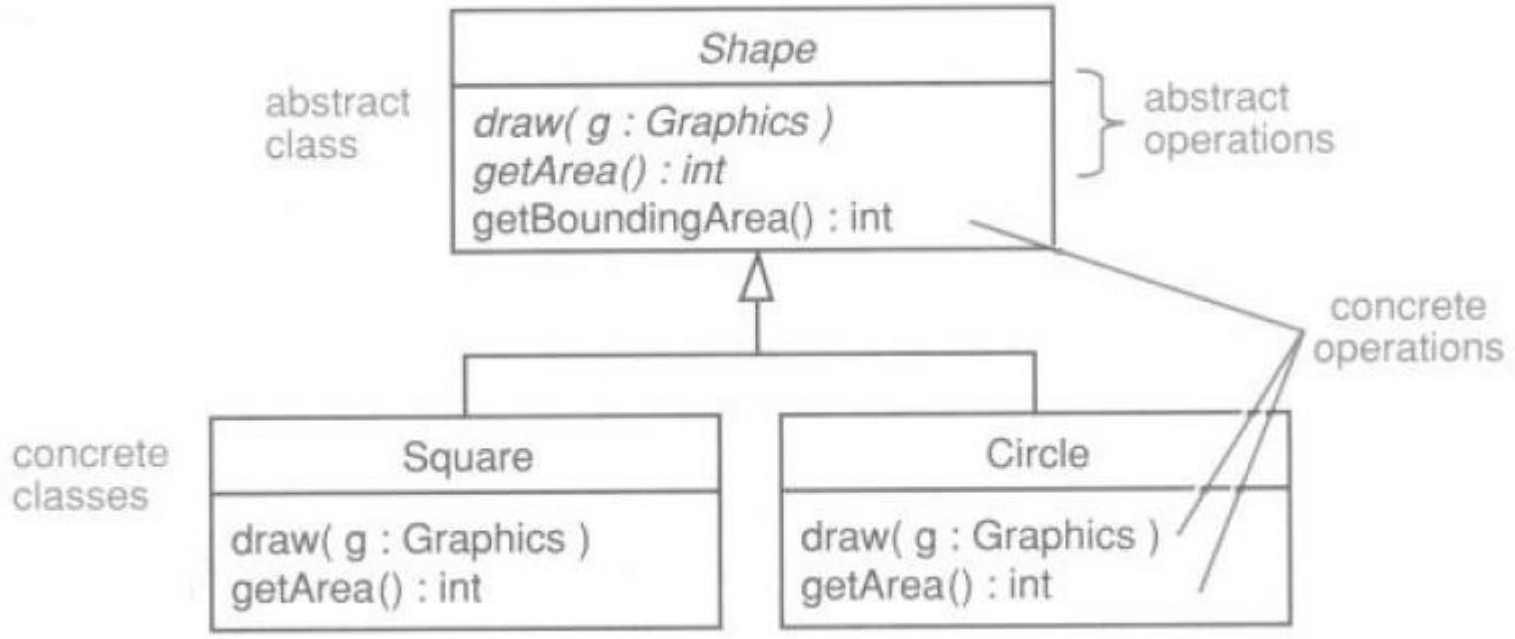
- C++ way to define the abstract interface.
- A pure virtual function is specified by writing =0 after the function parameter list.
- A pure virtual function provides an interface to be override but cannot be called in this class.
- Used very often in Polymorphism.

```
class Base {
public:
    virtual int fcn() = 0;
};
```

Examples of Common Terms

Different objects	Common term
Cars, buses, lorries, bikes	Vehicle
Integers, floating-point values, fractions, complex numbers	NumericValue
Circles, rectangles, lines	GeometricObject
Managers, engineers	Employee

Abstract Class Inheritance in UML



See Note

Containers and Inheritance

- We'd like to use containers to hold objects that are related by inheritance. However, the fact that objects are not polymorphic affects how we can use containers with types in an inheritance hierarchy.
- Can we use reference to the Base class?
- Can we use pointer to the Base class?

Containers and Inheritance

- The viable approach is to use the container to hold pointers to our objects.
- This pushes our users to manage the objects and pointers.
- You can do a more elegant alternative using smart pointers in C++11.

See Note

The Last Class

- Lab 6
- HW 6
- Happy Dragon Boat Festival!