UNIT 5 REVIEW

http://www.flickr.com/photos/digitalcolony/2725932075/sizes/o/

Key concepts in this unit

- Coupling
  - Content, common, control, stamp, data
- Cohesion
  - Functional/informational, communicational, procedural, temporal, logical, coincidental
- Law of Demeter, moving code, sandwiching, composition vs inheritance, interfaces, incremental and iterative development
- Design patterns
  - Builder, adapter, façade, memento, interpreter, observer, template method, factory method, strategy, decorator, composite, visitor

Check yourself

- Instances of class X modify members of a data structure that is located inside of class Y. Which of the following is true?

A. This is an example of content coupling

B. This is an example of control coupling

C. This is an example where there is no coupling

D. This is an example where there is no cohesion

Check yourself

- Instances of class X modify members of a data structure that is located inside of class Y. Which of the following is true?

A. This is a violation of the Law of Demeter

B. Fixing this problem calls for sandwiching

C. Fixing this problem calls for inheritance

D. There is no problem here – nothing to worry about.

Check yourself

- Instances of class X modify members of a data structure that is located inside of class Y. Which of the following is true?
  - A. This problem can be safely ignored if the only development planned is to insert some component Z between X and Y.
  - B. This problem can be safely ignored if the only development planned is to make some component Z send data to X.
  - C. This problem can be safely ignored if the only development planned is to improve X and Y later.

Check yourself

- Instances of class X modify members of a data structure that is located inside of class Y. Now another version of Y must be created (call it W). So X needs to use Y and W. But of course you want to minimize coupling. Which step below can be omitted?
  A. Step 1: Create an interface (call it V).
  B. Step 2: Make W and Y both implement V.
  C. Step 3: Modify X so it references W and Y via V.
  D. Step 4: Violate the Law of Demeter so W invokes Y.

Check yourself

- Why is it useful to know design patterns?
  A. Design patterns are fancy, and software engineers like fancy stuff.
  B. Design patterns are architectural styles that make software more efficient.
  C. Design patterns include valuable code that can be reused.
  D. Design patterns are ideas for how to organize code in response to common problems.

Check yourself

- Suppose your program has to support replication. You need a way for the program to save its state so the program can be copied to other servers. Which would you use?
  A. Visitor
  B. Factory method
  C. Memento
  D. Façade

Check yourself

- Your program generates various outputs. You need a way to notify users when certain outputs are generated. Which would you use?
  A. Strategy
  B. Observer
  C. Interpreter
  D. Façade

Check yourself

- Your program needs to iterate over an existing data structure (such as a tree or a graph) and take some operation on each item in the data structure. Which would you use?
  A. Builder
  B. Template method
  C. Visitor
  D. Façade