Object-Oriented Programming (OOP)

Week – 07

Mar 16-20, 2020

Instructor: **Basit Ali**

Email: basit.jasani@nu.edu.pk

Object-Oriented Programming (OOP)

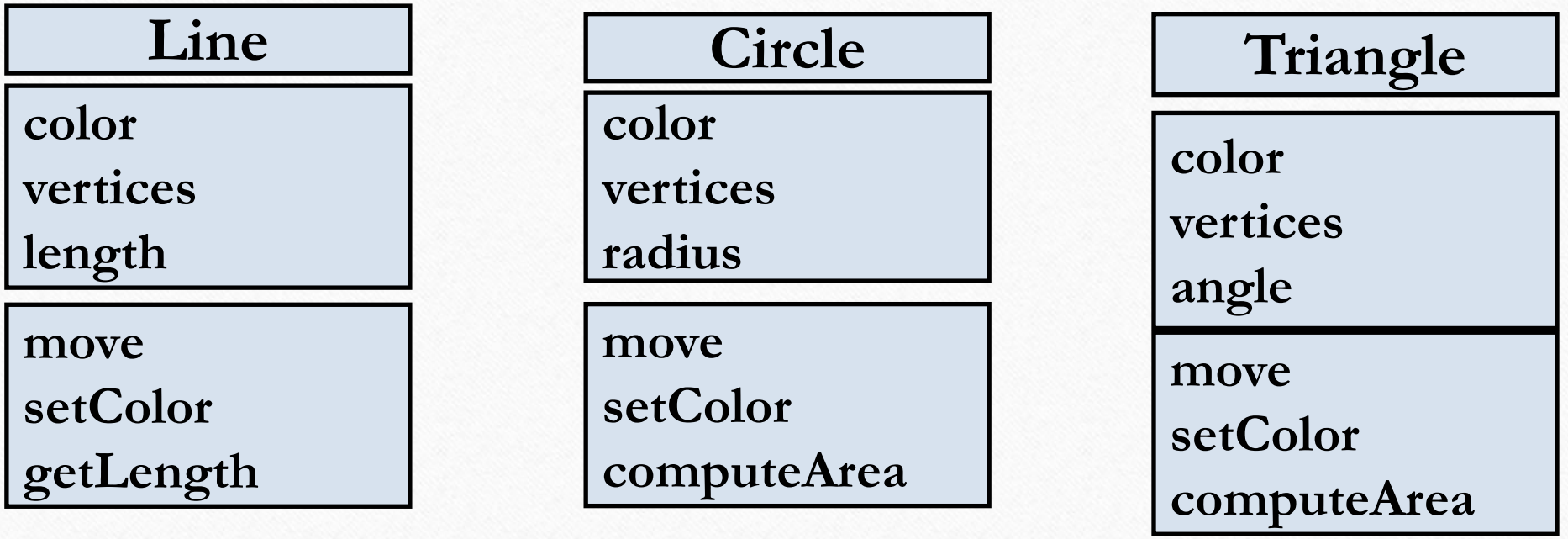
Recap – Inheritance

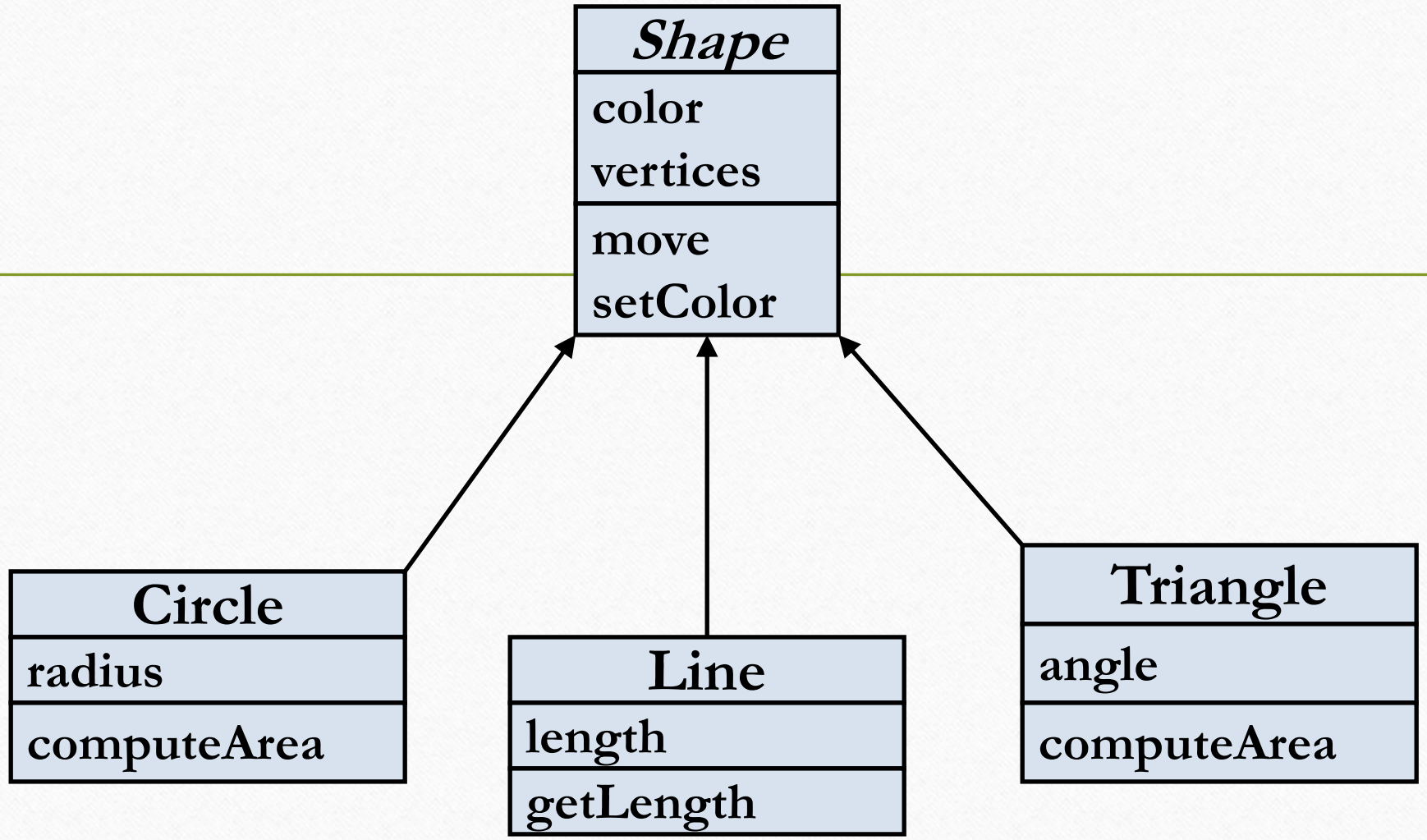
- Derived class inherits all the characteristics of the base class
- Besides inherited characteristics, derived class may have its own unique characteristics
- Major benefit of inheritance is reuse

Generalization

- In OO models, some classes may have common characteristics
- We extract these features into a new class and inherit original classes from this new class
- This concept is known as Generalization

Example – Generalization

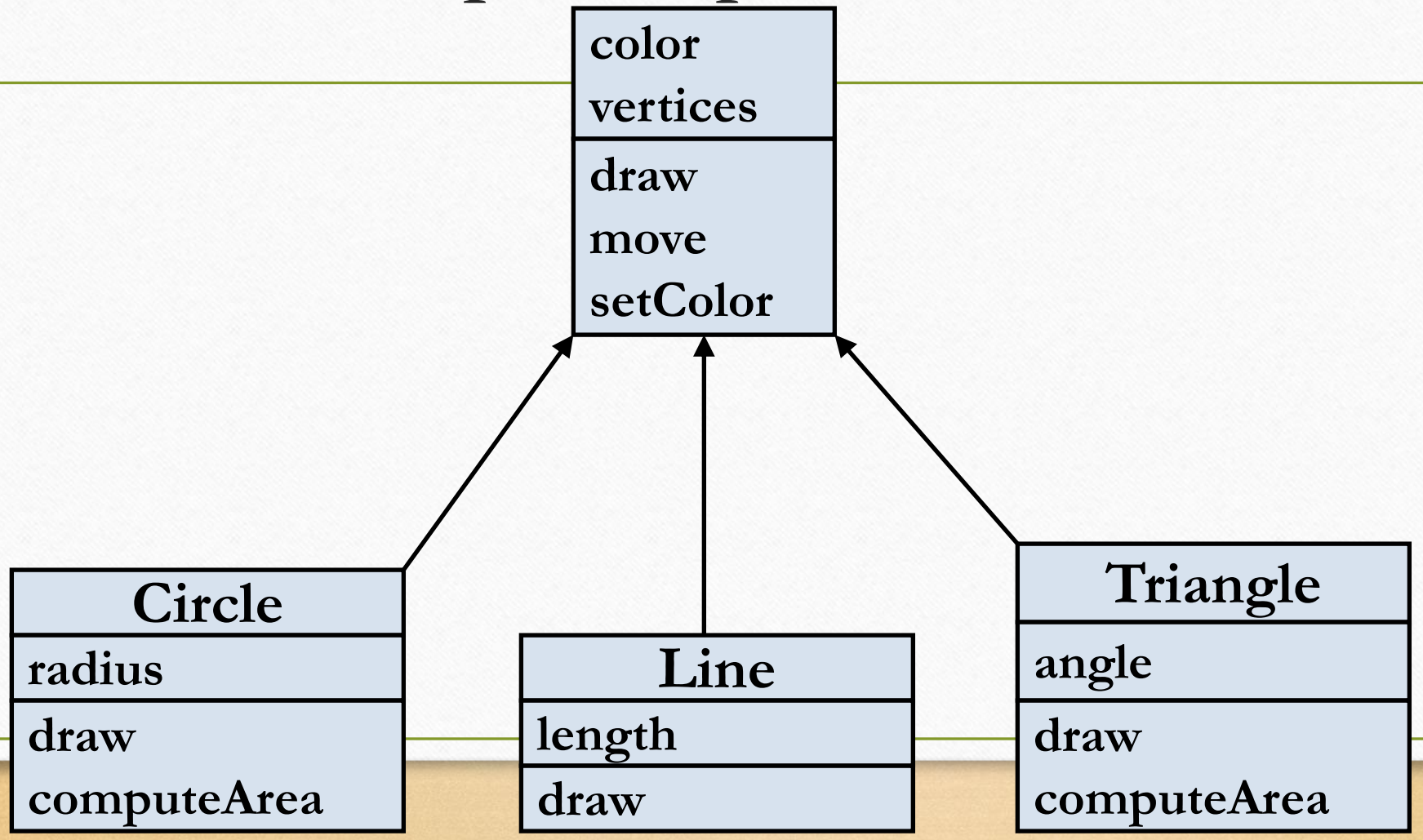



Shape
color
vertices
move
setColor
Circle
radius
computeArea
Line
length
getLength
Triangle
angle
computeArea

Overriding

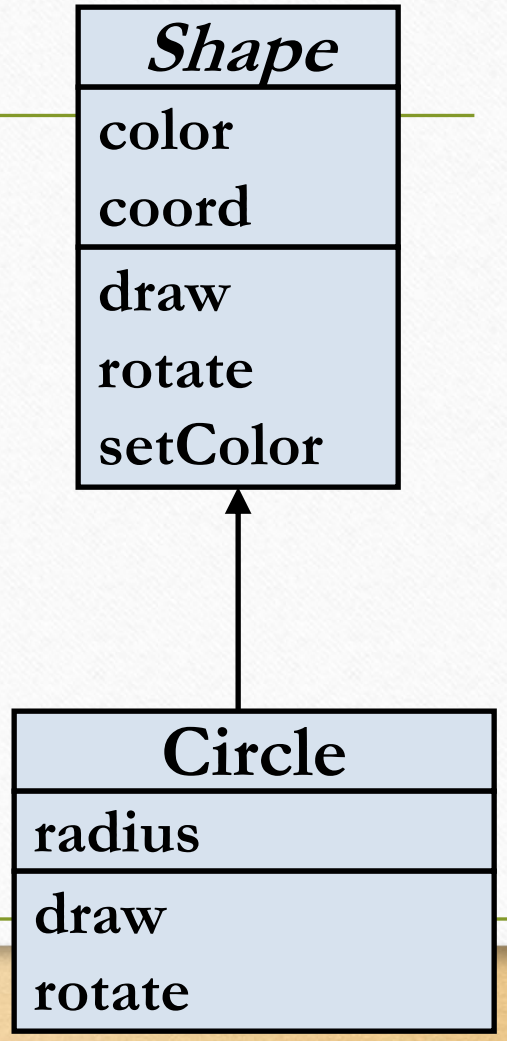
- A class may need to override the default behavior provided by its base class
- Reasons for overriding
  - Provide behavior specific to a derived class
  - Extend the default behavior
  - Restrict the default behavior
  - Improve performance

Example – Specific Behavior



Example – Improve Performance

- Class Circle overrides *rotate* operation of class Shape with a Null operation.



Polymorphism

- The process of representing one Form in multiple forms is known as **Polymorphism**

- Polymorphism is derived from 2 Greek words: **poly** and morphs. The word "poly" means many and **morphs** means forms. So polymorphism means many forms.

Real life example of Polymorphism

Suppose if you are in class room that time you behave like a student, when you are in market at that time you behave like a customer, when you at your home at that time you behave like a son or daughter, Here one person have different-different behaviors.

Type of Polymorphism

- Static / Compile time polymorphism
- Dynamic / Run time polymorphism

Static / Compile time polymorphism

- It is also called Early Binding
- It happens where more than one methods share the same name with different parameters or signature and different return type.
- It is **known** as Early Binding because the **compiler** is aware of the functions with same name and also which overloaded function is to be **called** is **known** at **compile time.**

Static / Compile time polymorphism

- **Overloading**
  - Function Overloading (ALREADY DISCUSSED)
  - Constructor Overloading (ALREADY DISCUSSED)
  - Operator Overloading (TO BE DISCUSSED)

Dynamic / Run time polymorphism

- This refers to the entity which changes its form depending on circumstances at runtime. This concept can be adopted as analogous to a chameleon changing its color at the sight of an approaching object.
- Method Overriding uses runtime Polymorphism.
- It is also called Late Binding.

Dynamic / Run time polymorphism

- Runtime Polymorphism is done using virtual and inheritance.
- When overriding a method, the behavior of the method is changed for the derived class.

Method Overloading

- Whenever same method name is exiting multiple times in the same class with different number of parameter or different order of parameters or different types of parameters is known as **method overloading**

Example

```
2   #include<iostream.h>
3   #include<conio.h>
4
5   class Addition
6   {
7   public:
8   void sum(int a, int b)
9   {
10  cout<<a+b;
11  }
12  void sum(int a, int b, int c)
13  {
14  cout<<a+b+c;
15  }
16  };
```

```
18  void main()
19  {
20  clrscr();
21  Addition obj;
22  obj.sum(10, 20);
23  cout<<endl;
24  obj.sum(10, 20, 30);
25  }
```

Output:

30

60

Method Overriding

- Define any method in both base class and derived class with same name, same parameters or signature, this concept is known as **method overriding**

Example

```
1   #include<iostream.h>
2   #include<conio.h>
3
4   class Base
5   {
6    public:
7    void show()
8    {
9     cout<<"Base class";
10   }
11  };
12
13  class Derived:public Base
14  {
15   public:
16   void show()
17   {
18    cout<<"Derived Class";
19   }
20  };
```

```
22  int mian()
23  {
24   Base b;        //Base class object
25   Derived d;  //Derived class object
26   b.show();     //Early Binding Ocuurs
27   d.show();
28  getch();
29  }
```

Output:
Base class
Derived class

Exercise

- *Consider the following scenario:*
- There is a motor repair shop which has different mechanics working in it. The mechanics are paid salary and their names and experience is considered. The shop is reliable and has a lot of customers. Each mechanic can repair or paint a car and generate bill accordingly. The customers have to pay *Rs.5000* for repair and *Rs10,000* for paint. Furthermore, some of the customer are valued customers, for these customers the mechanic charges only *Rs.3000* for repair and *Rs.7500* for paint. A mechanic is paid 5% of this charged amount as daily wages (apart from monthly salary).