OOP Design

Solving a problem using Object Oriented Programming Involves:

- Requirement gathering: Meaning, collecting user stories
- Identify nouns to make up your class in the problem domain
- Identify verbs to go with each noun which make up the behaviors (methods) of that class
- Identify adjectives to go with each noun which make up the state (variables) for that class
- Analyze classes and eliminate overlap of states and behaviors by organizing their hierarchies.
- Promote or demote a class in the inheritance chain as necessary
- Create interfaces to dictate certain behaviors to be implemented in the classes
- Organize classes into composition (has-a relationship) or inheritance chain (is a relationship) and send messages to each other to solve user stories
- Implement encapsulation while writing classes by restricting access to data within the class.
- Use visibility modifiers in methods for accessibility to clients (of class)
- Pass data to/from static methods as parameters
- Create helper methods in some utility class as needed

Example 1: An example of OOP in action is a functioning house (think of your own home where you grew up). You should try to design your OOP based on the above procedure and see how an OOP modeling would work in your house.

Example 2: Another example could be a system, which is capable of drawing a 2D box in the console output window. Let us solve this with OOP approach:

You model real world problems into a solvable software solution:

- Take one simple user stories.
- Identify **nouns** to make up your class
- Verb make up *behavior* and adj. <u>*state*</u>
- Organize classes to solve user stories

Step 1 (User story): **John** *loves* **boxes**. **He** *draws* <u>big</u> **boxes** and <u>small</u> **boxes**, <u>white</u> **boxes** and <u>black</u> **boxes**, <u>simple</u> and <u>fancy</u> **boxes**.

Notice how I have **bolded** the nouns, *italicized* the behavior, and <u>*underlined the italicized*</u> word for state?

Step 2: Group these nouns, verb and the adverb together. Go through little iteration and eliminate redundancy (singular, plural of same noun, same meaning etc.). What come out are distinct nouns, verb and the adverb.

Nouns: John (client), He, Box
Verb: Draws, loves
Adjective: Big, small, black, white, simple, fancy

Step 3: Now, put analytical hat and observe what they really mean, and how could you combine them.

Really, there are only two nouns, a Client class that uses a Box class to draw. Client sends a draw message to draw different types of boxes. So, big small, black, white, simple, fancy are part of the Box's states. You can represent big, small using X, Y co-ordinates, color state will store all colors, and a shape type will make it fancy or simple.

Now, the behavior of the Box can be dictated by a Interface Drawable which has only one method drawYourself() which is used by the client to ask the box to draw.

Note: This process is overly simplified. When you go through OO design and analysis course you would do in-depth analysis and be able to make a good design. Since this is not an OOA & D course, I will help you get through the design aspect.