PREPARING FOR PROGRAMMING ASSIGNMENT #3

Noun-Verb Analysis

Use it. Live it. Love it. Learn it. Practice it.

[0555] Object Oriented Design via Noun-Verb Analysis (https://lwtech.instructure.com/courses/1841516/pages/0555-object-oriented-design-via-noun-verb-analysis)

Write down the assignment in your own words based on your understanding of the assignment's description. Circle the nouns, underline the verbs. Decide if each noun is a class, a property of a class, or a distraction. Name your classes, enums, and properties accordingly (single nouns). Decide if each verb is a method or something that your Main() method does. Add stub methods to your classes accordingly. Flesh out each class with appropriate constructors, methods and ToString()'s. Finally, add code to Main() to use your new classes to accomplish the assignment. Easy!

Recipe for a POJO

Use it. Live it. Love it. Learn it. Practice it. **(https://lwtech.instructure.com/courses/1841516/pages/0580-extra-recipe-for-a-pojo-class)**

[0580] EXTRA: Recipe for a POJO (Class) (https://lwtech.instructure.com/courses/1841516/pages/0580-extra-recipe-for-a-pojo-class)

I expect to see ALL of the items mentioned in the POJO recipe in your POJO classes - even if your Main() method doesn't need them (yet).

Modeling Playing Cards

A playing card has a "suit" (Hearts, Diamonds, Clubs or Spades) and a "rank" (Ace, Two, Three, ...). How can those things be represented in C#?

Shuffling Cards

It turns out that shuffling cards correctly so that true randomness occurs throughout the deck is harder than you might think. Spend some time researching card-shuffling algorithms on the Internet. You'll find one that has a hyphenated name. Try to implement it in your program. Be sure to include a reference/link to the website you used as a comment next to your card shuffling code.

Do It Right! You'll be Reusing Your Classes!

It's important that you complete this assignment correctly because we will be doing more with these classes later in the course. Make sure you are confident you have great, easy-to-use, easy-to-reuse, easy-to-maintain, bullet-proof card classes. In other words, spend time making those classes as general purpose as possible. Think about adding methods that might not be needed for this assignment, but might be needed in a typical card game program.