Lecture One

General Introduction

1. Programming Paradigm:

Programming paradigms are different styles of programming, like recipes for writing code. Major examples include:

- **Imperative** (step-by-step instructions, like C).
- **Declarative** (describing the result, not the process, as seen in SQL and HTML).
- **Functional** (using pure, stateless functions, evident in Haskell and JavaScript's map function).
- **Object-Oriented** (organizing code around data and behaviors, used in Java, C#, and Python),

The focus of this course is on object-oriented programming.

2. The Object-Oriented Programming:

Object-Oriented Programming (OOP) is a programming paradigm that organizes software design around data, or objects, rather than functions and logic. This approach models real-world entities within a program, representing them as objects with unique *attributes* (data) and *behaviors* (methods).

- **Key Principles of OOP:**
  - ✓ **Encapsulation:** This principle involves bundling data and the methods that operate on that data within a single unit, an object. It hides the internal workings of an object from external access, exposing only a controlled interface.

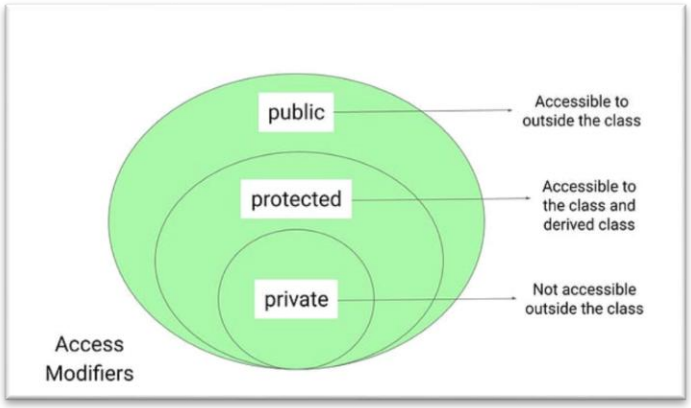

Figure 1. The concept of encapsulation

- ✓ **Inheritance:** Inheritance allows new classes (child classes) to derive properties and behaviors from existing classes (parent classes). This promotes code reuse and establishes a hierarchical relationship between classes.

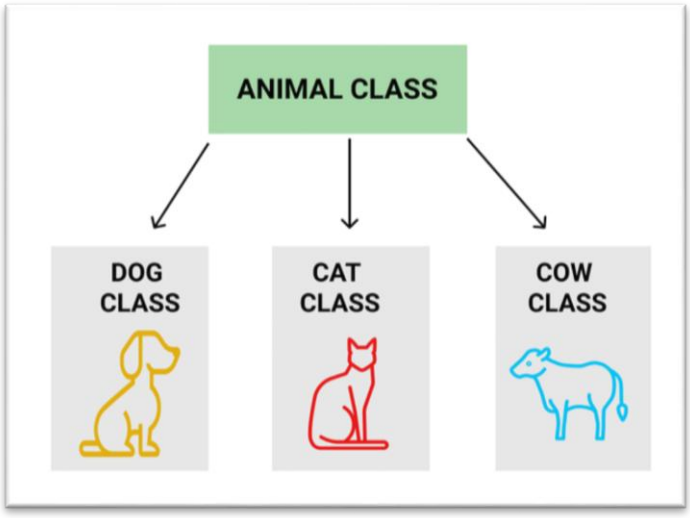

Figure 2. The concept of inheritance

- ✓ **Polymorphism:** Polymorphism enables objects of different classes to be treated as objects of a common type. It allows for a single interface to be used for a general class of actions, with specific implementations varying based on the object's type.

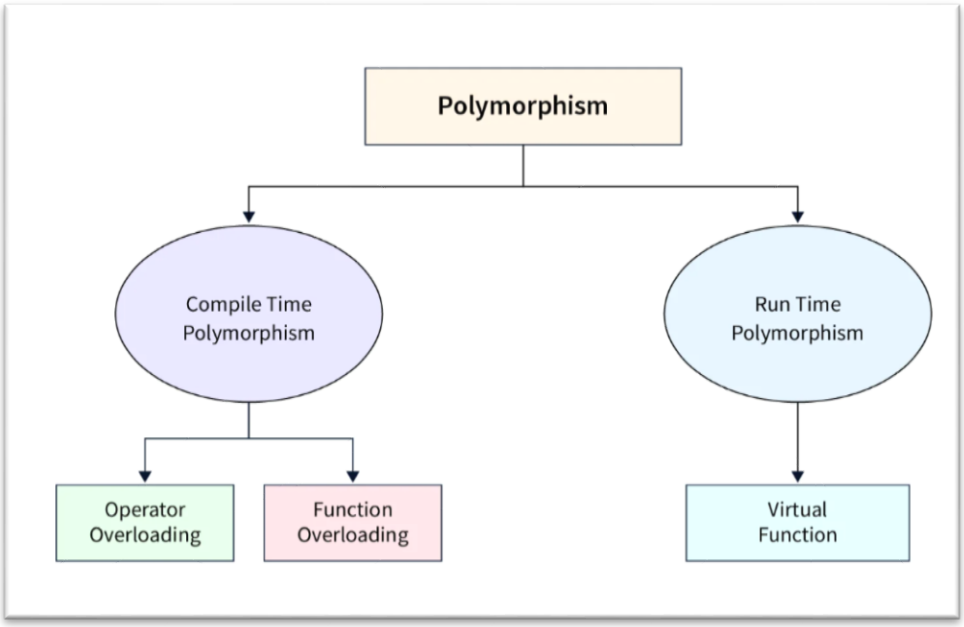

Figure 3. The concept of polymorphism

- ✓ **Abstraction:** Abstraction focuses on exposing only essential information to the user while hiding complex implementation details. It provides a simplified view of an object's functionality.

- **Core Concepts in OOP:**
  - ✓ **Class:** A class acts as a blueprint or template for creating objects. It defines the structure and behavior that objects of that class will possess.
  - ✓ **Object:** An object is an instance of a class, representing a concrete entity with its own specific data values and the ability to perform the actions defined by its class.
  - ✓ **Method:** Methods are functions associated with an object, defining its behaviors and how it interacts with its data.
  - ✓ **Property/Attribute:** Properties or attributes are variables within an object that store its data and define its characteristics.

- **Benefits of OOP**
  - ✓ **Modularity** – Code is organized into classes.
  - ✓ **Reusability** – Classes and methods can be reused in different programs.
  - ✓ **Maintainability** – Easier to modify and fix.
  - ✓ **Scalability** – Suitable for large, complex applications.
  - ✓ **Security** – Encapsulation helps protect data.

Activity

1. **Find another programming paradigm?**
2. **Compare between the (Imperative, Declarative, Functional and OOP) programming paradigms.**