- **Basic Principle of OO**
  - Abstraction. A model that includes most important aspects of a given problem while ignoring less important details. Enable designers to describe their solution in more generic ways (easily understandable and promoting reusability)
  - Encapsulation. Hiding implementation details (the key is an interface). This allows or enables an easily modular design/development and maintenance.
  - Modularity. The breaking up of something complex into manageable pieces
  - Hierarchy. Level of abstractions. Right amount of information at each level, promote reusability
Design/UML notation/Diagram

- Fundamental DS such as Data Type, Data Structure, and ADT and give illustrative examples of each.
• Array data structure. Both simple and complex data elements with some operational analysis –
  • Inserting (adding a new item)
  • Searching
  • Deleting
  • Complexity analysis - Big O

• Simple sorting algorithms (bubble, selection, insertion)
  • how they achieve the sorting
  • pros and cons (operational)
  • complexity analysis
• ADT (stack and queue)
• Using ADT to solve problems (e.g. parsing, given an expression -> transformation and evaluation)
• Linked List
  – comparison with array
  – pros and cons
• varieties of linked list (simple, double-ended, doubled)
• Linked list operations (how to and complexity analysis)