The principle of “Inheritance” in programming with classes is used to allow derived classes to “inherit” the properties of the base class. This means that classes that have tasks in common can have a base class that takes care of those tasks, leaving the class to focus only on the tasks that are unique to them. This provides the ability for each class to do more without repeating code and for changes to be made only once. It is important to only use inheritance for situations where the derived classes are truly a more specific version of the base class, not just classes that share common tasks. In the mindfulness project there were multiple types of activities that each did specific things, but they all had common traits as activities. The different types of activities, Breathing, Listing, Reflecting, were all “activities’, so it made sense to have an Activity class to handle common tasks. The following code shows that the ReflectingActivity class is able to use the attributes and methods of the Activity class directly by declaring itself a child of the Activity class (in this example specifically, DisplayStartingMessage and ShowSpinner).

public class ReflectingActivity : Activity

public void Run()

    {

        DisplayStartingMessage();

        Console.Clear();

        Console.WriteLine("Get ready... ");

        ShowSpinner(6);

        Console.WriteLine("Consider the following prompt:");

        Console.WriteLine();

        string prompt = GetRandomPrompt();

        DisplayPrompt(prompt);