

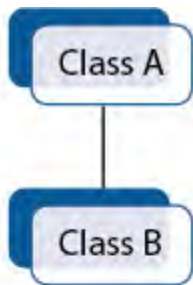
Classes in Python



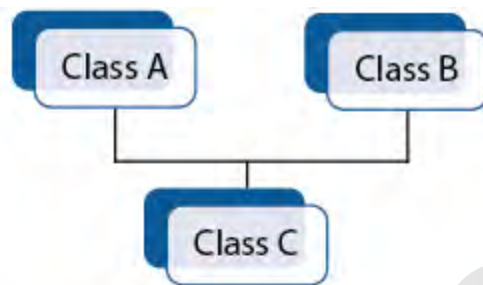
Class is a blueprint of a house

Objects

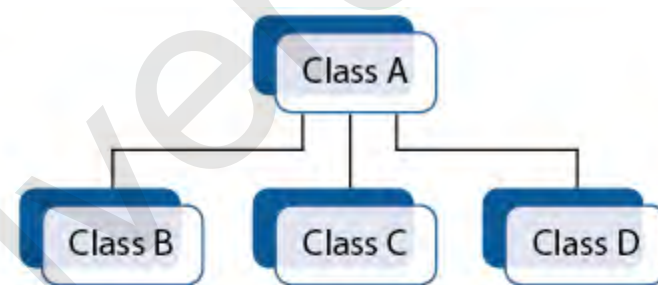
Types of inheritance



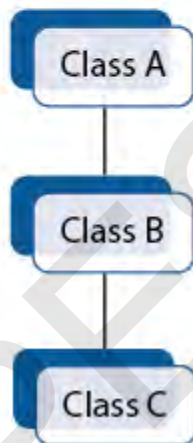
Single Inheritance



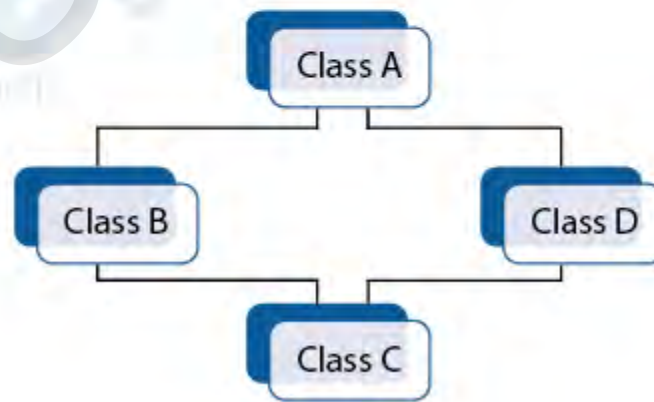
Multiple Inheritance



Hierarchical Inheritance



Multilevel Inheritance



Hybrid Inheritance

Method overriding

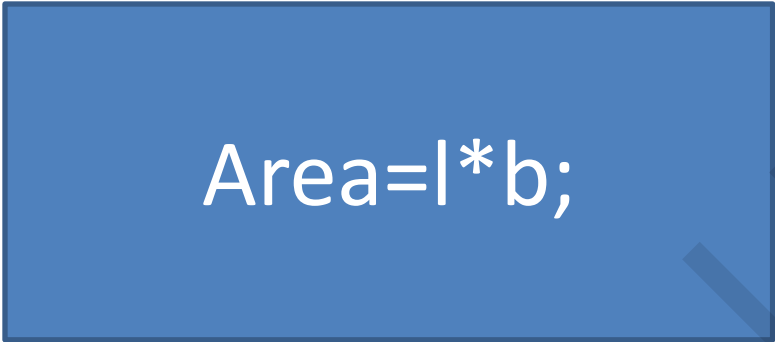
```
class A:  
    def display(self):  
        print("this is class A")  
class B(A):  
    def display(self):  
        print("this is class B")
```

```
a=A()  
a.display()  
b=B()  
b.display()
```

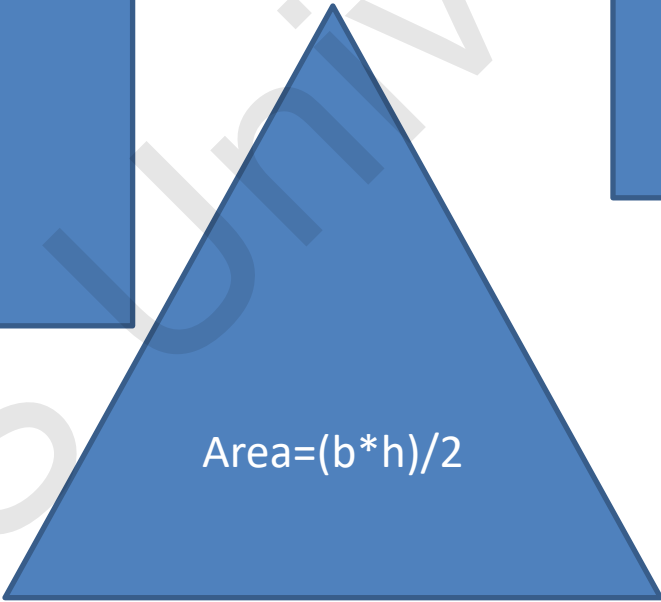
Python Class: Polymorphism

```
class Animal:
    def talk(self):
        pass
class Dog(Animal):
    def talk(self):
        print('bow bow!')
class Cat(Animal):
    def talk(self):
        print('MEOW!')
c= Cat()
c.talk()
d=Dog()
d.talk()
```

Polygons


$$\text{Area} = l * b;$$


$$\text{Area} = s^2$$


$$\text{Area} = (b * h) / 2$$


$$\text{Area} = l * b$$


$$(3 * \sqrt{3} * s^2) / 2$$

Polymorphism

```
class Polygon:
    def findarea(self):
        pass

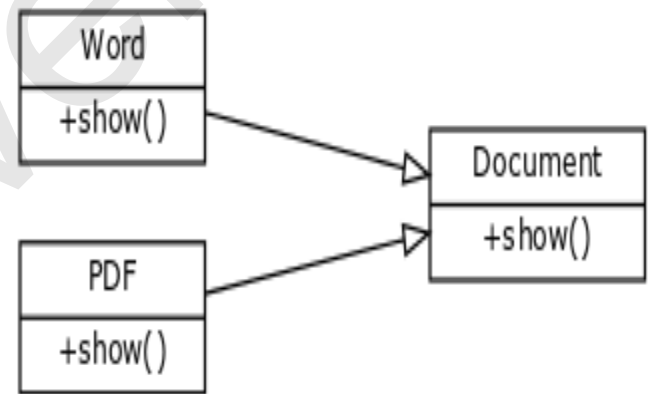
class Circle(Polygon):
    def __init__(self,r):
        self.radius=r
    def findarea(self):
        print(3.14*self.radius*self.radius)

class Triangle(Polygon):
    def __init__(self,b,h):
        self.base=b
        self.height=h
    def findarea(self):
        print(0.5*self.base*self.height)
```

```
c=Circle(2)
c.findarea()
t=Triangle(2,3)
t.findarea()
```

Polymorphism with abstract class

If you create an editor you may not know in advance what type of documents a user will open (pdf format or word format?).



for document in documents:
 print document.name + ': ' + document.show()

Polymorphism

```
class Fruit:
    def __init__(self,name):
        self.name=name
    def Eat(self):
        pass
class Apple(Fruit):
    def Eat(self):
        return(self.name,'Cut and eat!')
class Banana(Fruit):
    def Eat(self):
        return(self.name,'Peel and eat!')
class Orange(Fruit):
    def Eat(self):
        return(self.name,'Peel and eat!')

Fruits = [Apple("apple"),Banana("banana"),Orange("orange")]

for Fruit in Fruits:
    print (Fruit.name , Fruit.Eat())
```


Benefits of classes and inheritance

- the class is sharable, so codes can be reused.
- The productivity of programmers increases
- Data is safe and secure with data abstraction.
- Extensibility