Copy constructors

- Tell a class how instances of it should be created from another instances
- When we have an Object obj{}; and need to create another Object obj2{obj};
- Syntax -- ClassName(const ClassName& object)
- Compiler automatically generates default copy constructor if we do not supply one, but there are scenarios in which no default members are generated at all
- Implementation syntax: ClassName::ClassName(const ClassName& instance)
- Has a member initialization list, as any constructor

Inheritance

Inheritance is a mechanism for

- building class types from existing class types
- defining new class types to be a
  - specialization
  - augmentation

of existing types

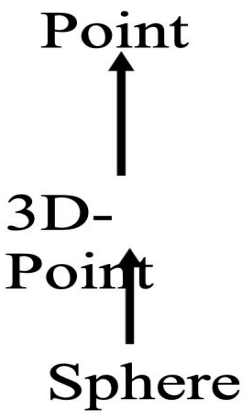
Inheritance -- syntax

- Syntax:

  class *DerivedClassName* : access-level *BaseClassName*

  where
  - access-level specifies the type of derivation
    - private by default, or
    - public
- Any class can serve as a base class
  - Thus a derived class can also be a base class

Inheritance example



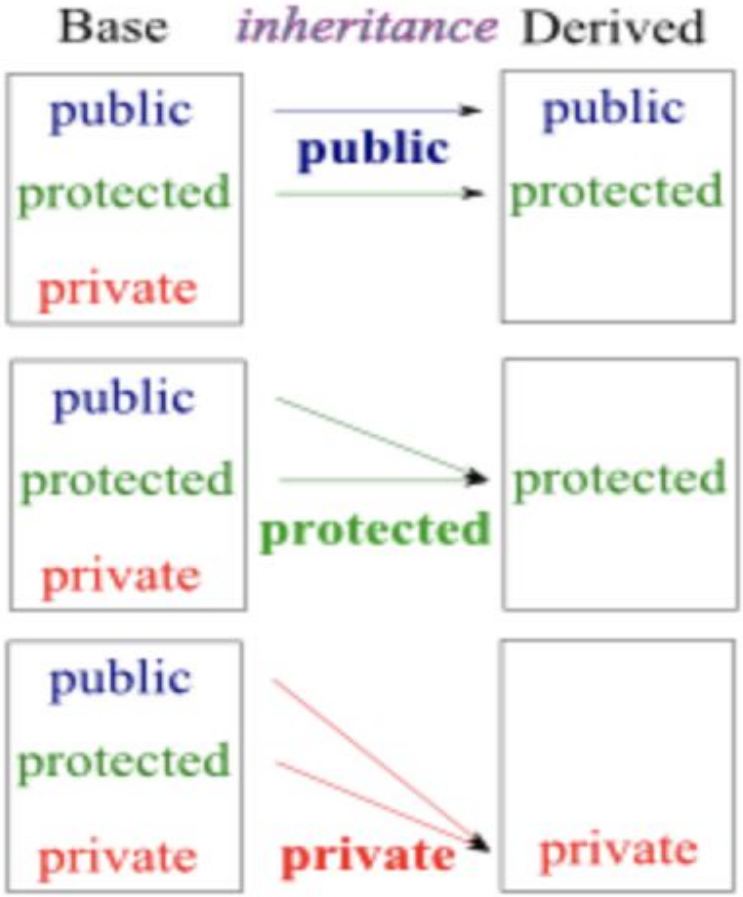

```
class Point{
  protected:
    int x, y;
  public:
    void set (int a, int b);
};
```

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

```
class Sphere : public 3D-Point{
  private:
     double r;
  … …
};
```

Point is the base class of 3D-Point, while 3D-Point is the base class of Sphere

Types of inheritance



Inheritance

- **Multiple inheritance**
- **Diamond problem**
- **Virtual inheritance**