Inheritance is a fundamental concept in object-oriented programming that allows a class (called a *child* or *derived class*) to inherit properties and methods from another class (called a *base* or *parent class*). This promotes code reuse, reduces duplication, and makes programs easier to manage and scale.

One major benefit of inheritance is code reusability. By writing shared code in a base class, we can reuse it in multiple derived classes, saving time and reducing errors. Inheritance also supports polymorphism, which allows different child classes to implement their own version of a method, improving flexibility and maintainability.

In my W05 Mindfulness Program written in C#, I used inheritance to define a base class called Activity, which holds shared data like Name and Description, and a method called DisplayStartingMessage(). Then I created specific activity classes like BreathingActivity and ReflectionActivity that inherit from Activity.

Here is a code example demonstrating inheritance in C#:

public class Activity

{

public string Name { get; set; }

public string Description { get; set; }

public void DisplayStartingMessage()

{

Console.WriteLine($"Welcome to the {Name}.");

Console.WriteLine(Description);

Console.WriteLine();

}

}

public class BreathingActivity : Activity

{

public void StartBreathing()

{

Console.WriteLine("Start breathing in... and out...");

}

}

public class ReflectionActivity : Activity

{

public void StartReflection()

{

Console.WriteLine("Think about a time you overcame a challenge.");

}

}

By using inheritance, I was able to define shared behavior once and apply it to multiple activities in my program. This made my code cleaner, more modular, and easier to extend in the future.