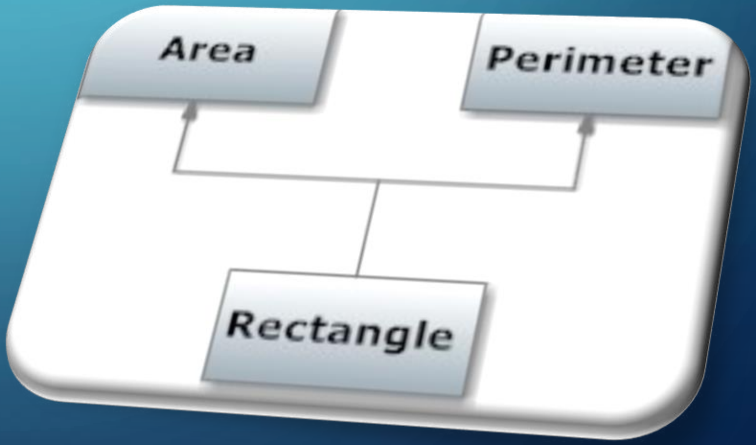
INHERITANCE

CREATING RELATIONSHIPS BETWEEN TYPES

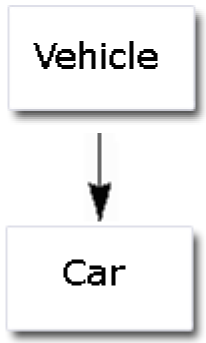


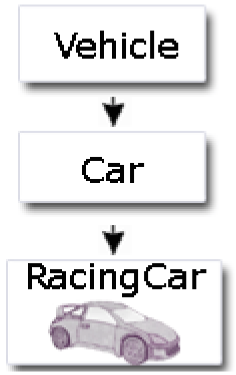
Introduction

- Inheritance defines a relationship among classes where the classes share state or behavior.
- Inheritance means the new class is a more specialized version of the older class.

Definition

- Creating or deriving a new class using another class as a base is called Inheritance.
- The new class is called a Derived class and the class inherited from is called the Base class.

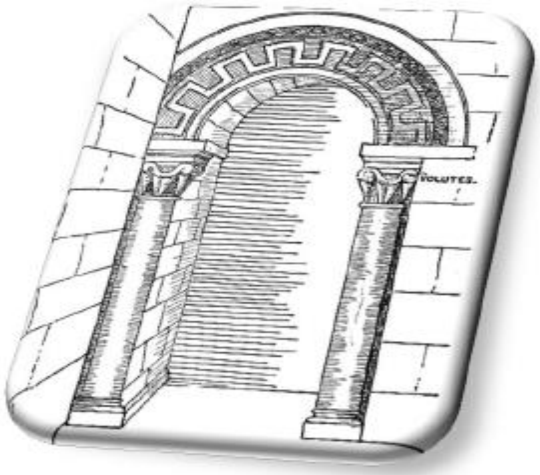




IS-A

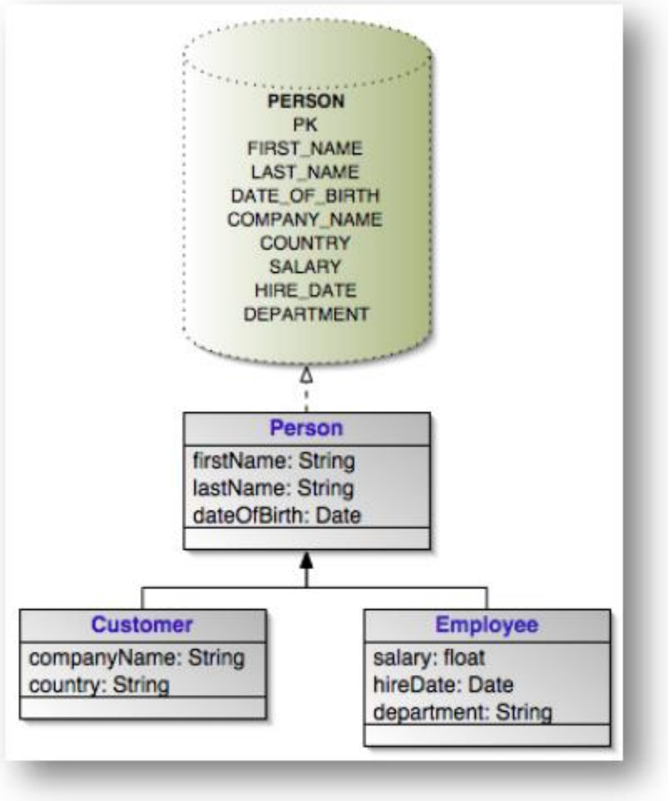
- Given an architecture where Inheritance is appropriate:

A class derived from another class can substituted for the original class object in all cases.



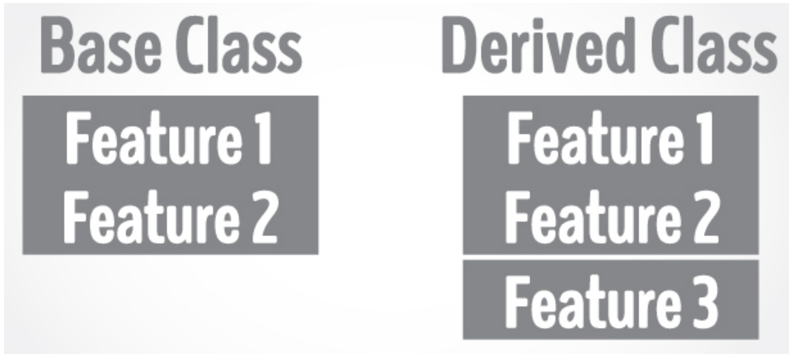
Inheritance Diagram

- Base Class Person
- Customer is-a Person
- Employee is-a Person



Features

- The derived class will inherit all features of the base class.
- Derived classes inherit the attributes and behaviors from their base classes, and can introduce their own.
- Derived classes can also override some of the features (functions) of the base class.

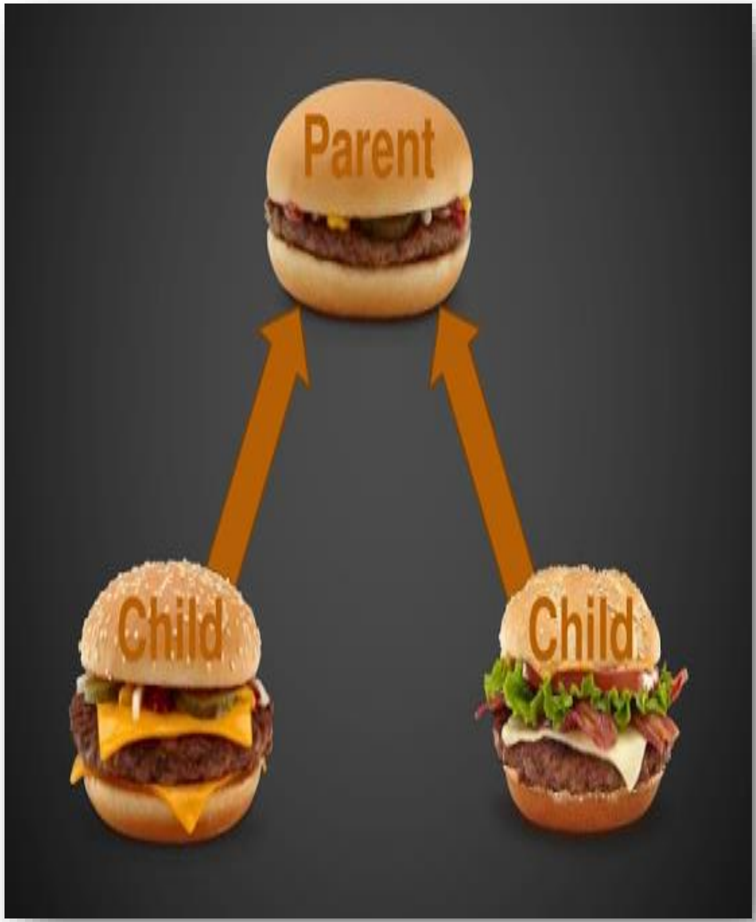


Example

- An Animal class may have some sub-classes called Dog and Cat. The Animal class may provide a member function called “speak”.
- The Dog and Cat will both inherit the Animal’s “speak” function.
- The Dog class will provide it’s implementation of the “speak” function and “bark”.
- The Cat class will provide it’s implementation of the “speak” function and “meow”.

Inheritance Usage

- Inheritance is best applied in cases where a relationship between classes makes sense.
- Inheritance creates a “high coupling” relationship, meaning any change to the base class is likely to affect the derived class.
- In cases where inheritance fits, it is a good solution, but artificial relationships should not forced.

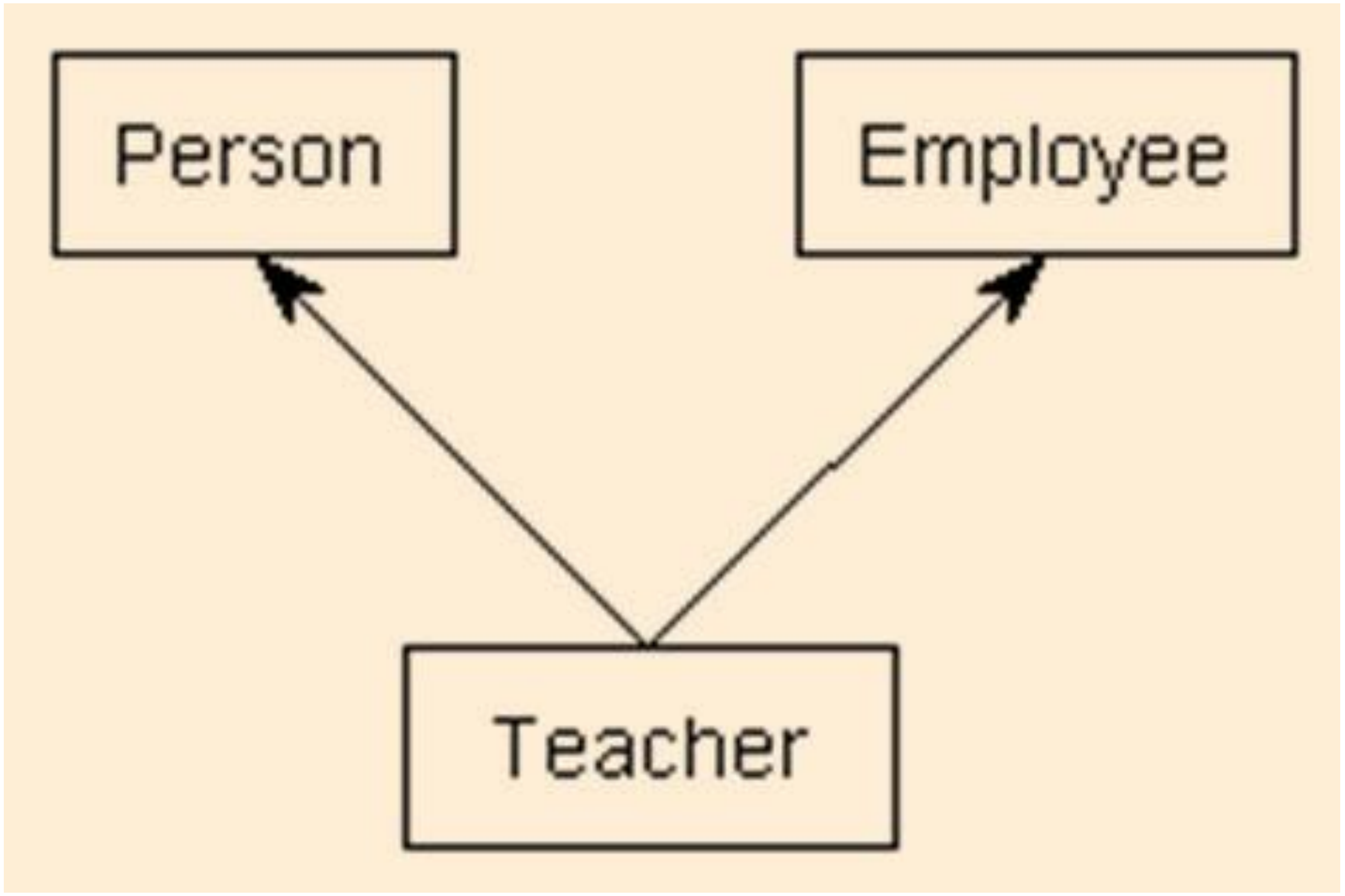


Example

```
class Cheese:
    def __init__(self):
        self.fat = 100
        self.salt = 100

class Bacon:
    def __init__(self):
        self.deadpig = 0.10

class Hamburger:
    def __init__(self):
        self.deadcow = 0.25
        self.bun = 1.0
```

```
class CheeseBurger( Hamburger ):
    def __init__(self):
        self.cheese = Cheese()


class BaconBurger( Hamburger ):
    def __init__(self):
        self.bacon = Bacon()
```

Multiple Inheritance



- A class can be derived from more than one base class in Python, similar to C++. Java doesn’t support multiple inheritance.

- In multiple inheritance, the features of all the base classes are inherited into the derived class.

- Inheritance is the ability to define a new class that is a modified version of an existing class.

Example

```
class A:
        def __init__(self):
                pass
class B(A):
        def __init__(self):
                super().__init__(self)
class C(A,B):
        def __init__(self):
                A.__init__()
                B.__init__()
```