# What is Inheritance?

Inheritance in object-oriented programming allows one class to inherit properties and methods from another class. This makes the code more efficient by reducing duplication. The parent class holds the shared behaviours, and the child classes use or modify them as needed. This concept helps with easier code maintenance and organization. For example, in the mindfulness program, you have an “Activity” parent class that includes common functions for activities, like displaying the starting message. The “BreathingActivity”, “ReflectingActivity”, and “ListingActivity” classes inherit those common functions from the parent class, but each adds their own specific functions, applicable to their activity. For example, the “ReflectingActivity” child class has its own methods for displaying questions and prompts to the user. This shows how inheritance helps maintain a clean, flexible design in programming.

Below is an example of how the Activity class has a DisplayStartingMessage() method which is called by the Reflecting Activity class. The child class also has its own methods to get and display prompts to the user.

**Activity.cs**

protected string \_name;

protected string \_description;

protected int \_duration;

public Activity()

{

    \_name = "";

    \_description = "";

    \_duration = 0;

}

public void DisplayStartingMessage()

{

    Console.Clear();

    Console.WriteLine($"Welcome to the {\_name} activity.\n");

    Console.WriteLine($"{\_description}\n");

}

**ReflectingActivity.cs**

protected List<string> \_prompts;

protected List<string> \_usedPrompts;

protected List<string> \_questions;

protected List<string> \_usedQuestions;

protected Random \_random = new Random();

public void Run()

{

    DisplayStartingMessage();

    Console.Write("How long would you like your session to be (in seconds)? ");

    \_duration = int.Parse(Console.ReadLine());

}

private void DisplayPrompt()

{

    string prompt = GetRandomPrompt();

    Console.WriteLine("\nConsider the following prompt:\n");

    Console.WriteLine($"--- {prompt} ---\n");

}

private string GetRandomPrompt()

{

    if (\_prompts.Count == 0)

    {

        ResetPrompts();

    }

    int index = \_random.Next(\_prompts.Count);

    string prompt = \_prompts[index];

    \_usedPrompts.Add(prompt);

    \_prompts.RemoveAt(index);

    return prompt;

}