RB109 Assessment Study Guide Notes

Be able to explain clearly the following topics:

- **Local variable scope, especially how local variables interact with method invocations with blocks and method definitions**
    - A variable's scope determines where in a program a variable is available for use. A variable's scope is defined by where the variable is initialized or created. Variable scope is defined by a *block*. A block is a piece of code following a method invocation, usually delimited by either curly braces {} or do/end.
        1. Inner scope can access variables initialized in an outer scope, but not vice versa.
        2. Outer scope variables can be accessed by inner scope.
        3. Inner scope variables cannot be accessed in outer scope.
        4. Peer scopes do not conflict
        5. Nested blocks create nested scopes
        6. Variable shadowing prevents access to the outer scope local variable. It also prevents us from making changes to the outer scoped variable.
    - How local variables interact with method definitions:

Method definitions can be thought of as setting a certain scope for any local variables in terms of the parameters that the method definition has, what it does with those parameters, and also how it interacts(if at all) with a block. Method definitions *cannot* directly access local variables initialized outside of the method definition, nor can local variables initialized outside of the method definition be reassigned from within it.

- How local variables interact with method invocations with blocks: **Method invocations** can be thought of as using the scopes set by the method definition. If the method definition is defined to use a block, then the scope of the block can provide additional flexibility in terms of how the method invocation interacts with its surroundings. A block can* access local variables initialized outside of the block and can reassin those variables.