

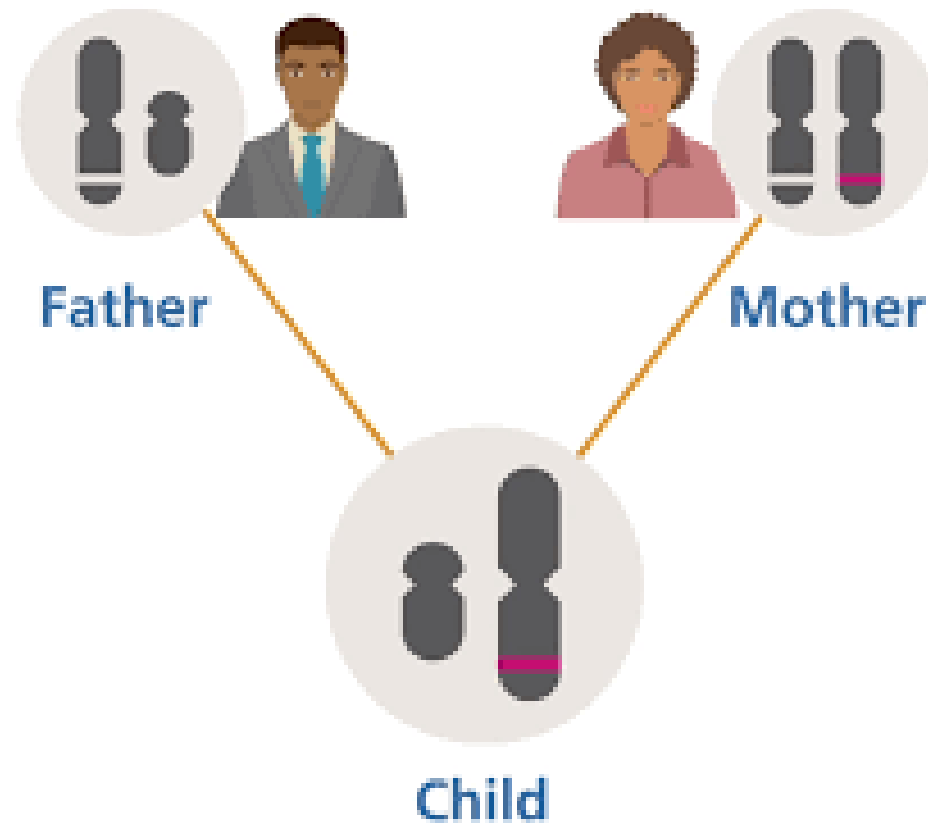


**COURSE CODE: CSE 205**

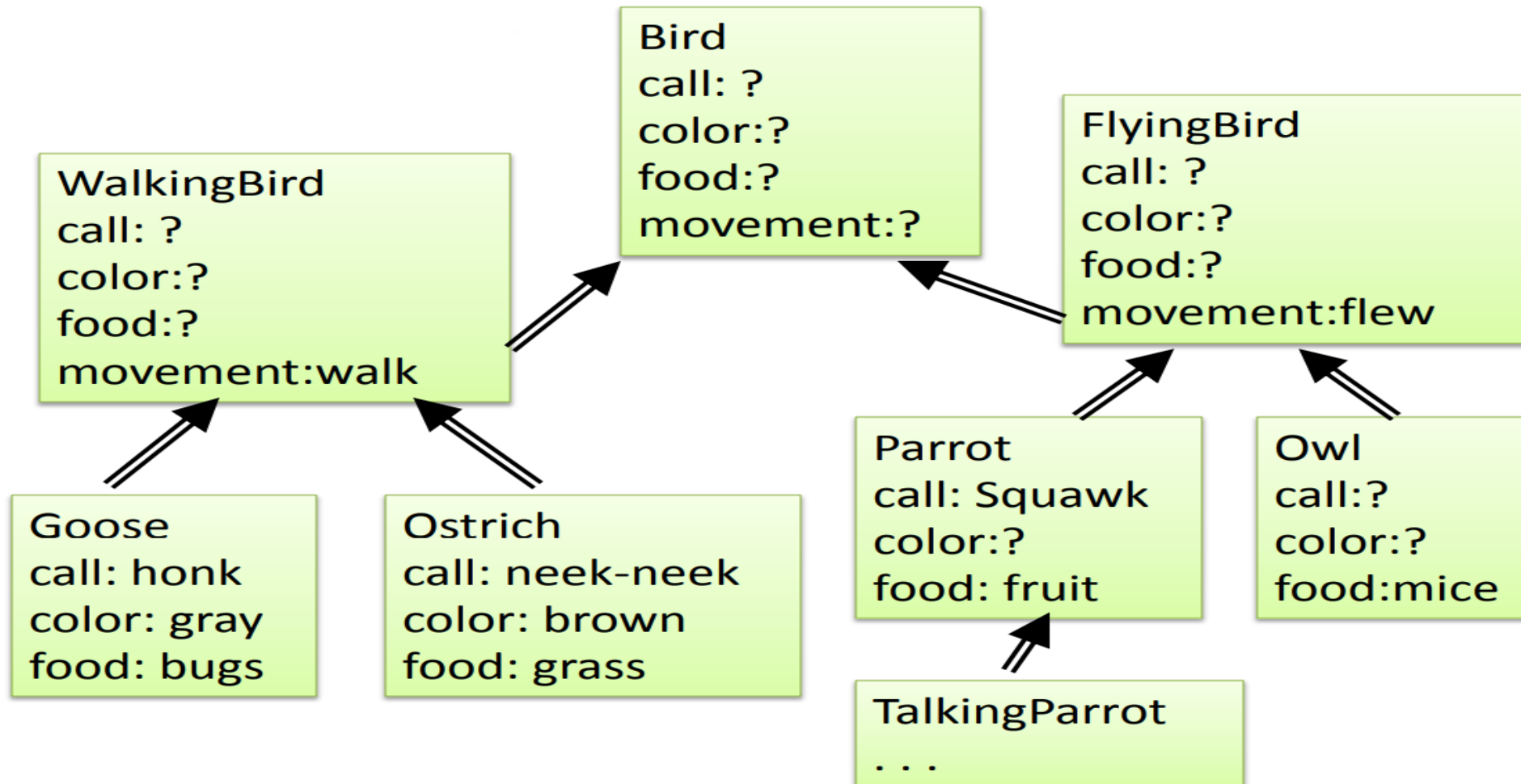
**COURSE TITLE: OBJECT ORIENTED PROGRAMMING**

***Presented By: Lec Faria Alam***

# What Is Inheritance in General?



# What Is Inheritance?



# What Is Inheritance in OOP?

- Inheritance is the mechanism by which one class can inherit the properties of another.
- The new class is a specialized version of the existing class
- Two concepts of Inheritance
  - derived class (child) - The class that inherits properties from another class
  - base class (parent) - The class whose properties are inherited by a derived class



# Inheritance

```
class Point
{
public:
    int x;
    int y;
};
```


# Inheritance

```
class Point
{
public:
    int x;
    int y;
};
```

```
class Point3D
{
public:
    int x;
    int y;
    int z;
};
```

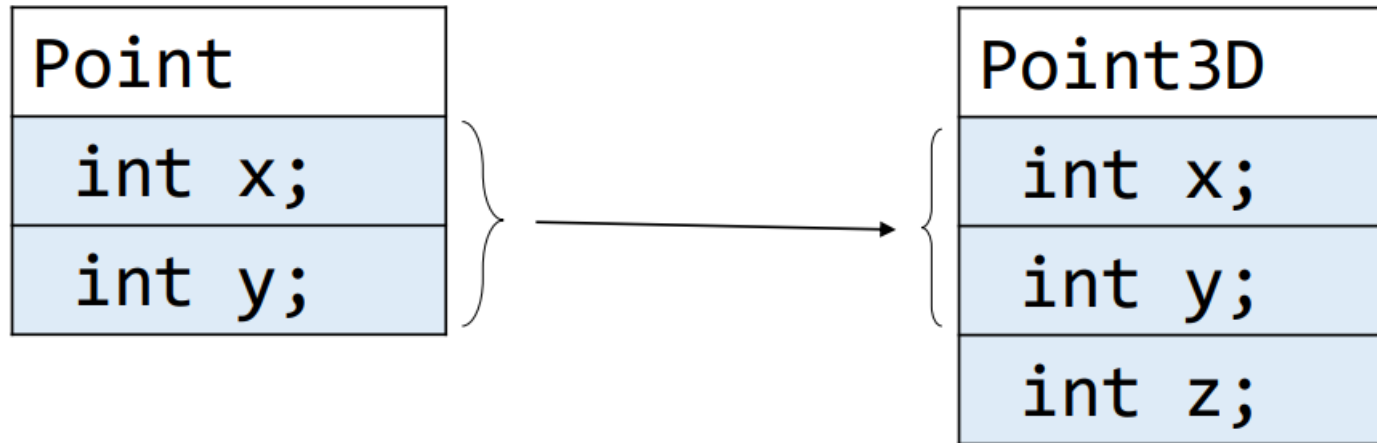
# Inheritance

```
class Point
{
public:
    int x;
    int y;
};
```

An arrow points from the 'public:' section of the Point class to the 'public:' section of the Point3D class, indicating that Point3D inherits from Point.

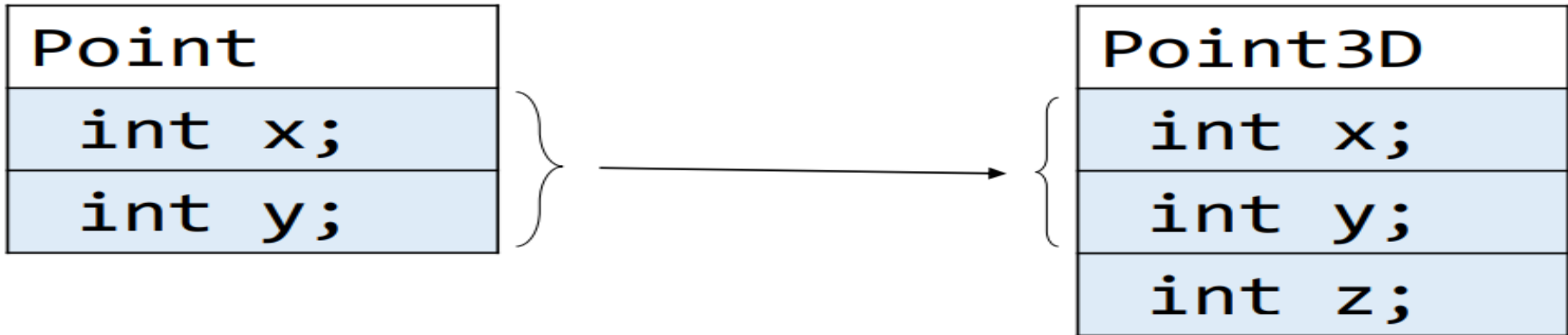
```
class Point3D
{
public:
    int x;
    int y;
    int z;
};
```

# Inheritance

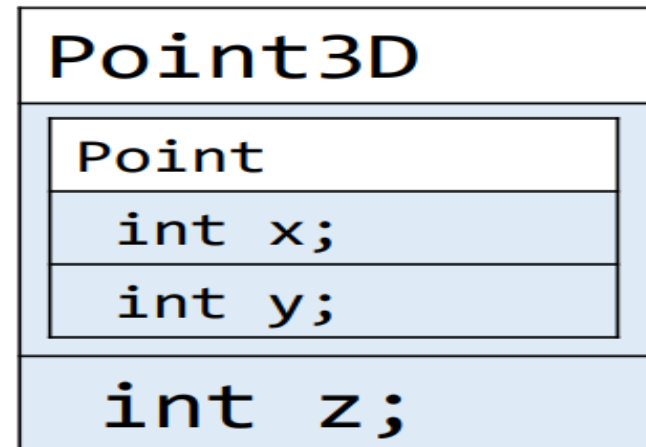




# Inheritance



We need something like this:



# Inheritance

```
class Point ← Base Class
```

```
{  
public:  
    int x;  
    int y;  
};
```

Derived Class

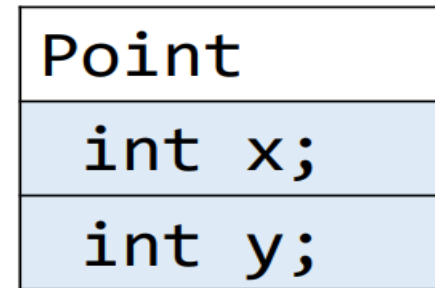
```
class Point3D : public Point  
{  
  
};
```

Point
int x;
int y;

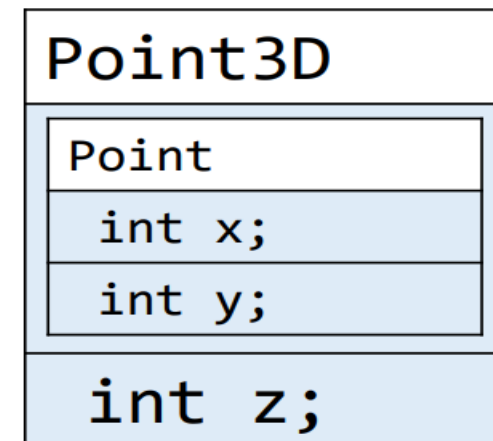
Point3D			
<table><tr><th>Point</th></tr><tr><td>int x;</td></tr><tr><td>int y;</td></tr></table>	Point	int x;	int y;
Point			
int x;			
int y;			

# Inheritance

```
class Point
{
public:
    int x;
    int y;
};
```



```
class Point3D : public Point
{
public:
    int z;
};
```



*Class designer's view*



# Inheritance

## Student Class

 Private member

Student
<pre>int theory; int sessional;</pre>
<pre>void setMarks(...); int getTheory(); int getSessional();</pre>

# Inheritance

As we know, each student at every level has both theoretical and sessional courses. However, at level 4, there is an additional course called 'thesis.' Therefore, we need to design a class specifically for Level 4 students.

Student
<code>int theory;</code> <code>int sessional;</code>
<code>void setMarks(_);</code> <code>int getTheory();</code> <code>int getSessional();</code>

L4Student



# Extending Student Class

 Private member

Student
<code>int theory;</code> <code>int sessional;</code>
<code>void setMarks(...);</code> <code>int getTheory();</code> <code>int getSessional();</code>

L4Student
<code>...</code> <code>int thesis;</code>
<code>...</code> <code>void setThesis(...);</code> <code>int getThesis();</code>

# Extending Student Class

L4Student
... int thesis;
... void setThesis(...); int getThesis();

 Private member

L4 students also have theoretical and sessional courses. Therefore, in the L4Student class, we also need to implement functionalities for setting theory and sessional marks. We already have a class called 'Student' that includes these functionalities, so we can inherit from that class.

# Extending Student Class

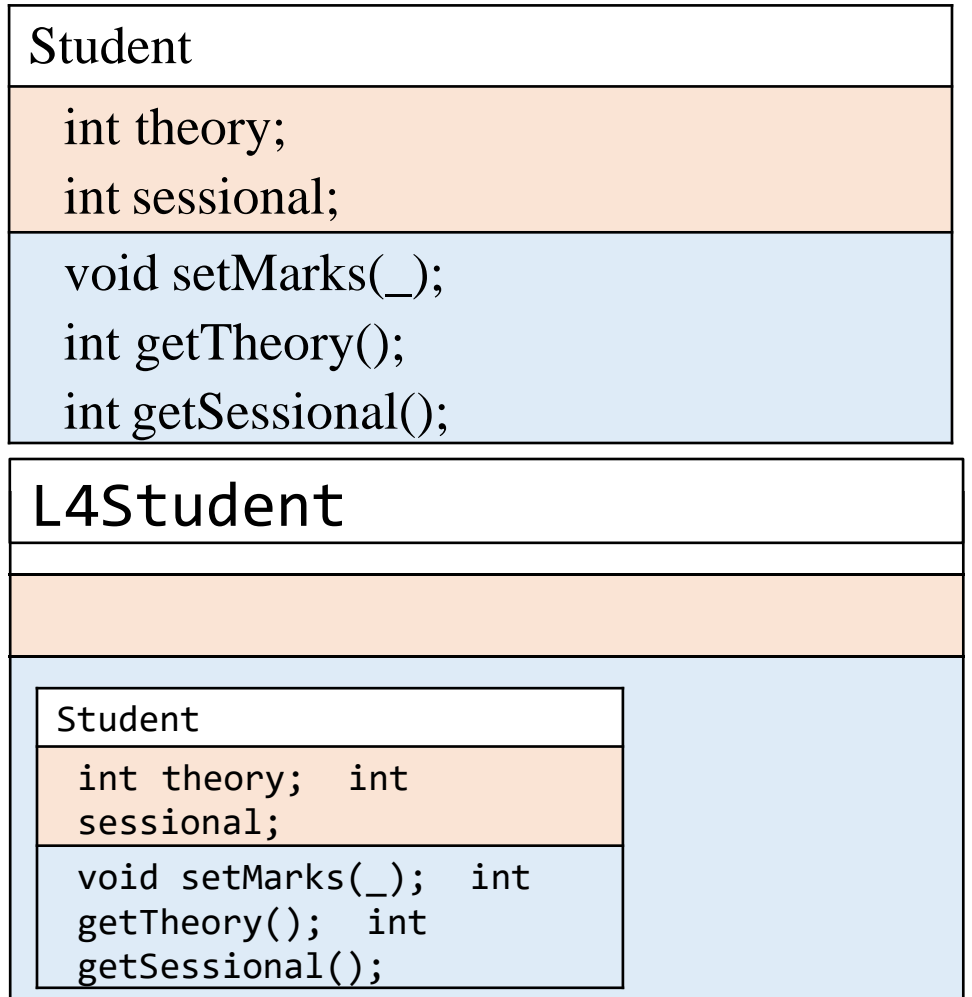
```
class L4Student : public Student
```

Student
int theory; int sessional;
void setMarks(_); int getTheory(); int getSessional();

L4Student

# Extending Student Class

```
class L4Student : public Student
```



# Extending Student Class

```
class L4Student : public Student
```

Student
int theory; int sessional;
void setMarks(_); int getTheory(); int getSessional();

L4Student			
int thesis;			
<table><tr><th>Student</th></tr><tr><td></td></tr><tr><td>void setMarks(_); int getTheory(); int getSessional();</td></tr></table>	Student		void setMarks(_); int getTheory(); int getSessional();
Student			
void setMarks(_); int getTheory(); int getSessional();			
void setThesis(...); int getThesis();			








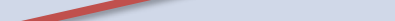
# Extending Student Class

**class** L4Student : **private** Student

Student
<code>int theory;</code> <code>int sessional;</code>
<code>void setMarks(_);</code> <code>int getTheory();</code> <code>int getSessional();</code>

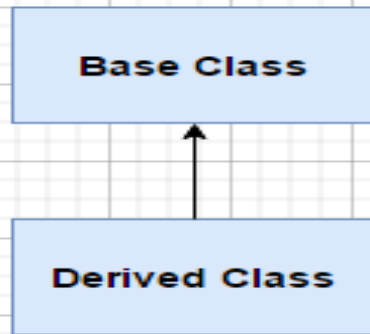
L4Student			
<code>int thesis;</code> <table><tr><th>Student</th></tr><tr><td></td></tr><tr><td><code>void setMarks(_);</code> <code>int getTheory();</code> <code>int getSessional();</code></td></tr></table>	Student		<code>void setMarks(_);</code> <code>int getTheory();</code> <code>int getSessional();</code>
Student			
<code>void setMarks(_);</code> <code>int getTheory();</code> <code>int getSessional();</code>			
<code>void setThesis(...);</code> <code>int getThesis();</code>			

# Access Specifier Summary

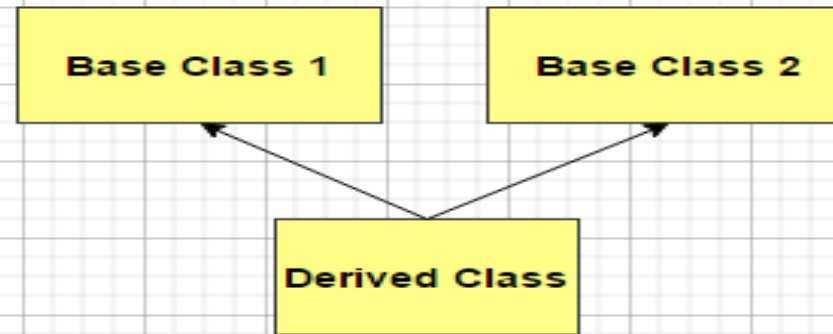
Inheritance	Base Class	Derived Class
public	Private: Protected:  Public: 	Private: Protected: Public:
protected	Private: Protected:  Public: 	Private: Protected: Public:
private	Private: Protected:  Public: 	Private: Protected: Public:

# Types of Inheritance

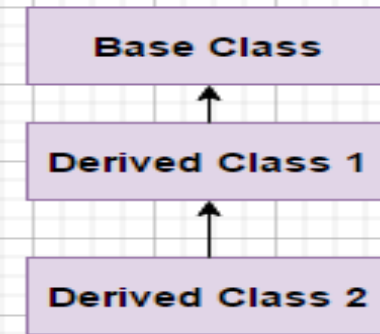
single Inheritance



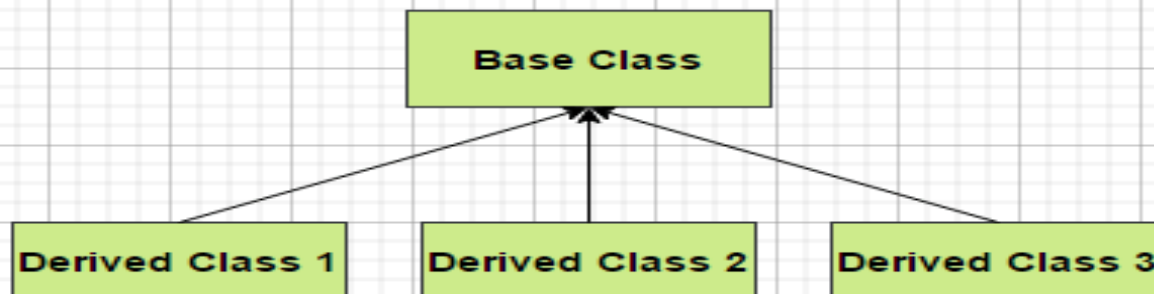
Multiple Inheritance



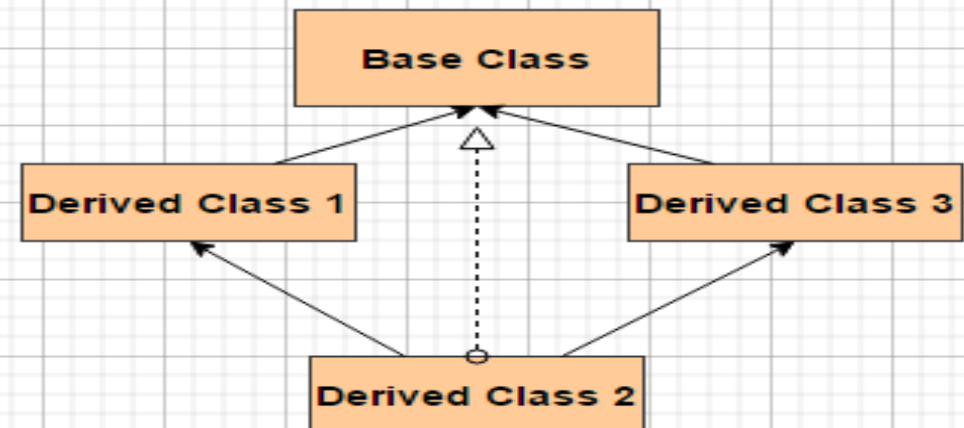
MultiLevel Inheritance



hierarchial Inheritance



Hybrid Inheritance



*Thank  
you!*