**SSW 555 Agile Methods for Software Development**

**Quiz 6: Refactoring**

**Name: Mrunal Phadke**

-------------------------------------------------------------------------------------------------------------------------------

**1. What is refactoring?**

* Refactoring is the process of changing a software in such a way that it does not alter the external behavior of the code yet improves its internal code.
* It is a disciplined way to clean up code that minimizes the chances of introducing bugs.
* That is when you refactor you are improving the design of the code after it has been written.
* Thus, with refactoring you can take a bad design, chaos even, and rework it into well-designed code.

**2. Describe the 2 Hats of Software Development.**

* 2 hats of software development are: Adding function and refactoring.
* When you add function, you shouldn't be changing existing code; you are just adding new capabilities. You can measure your progress by adding tests and getting the tests to work.
* When you refactor, you make a point of not adding function; you only restructure the code. You don't add any tests. You only restructure the code.

**3. What is technical debt? How is it paid off?**

* Refactoring is needed to pay off "technical debt": Technical debt is incurred when the design of software evolves without making consistent refactorings, which may include many bugs and unstructured code.
* The debt can be paid off by refactoring.
* With technical debt come interest payments, that is, the extra cost of maintenance and extension caused by overly complex code. You can bear some interest payments, but if the payments become too great, they can become way too much.
* It is important to manage your debt, paying parts of it off by means of refactoring the code.
* Instead of increasing debts and further continuing the development or work, it can be that done later.

**4. Describe 2 other reasons (besides technical debt) for refactoring code.**

* Refactoring Makes Software Easier to Understand.

Refactoring helps you to make your code more readable. When refactoring you have code that works but is not ideally structured. The time spent refactoring can make the code better communicate its purpose.

* Refactoring Helps You Find Bugs.

By clearly defining structure of the program, you get clear with purpose of the code, and while doing it you may spot the bugs you make while redoing it.

**5. Describe 2 bad smells in code that suggest you should refactor.**

* **Duplicated code**

If you find the same code segments in more than one place you can just write them in a function and reuse the function in the place of the code.

The simplest duplicated code problem is when you have the same expression in two methods of the same class. Then all you have to do is Extract method and invoke the method from both places.

* **Large Class**

When a class is trying to do too much, it can have too many instance variables. In this case you can have an Extract Class to bundle a number of the variables.

You can group the variables in some component. For example, "getMovieId" and "getMovieName" are likely to belong together in a component.