Introduction to C♯ and OOP

C♯ follows the principles of Object-Oriented Programming (OOP), an approach that structures code around objects.

Objects

An object is a fundamental building block representing a real-world entity or concept within a program. Objects encapsulate both data, often referred to as attributes or properties, and behaviors, defined by methods or functions. For instance, in a game, an object could be a player with properties like name, score, and health, and behaviors like moving and attacking. The concept of objects allows developers to model and organize code in a way that mirrors the structure and interactions of the systems they are addressing. Each object is an instance of a class, a blueprint that defines its structure and behavior. The use of objects facilitates code organization, reusability, and the ability to represent complex systems in a more intuitive and manageable manner.

OOP Principles

You will expand on these concepts later, but here is a quick summary:

- **Encapsulation:** Bundling data and methods into a single unit (class) for organized code.
- **Inheritance:** Allows a class to inherit properties and behaviors from another class, promoting code reuse.
- **Polymorphism:** Objects can take multiple forms, enabling a single interface for different data types.
- **Abstraction:** Simplifying complex systems by modeling classes based on essential properties.