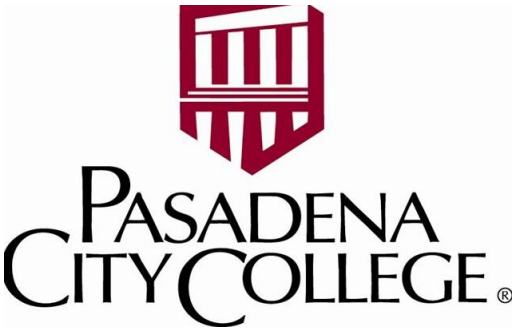
CIS 014 – C++ Programming

Lecturer: Joseph Su



REFERENCES

Optional Textbook:

Programming: Principles and Practice Using C++, 2nd ed, B. Stroustrup, Addison-Wesley, 2014

PDF:

http://www.cplusplus.com/files/tutorial.pdf

Online:

http://www.cplusplus.com/doc/tutorial/

The C++ Programming Language, 4th ed.

B. Stroustrup, Addison-Wesley, 2013

C++ How to Program, 10th ed

Deitel & Deitel, Pearson Hall, 2016

C++ Primer, 5th ed

S. Lippman, J. Lajoie, and B. Moo, Addison-Wesley, 2012

READING ASSIGNMENTS

ONLINE

- C++ Inheritance (Tutorialpoint)
- C++ Inheritance (W3School)
- Polymorphism
- C++ Polymorphism (W3School)
- Derived Classes

REFERENCES

http://www.cplusplus.com/files/tutorial.pdf (pages 1-112)
http://www.cplusplus.com/doc/tutorial/

- ✓ Program Structure
  - Complete all chapters
- ✓ Compound Data Types
- ✓ Classes
  - Classes I, Classes II
  - Inheritance, polymorphism, etc.

TODAY

- Inheritance/Polymorphism/Derived Class
- Inheritance: concept

INHERITANCE/POLYMORPHISM/ DERIVED CLASS

- Inheritance
  - New classes created from existing classes
  - Absorb attributes and behaviors.
- Polymorphism
  - Write programs in a general fashion
  - Handle a wide variety of existing (and unspecified) related classes
- Derived class
  - Class that inherits data members and member functions from a previously defined base class

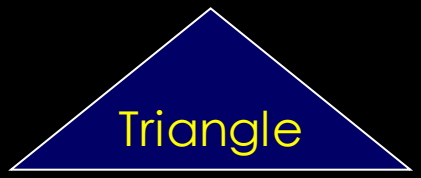
INHERITANCE: CONCEPT

- Single Inheritance
  - Class inherits from one base class
- Multiple Inheritance
  - Class inherits from multiple base classes
- Three types of inheritance:
  - `public`: Derived objects are accessible by the base class objects
  - `private`: Derived objects are inaccessible by the base class
  - `protected`: Derived classes and `friend`s can access protected members of the base class

INHERITANCE: CONCEPT

Polygon

Rectangle

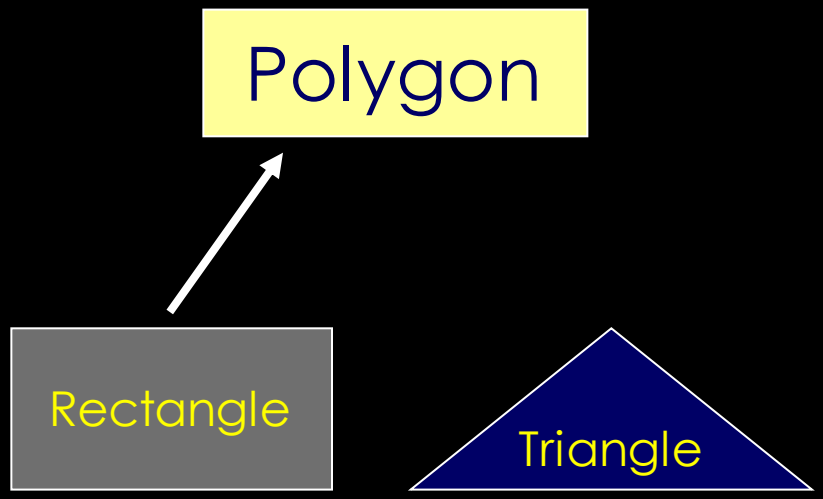



```
class Rectangle {
      private:
       int numVertices;
       float *xCoord, *yCoord;
      public:
        void set(float *x, float *y, int nV);
         float area();
};
```

```
class Polygon {
     private:
      int numVertices;
      float *xCoord, *yCoord;
     public:
      void set(float *x, float *y, int nV);
};
```

```
class Triangle {
      private:
       int numVertices;
       float *xCoord, *yCoord;
      public:
       void set(float *x, float *y, int nV);
        float area();
};
```

INHERITANCE: CONCEPT

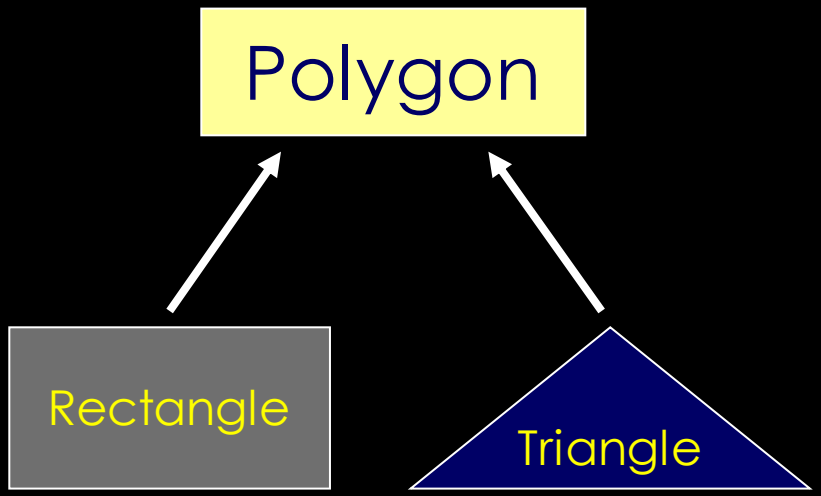



```
class Polygon {
    protected:
      int numVertices;
      float *xCoord, float *yCoord;
    public:
       void set(float *x, float *y, int nV);
};
```

```
class Rectangle : public Polygon {
    public:
       float area();
};
```

```
class Rectangle {
    protected:
      int numVertices;
      float *xCoord, float *yCoord;
    public:
       void set(float *x, float *y, int nV);
       float area();
};
```

INHERITANCE: CONCEPT

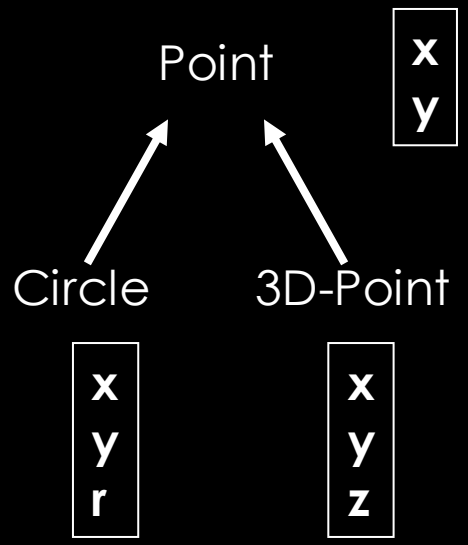



```
class Polygon {
    protected:
      int numVertices;
      float *xCoord, float *yCoord;
    public:
      void set(float *x, float *y, int nV);
};
```

```
class Triangle : public Polygon {
    public:
     float area();
};
```

```
class Triangle {
    protected:
      int numVertices;
      float *xCoord, float *yCoord;
    public:
      void set(float *x, float *y, int nV);
      float area();
};
```

INHERITANCE: CONCEPT



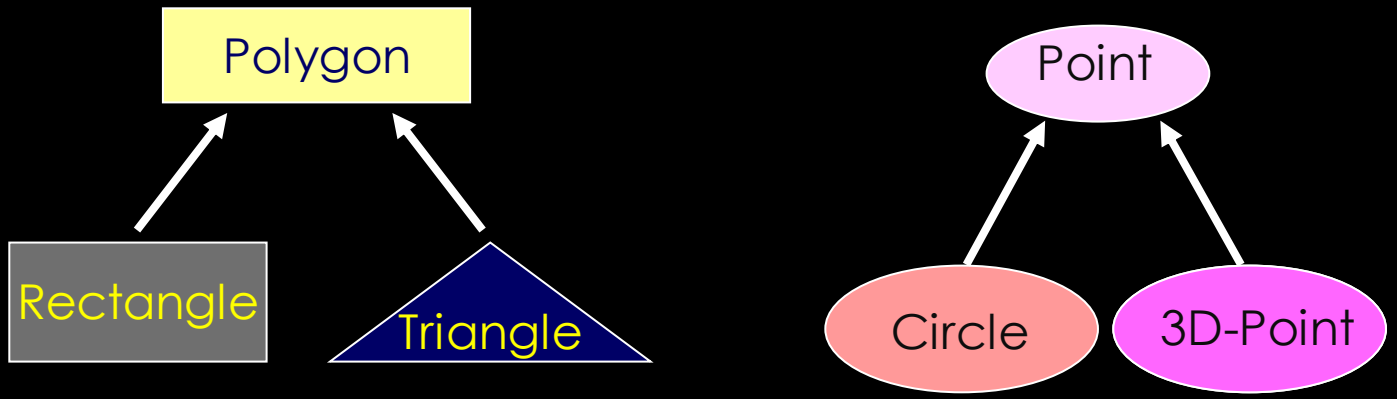

```
class Point {
    protected:
      int x, y;
    public:
      void set (int a, int b);
};
```

```
class Circle : public Point {
    private:
     double r;
};
```

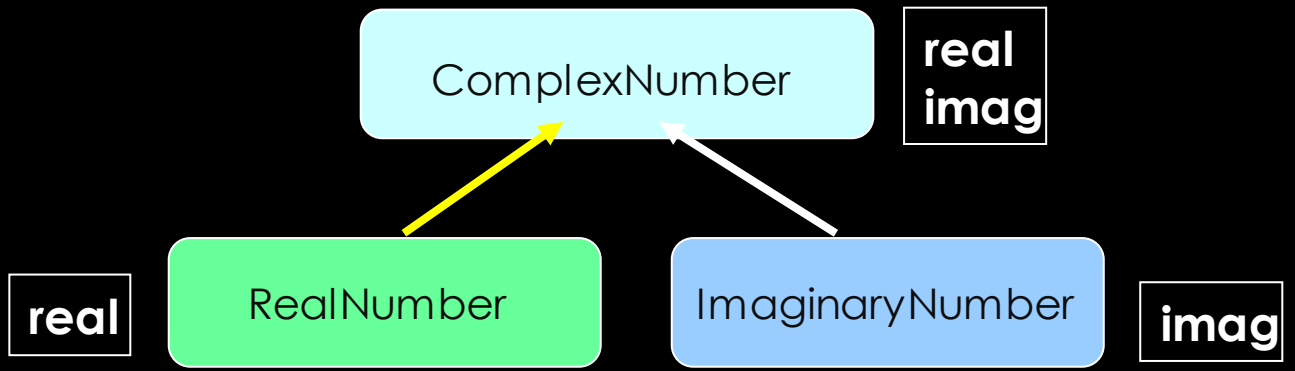
```
class 3D-Point: public Point {
    private:
     int z;
};
```

INHERITANCE: CONCEPT

- AUGMENTING THE ORIGINAL CLASS



- SPECIALIZING THE ORIGINAL CLASS



BASE CLASS / DERIVED CLASES

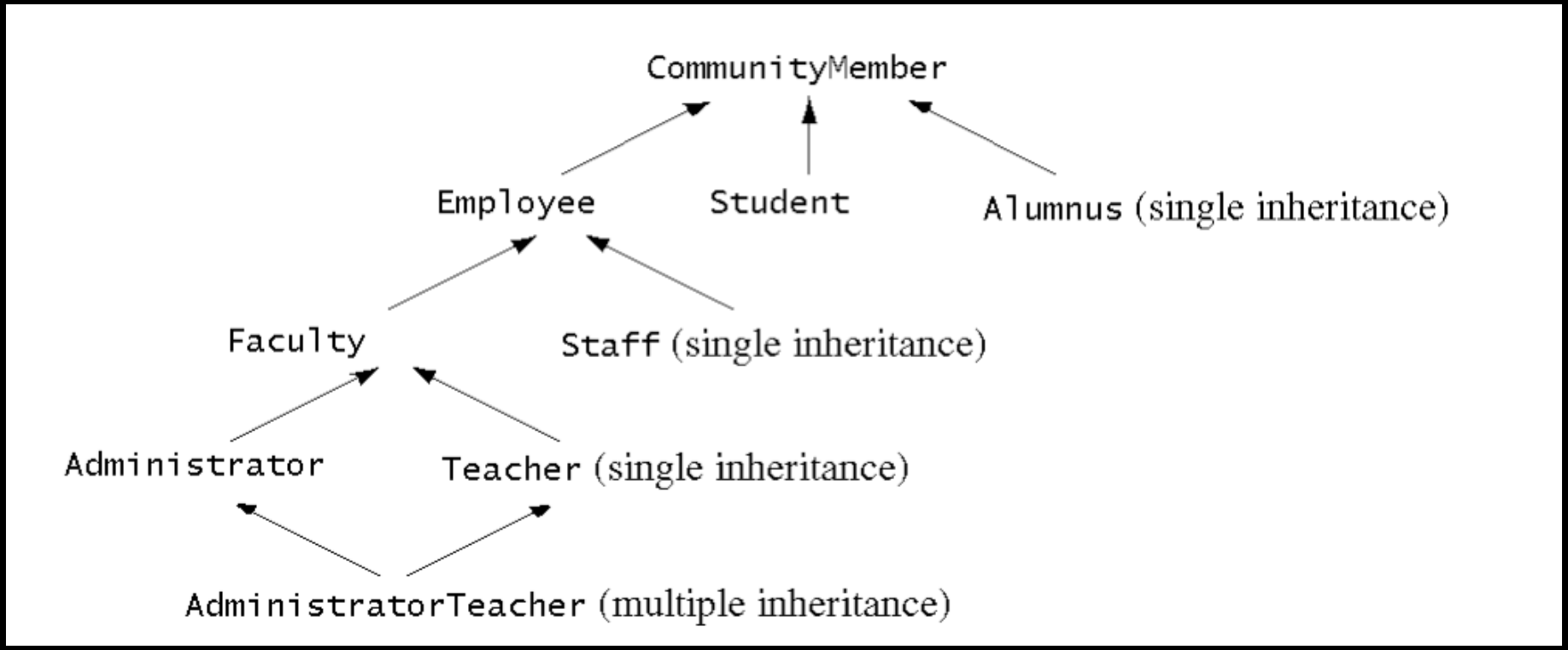
- Sometimes base classes are also called parent or superclasses.
- Derived classes are called children or subclasses.

Examples:

Base class	Derived classes
Student	GraduateStudent UndergraduateStudent
Shape	Circle Triangle Rectangle
Loan	CarLoan HomeImprovementLoan MortgageLoan
Employee	FacultyMember StaffMember
Account	CheckingAccount SavingsAccount

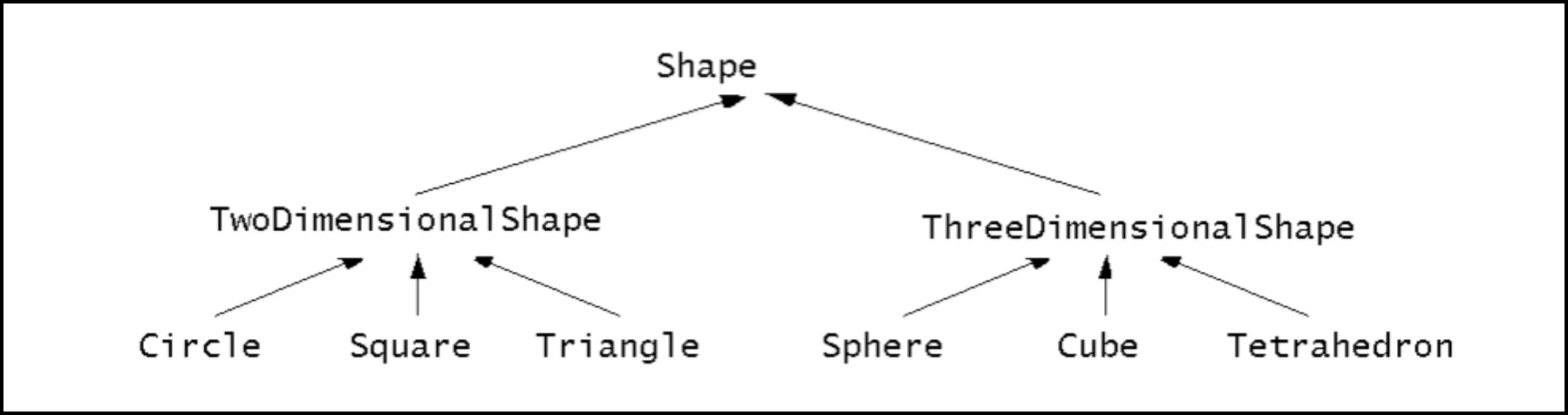
BASE CLASS / DERIVED CLASES

Example: college members and their relationship in the inheritance hierarchy



BASE CLASS / DERIVED CLASES

Example: Shapes and their relationship with each other in the inheritance hierarchy



PUBLIC INHERITANCE

- Implementation of `public` inheritance

```
class Student : public People {
   …
};
```

- Class Student inherits from class People
  - `friend` functions are not inherited.
  - `private` members of base class are not accessible from the derived class

PUBLIC INHERITANCE

- Derived class object can be treated as an object of the base class

```
class Student : public People {
   …
};

// s can be treated as an object of the People class
// superclass, because Student "is…a" People
Student* s = new Student();
```

- Base class object cannot be treated as an object of the derived class .

PROTECTED INHERITANCE

Intermediate level of protection between public and private inheritance.

- Derived-class members can refer to public and protected members of the base class simply by using the member names.
- Note that protected data, just like friend data, "breaks" data encapsulation.

INHERITANCE: PUBLIC, PROTECTED, PRIVATE

```
class A
{
public:
    int x;
protected:
    int y;
private:
    int z;
};

class B : public A
{
    // x is public
    // y is protected
    // z is not accessible from B
};

class C : protected A
{
    // x is protected
    // y is protected
    // z is not accessible from C
};

class D : private A
{
    // x is private
    // y is private
    // z is not accessible from D
};
```