**What is inheritance and why is it important?**

Inheritance is a fundamental principle in object-oriented programming that allows a class to acquire properties and methods from another class. The class that provides the properties is called the **base class**, while the class that inherits them is called the **derived class**. This principle is important because it promotes **code reuse**, reduces duplication, and makes programs easier to maintain and expand. Instead of writing the same code multiple times, developers can write common functionality once in the base class and extend it in derived classes.

A benefit of inheritance is **organization and modularity**. For example, in my Mindfulness Program, I created a base class called Activity that contained properties like Name, Description, and Duration, as well as methods for showing start and end messages. Then, I created derived classes such as BreathingActivity, ReflectionActivity, and ListingActivity that inherited these properties and methods, but also implemented their specific behaviors.

Here is an example of inheritance from my program:

// Base class

class Activity

{

public string Name { get; set; }

public string Description { get; set; }

public int Duration { get; set; }

}

// Derived class

class BreathingActivity : Activity

{

public void Run()

{

Console.WriteLine("Breathe in...");

}

}

In this example, BreathingActivity automatically has the properties Name, Description, and Duration from Activity without having to redefine them. This demonstrates how inheritance allows developers to **build specialized classes based on general classes**, making the code more efficient and easier to maintain. By using inheritance, I was able to create multiple types of activities in my program without duplicating code, which is both practical and professional.