CSSE 304 Day 36 Summary

1. Recap and Summary of yesterday's material on imperative form. Code is in the slides.

2. Engines (built-in in *Chez* Scheme, can be added to any Scheme implementation, based on `call/cc`) are a way of simulating multi-tasking in a language that does not have it built-in. For the exam, you should understand the main concepts and terminology, and be able to read and understand simple code that uses them, and (on the written part of the exam) write VERY simple code that is based on the code from class.

3. `(make-engine thunk)` creates an engine that will, when applied to three arguments, activate a timer-interrupt mechanism and then evaluate the body of `thunk`. A thunk is simply a procedure that takes no arguments, similar to what we pass as the producer to `call-with-values`.

4. The three arguments passed to an engine are:
    a. **ticks:** a positive integer specifying the amount of "fuel" given to the engine.
    b. **complete:** a procedure that specifies what to do if the evaluation of thunk finishes before the fuel expires.
        i. complete is a procedure of two arguments: amount of fuel "left over" and the result of the computation.
    c. **expire:** a one-argument procedure to be executed if the computation runs out of fuel before it completes.
        i. The argument that will be passed to expire is a new engine that can finish the computation from where it left off.
5. `engine-fib` example. Code is on the slides; this space is for your notes

6. `round-robin` example. Code is on the slides; this space is for your notes

7. `amoeba` example. Code is on the slides; this space is for your notes