Polymorphism is like inheritance but includes a bit more. All subclasses of a base class share the same methods and properties as the base class. Because of this, they are all counted as a base class in the code. In addition, using abstract and virtual methods, subclasses can call the same method of the base, however they execute their own unique methods.

    public virtual void DisplayGoalFull()

    {

        char state = ' ';

        if (goalCompleted)

        {

            state = 'X';

        }

        Console.WriteLine($"[{state}] {goalName} - {goalPoints} | {goalDescription}");

    }

This is an example of a virtual method. Every subclass shares this method. However, the subclasses may have an override method that uses their method instead.

    public override void DisplayGoalFull()

    {

        Console.WriteLine($"[{currentProgress}/{totalProgress}] {base.GetName()} - {base.GetPoints()} + {bonusPoints} | {base.GetDescription()}");

This override method will display when the program calls the method of the base class using the subclass. In addition, an abstract method requires all subclasses to have their own override method in their own class. This works great as it allows subclasses to do their own things while still being able to be apart of the base class.