The term "polymorphism" simply refers to the ability to treat various things as though they are members of the same type even when they are not. It enables us to create versatile code that can interact with a variety of things. So, based on the application I developed this week, we were asked to developed a game that had multiple goal types, including simple, eternal, and checklist goals. Each goal has unique characteristics and behavior. I can make a list of goals and treat them all as though they are just "goals," even though they are of different types, thanks to polymorphism. This makes it possible to construct new classes that share traits with the parent class while still having their own special properties and methods by creating classes that inherit from the parent class. This makes it simpler to maintain the code and can speed up the process of changing or adding new features.

An application of polymorphism onto the program I developed this week is the use of the `Display` method in the `Goal` class. This method is overridden in the child classes `EternalGoal` and `CheckListGoal` to display different information. For example, the `Display` method in the `CheckListGoal` class displays the list of items that need to be completed for the goal, while the `Display` method in the `EternalGoal` class displays a message indicating that the goal can never be completed. When the `View` method of the `Game` class is called, it iterates through the `\_goals` list and calls the `Display` method for each goal. This allows the method to display the correct information for each goal, depending on its type.

Another instance is where the `AddGoal()` method takes a `Goal` object as a parameter, but it can also accept any object that is derived from `Goal`. This allows the method to add any type of goal object to the `\_goals` list. Because each of the derived goal classes implement the same interface as `Goal`, the `AddGoal()` method can treat all of these objects as if they were of type `Goal`, making it more generic and flexible.

private void AddGoal(Goal goal)

{

\_goals.Add(goal);

Console.WriteLine("Goal created!");

Console.Write("Going back to Main Menu");

System.Threading.Thread.Sleep(3000);

Console.Clear();

}