Homework 11, CPSC-4175

Chapter 05, Object-Oriented and Classical Software Engineering

October 23, 2017

1. Give an informal description or definition of *stepwise refinement*. Pretend that you are trying to explain it to your grandmother or a younger brother or sister.

2. Consider *topdown developnent, stepwise refinement*. We have now encountered both of these. Traditionally, software engineering was taught developing programs using the topdown, stepwise refinement paradigm. (OOAD has replaced this paradigm.) You have probably never developed a program using that paradigm, but think about the implications of that technique versus OOAD. Discuss several of the major differences you can think of.

3. Read the article in my PDF directory on Github by Nicklaus Wirth entitled *Program Development by Stepwise Refinement* and write a one paragraph appreciation of that article.

4. In business school *cost benefit analysis* goes by the nae *due diligence*. Following the example in the book, give a non-technical example of the exercise of due diligence. ("Non-technical" in the sense of a business or management decision, for example, opening a branch office or expanding a product line.)

5. Your book gives the following illustration: "suppose that object A contains an invocation of a method of object B. In this situation, object A cannot be reused without reusing object B as well." Please discuss this problem and state a *specific* solution to the problem that advances separation of concerns.

6. Answer one of the following two questions. Use the analytical tools you have read about in this chapter.

   (a) Design a small program and write the pseudo-code that generates documentation from a source file. Input is a source listing. Output is a formatted documentation document. You documentation should document all variables by giving the visibility, type, name, and initial value, and all functions giving the visibility, type, name, and input parameters. You are allowed to require your source listings to use a particular formatting, similar to Javadoc.

   (b) Design a small module and write the pseudo-code that generates four levels of debugging information while a program is running. You should import, use, or require the module in your program. Your program should reference the debugging module and initialize a configuration parameter giving the debugging level. There is no input. Output should be a debugging log. Your output should include the name of the program and the date on which it was tested. In your program, you would invoke the debugging module by calling the appropriate debugging functions.

7. One very common problem, particularly with languages similar to C, is that many files must be compiled in a specific order because of (among other things) dependencies among files. For example, `program.c` may depend on `program.h`. Make files are almost universally used. Search for a brief makefile example, and comment on it line by line.