~ Information Hiding: Information hiding involves concealing the internal details of a module (such as data structures, algorithms, and internal logic) from other modules. It ensures that only necessary information (such as a module's interface) is exposed to the rest of the system.
~ Module independence refers to the degree to which a module can operate and evolve without being affected by, or affecting, other modules. It implies low coupling (reduced interdependence) and high cohesion (focused, self-contained functionality) within a module.

CSE6214 Software Engineering Fundamentals

Tutorial 5

Part A: Discussion
Topic (Lecture 5): Design Concepts

---

1. Do you design software when you "write" a program? What makes software design different from coding?

2. How do we assess the quality of a software design?

3. What benefits can be obtained from designing based on the following concepts:
   - Abstraction - understand overall by essential point, instead of small details (clarity)
   - Software Architecture - blueprint: clear structure for system development, identify potential risks early in design process
   - Separation of Concerns - improve management & simplifying troubleshooting
   - Modularity
   - Information Hiding - protect system by restrict information from unauthorized access
   - Functional Independence - independent function can be tested individually, ensuring better quality control
   - Refinement -

4. Discuss the relationship between the concept of information hiding as an attribute of effective modularity and the concept of module independence.
   data that needs to be hide can be isolate into one module so that if there's any changes, it doesn't affect the other parts of system (flexibility to change any data inside it)
5. Does "refactoring" mean that you modify the entire design iteratively? If not, what does it mean? no but to have better quality in the system itself

Part B: Project

---

6. For each use case assigned to you, elaborate the process into a sequence of interaction between the user and the system, for example (make appointment use case):

   a. Student enters date, time and lecturer name to make appointment.
   b. System creates new appointment record in Appointment table.
   c. System displays confirmation message to the student.

7. Based on the interactions, use Visual Paradigm to draw a sequence diagram for the use case. Refer to the following webpage on how to draw sequence diagrams:

http://www.visual-paradigm.com/support/documents/vpuserguide/94/2577/7025_creatingsequ.html