[0555] Object Oriented Design via Noun-Verb Analysis

Just like Structured Programming came with its own technique for writing programs (i.e., top-down design), Object-Oriented programming has its own technique called "Noun/Verb Analysis."

One of the best ways to design an Object Oriented program is to write down, in English, what the program needs to do. Be as specific as possible but limit yourself to 3 or 4 paragraphs. Here is an example:

Create a dice rolling simulation. Each die will have 6 sides with number 1 thru 6 on each side. When rolled, each die displays a random value from 1 to 6. The dice are rolled using a cup. The cup can contain up to 6 dice.

The next step in the design process is to identify all of the nouns in the description. These nouns have the potential for becoming classes or properties in our program. For our example, the nouns are:

- Dice
- Simulation
- Die
- Sides
- Number
- Value
- Cup

Next, we eliminate nouns that are duplicates or outside the scope of our program. In our example, "Dice", "Simulation", "Sides" and "Number" can be removed. "Dice" is just the plural of "Die." "Simulation" is another name for our program itself (i.e. our Main() method). "Sides" don't need to be modeled since all we care about is the value of each die after it is rolled. And "Number" is just another name for Value. That leaves us with:

- Die
- Value
- Cup

Next, decide which nouns are classes and which are properties. In this case, the "Cup" will contain an array (or Collection) of several "Die" objects, each of with have a "Value" property.

Finally, go back and find all the active verbs in the description. Those will potentially become methods in your classes. Our example has the following active verbs:

- Create
- Roll (from rolled) - for both a Die and the Cup

- Display
- Contain

"Create" is what our program does when it runs - Create is not a method (it's a "new" statement). "Display" just means using Console.WriteLine(). "Contains" means stuff is stored as a member variable inside of bigger stuff.

That leave us with "Roll" which applies to all the Dice in a cup as well as the Cup itself.

So, based on this noun-verb analysis, we need a Die class and a Cup class. Each Die has a member field called "value" and a method called "roll." The Cup class contains a member variable that is an array of Die objects. It also has a "roll" method that calls the roll method for each die object. Finally, both Die and Cup overload the ToString() method in order to return the values of each die as a string which allows the Main() method to then display it.

Whew! At this point, we should be able to code things up pretty quickly using this design.

Like This! **[0556] DiceSimulation.cs (https://lwtech.instructure.com/courses/1841516/pages/0556-dicesimulation-dot-cs)**

To review, here are the steps for performing Noun-Verb Analysis:

1. Write down in English what the program needs to do
2. Identify and write down all of the Nouns in your description
3. Remove nouns that are duplicates or outside of the scope of the program
4. Decide which nouns are Classes and which are properties
5. Go back to the English description and pick out all of the Verbs
6. Eliminate verbs which your Main() method will do and verbs that describe standard C# capabilities
7. Match up the remaining verbs with your classes. The verbs will probably be methods in those classes.