**In coding, inheritance is analogous to a family tree for classes. It occurs when one class (the kid) may inherit characteristics from another (the parent). If we Consider the code as if it was a family in which children inherit specific characteristics from their parents, such as the color of their eyes. Classes can also inherit characteristics and behaviors from other classes.**

One of the highlights about inheritance is that it helps me structure our source code and saves time. We can design a basic class (parent) with common traits and then create separate groups (children) that are comparable but have their own characteristics. This way, we don't have to rewrite everything that we did in one part of the code every time.

There is a basic class named *Mindfulness Activity* in our mindfulness software that contains shared data as well as methods. Then there are unique classifications that derive from *Mindfulness Activity*, for instance, *Breathing Activity*. They add their own unique features while using the same aspects. It's similar to constructing with LEGOs in that you start with a large box of common bricks (base class) and then use them to create unique constructions (child classes).

using System;

using System.Threading;

class MindfulnessActivity

{

protected string name;

protected string description;

public MindfulnessActivity(string name, string description)

{

this.name = name;

this.description = description;

}

public virtual void Start(int duration)

{

Console.WriteLine($"Starting {name} activity. {description}");

Console.WriteLine($"Duration: {duration} seconds.");

Console.WriteLine("Get ready to begin...\n");

Thread.Sleep(3000);

}

public void End()

{

Console.WriteLine("Good job! You've completed the activity.");

Console.WriteLine($"Activity: {name}");

Console.WriteLine("Duration: seconds");

Thread.Sleep(3000);

}

}