**Articulate: Inheritance**

**Definition of inheritance**

Inheritance is one of the key principle in object-oriented programming. A programmer is able to save himself sometimes if he employs this principle. This principle allows programmer to create a base class with some properties and behavior (methods) in which a new class known as the derived class inherits the properties and behaviors (methods) of base class. This mechanism enables the creation of a hierarchical relationship between classes, promoting code reuse and modularity. Inheritance provides a way to build upon existing code, extend functionality, and create more specific classes without rewriting common logic.

**Benefit of Inheritance**

Inheritance is very important to the programmer because code reusability. By inheriting from a base class, a derived class can reuse the methods and properties of the base class, reducing redundancy and making the code easier to maintain. This also promotes a clear and logical structure, making the program more organized and easier to understand.

**Application of Inheritance**

A classical application of inheritance the mindfulness assignment. Instead of writing separate and repetitive code for each activity. Inheritance can be used to create a base class with general attributes and methods, and then create derived classes that extend or specialize the base class.

**Below is an example of code from the mindfulness application:**

**// Base class representing a general mindfulness activity**

**public class MindfulnessActivity**

**{**

**protected int duration;**

**// Method to show a spinner for a specified number of seconds**

**public void ShowSpinner(int seconds)**

**{**

**// Implementation for showing a spinner or countdown**

**Console.WriteLine($"Showing spinner for {seconds} seconds...");**

**}**

**// Abstract method to be implemented by derived classes**

**public virtual void RunActivity()**

**{**

**// Base activity implementation**

**}**

**}**

**// Derived class representing a breathing activity**

**public class BreathingActivity : MindfulnessActivity**

**{**

**// Implementing the specific run method for breathing activity**

**public override void RunActivity()**

**{**

**Console.WriteLine("Starting Breathing Activity...");**

**ShowSpinner(5);**

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

**ShowSpinner(3);**

**Console.WriteLine("Breathe out...");**

**ShowSpinner(3);**

**Console.WriteLine("Breathing Activity Completed!");**

**}**

**}**

**// Derived class representing a reflection activity**

**public class ReflectionActivity : MindfulnessActivity**

**{**

**private List<string> prompts = new List<string>**

**{**

**"Think of a time when you stood up for someone else.",**

**"Think of a time when you did something really difficult.",**

**// Other prompts...**

**};**

**// Implementing the specific run method for reflection activity**

**public override void RunActivity()**

**{**

**Random rnd = new Random();**

**string prompt = prompts[rnd.Next(prompts.Count)];**

**Console.WriteLine($"Reflection Prompt: {prompt}");**

**ShowSpinner(5);**

**// Additional implementation for reflection...**

**Console.WriteLine("Reflection Activity Completed!");**

**}**

**}**

.