2. OOP/Class Diagram

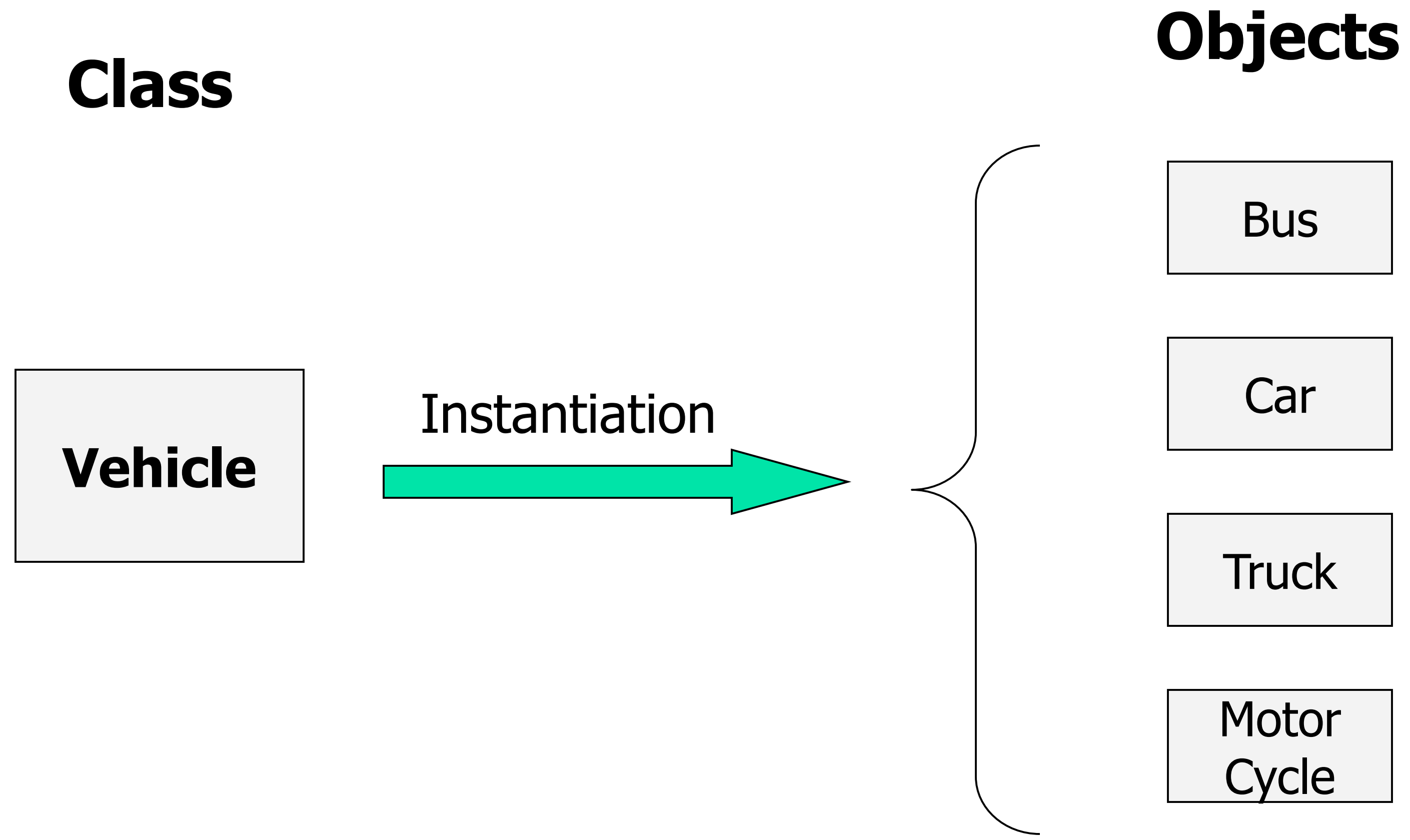
Major Concepts of OOP

1. Class/Objects
2. Abstraction
3. Encapsulation
4. Inheritance
5. Polymorphism

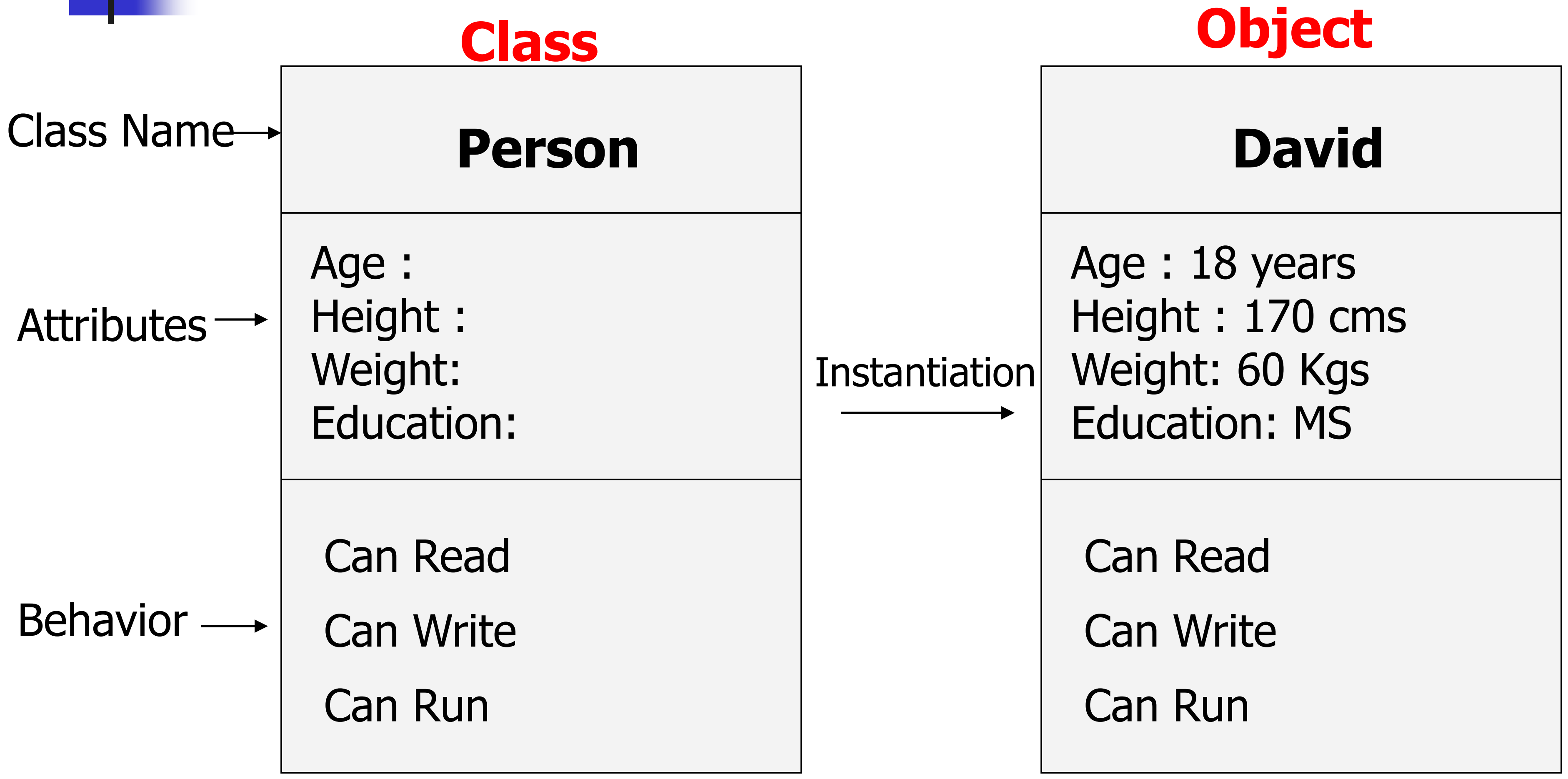
Objects and Classes

- Object is an instance of a Class. That is, every Object has a Class.
- Object has a unique identity. Two objects of a same class are distinguishable.
- A Class describes a group of Objects with similar properties (attributes), common behavior (operation).

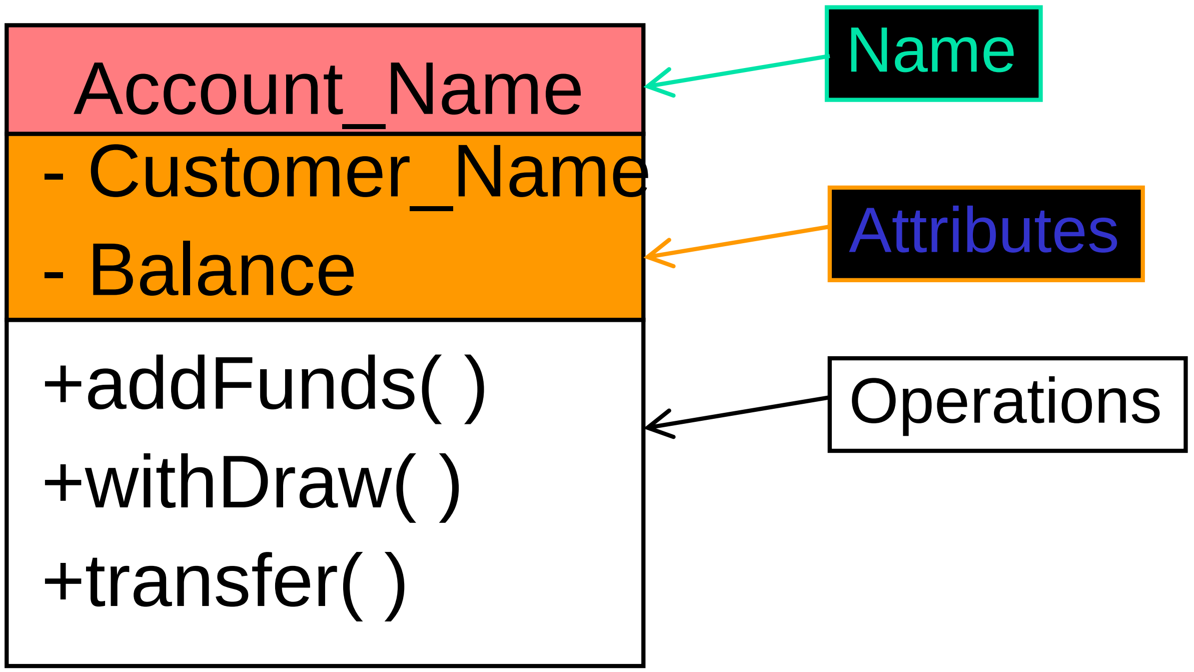
Objects and Classes



A Class of an Object



Class diagram



Class diagram

- A class diagram depicts classes and their interrelationships
- Used for describing structure and behavior in the use cases
- Provide a conceptual model of the system in terms of entities and their relationships
- Used for requirement capture, end-user interaction
- Detailed class diagrams are used for developers

Class diagram

- Each class is represented by a rectangle subdivided into three compartments
  - Name
  - Attributes
  - Operations

- Modifiers are used to indicate visibility of attributes and operations.
  - '+' is used to denote *Public* visibility (everyone)
  - '#' is used to denote *Protected* visibility (friends and derived)
  - '-' is used to denote *Private* visibility (no one)

- By default, attributes are hidden and operations are visible.

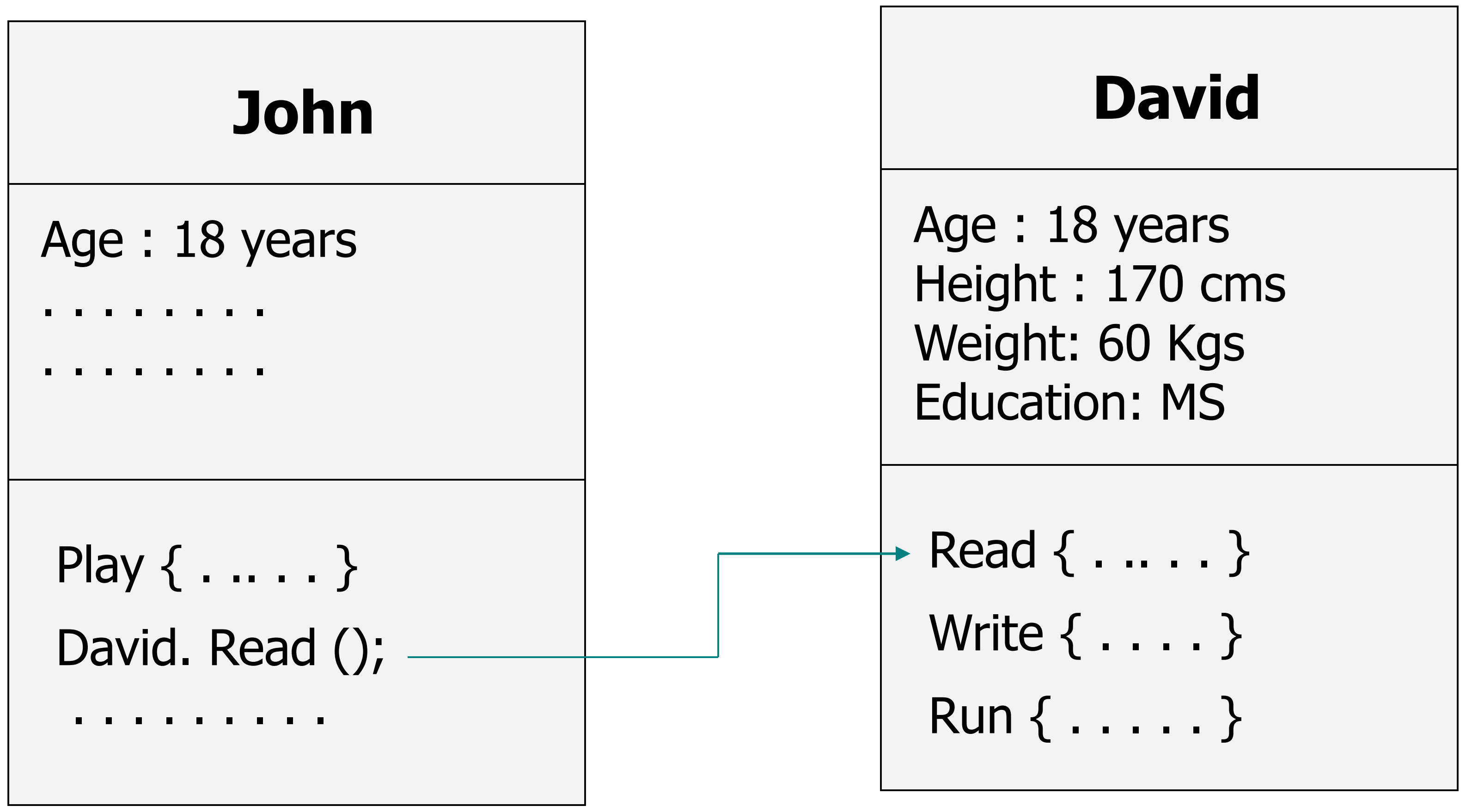
Abstraction

- In OOP, you can abstract the implementation details of a class and present a clean, easy-to-use interface through the class member functions.
- Abstract classes and interfaces are used to hide the internal details and show the functionality. It focuses on ideas rather than events, the user will get to understand of “what” than “how”.

Encapsulation

- Also known as ***Data Hiding***.
- Separating the external aspects of an object, which are accessible to other objects, from the internal implementation details of the object, which are hidden from other objects.
- Since data and behavior are combined in a single entity, this makes encapsulation cleaner and more powerful.

Encapsulation



Abstraction Vs Encapsulation

- In the popular programming text Object-Oriented Analysis and Design, Grady Booch writes that:

*"Abstraction and encapsulation are complementary concepts: **abstraction focuses on the observable behavior of an object...encapsulation focuses on the implementation that gives rise to this behavior**"*

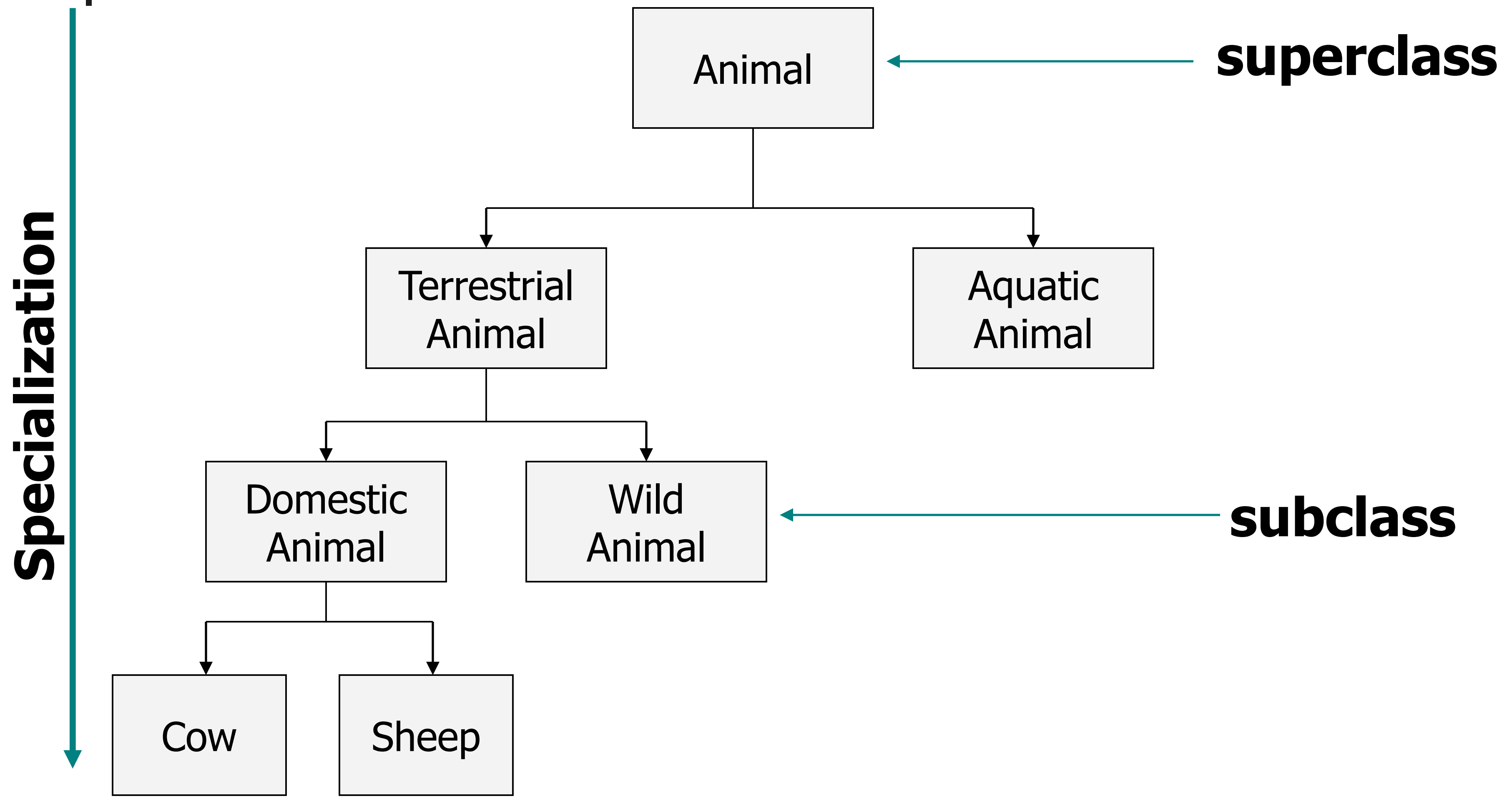
Abstraction Vs Encapsulation

- Stated differently, an abstraction relates to how an object and its behaviors are presented to the user and encapsulation is a methodology that helps create that experience.

Inheritance

- Inheritance is a powerful abstraction for sharing similarities among classes while preserving their differences.
- This is the relationship between a class and one or more refined versions of it.
  - The class being refined is called the ***superclass***
  - Each refined version is called ***subclass***

Inheritance

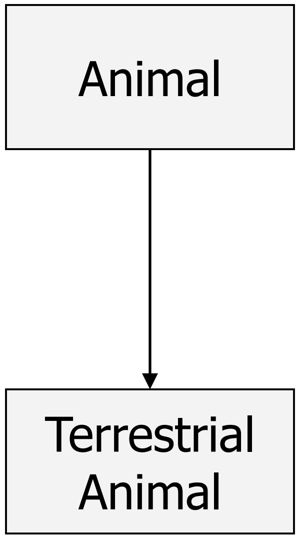


Inheritance

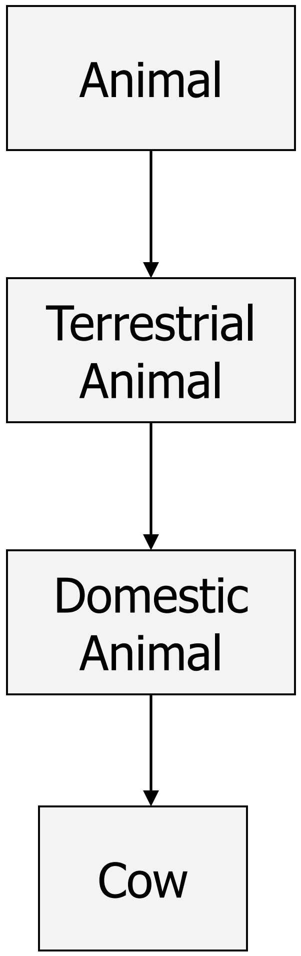
Types of Inheritance:

- Single Inheritance
- Multilevel Inheritance
- Multiple Inheritance

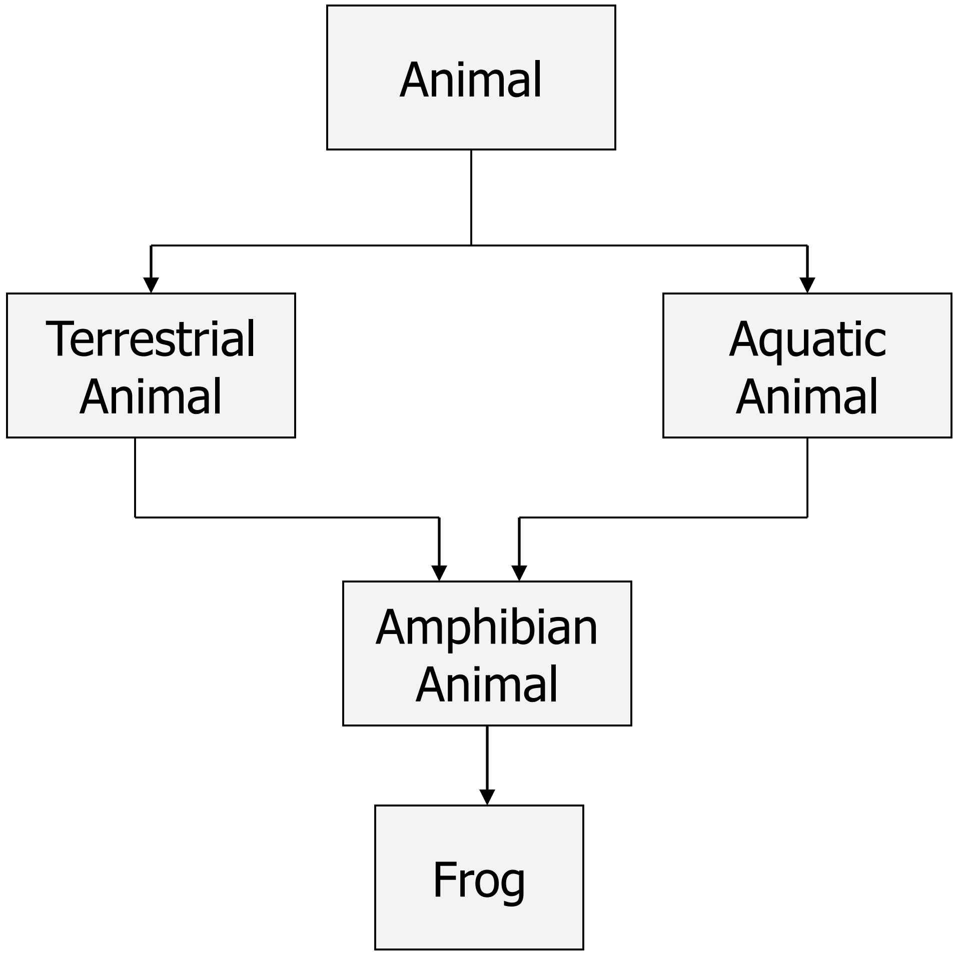
Single Inheritance



Multilevel Inheritance



Multiple Inheritance



Polymorphism

- *Polymorphism* means the ability to take more than one form.
- An operation may exhibit different behaviors in different instances. The behavior depends on the data types used in the operation.
- Polymorphism is extensively used in implementing Inheritance.