***Lab 3 – Maintenance***

Date Assigned: Thursday, August 29

Date due: **Thursday August 29**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

* Review/Understand what Maintenance Projects are about; and
* Review/Understand refactoring.

**Instructions:**

1. Download a copy of the Lab 4 file from the Moodle page.
2. Save this document as a Word document named YourUserName\_K30\_L03\_Maintenance.docxThe document will hold your answers for your lab.
3. Research from class notes and resources, and online resources to answer the questions.
4. Write your answers to the lab questions in the appropriate locations in this file and be sure to save.
5. When you are ready to have your lab marked, notify the professor.
6. When you are finished, submit to Moodle.

To do:

**Use our class notes, discussions, and online resources to answer the following questions.**

1. For each of the following, include 2 or 3 sentences to explain the type of maintenance and include an example. Make sure you put it in your own words, i.e. don’t just cut and paste.
   1. What is corrective maintenance? Give an example.
      1. Corrective maintenance is fixing something that isn’t working properly. For example a spelling error in some text
   2. What is perfective maintenance? Give an example.
      1. Perfective maintenance is taking something that works and making it better. For example making a function more efficient
   3. What is adaptive maintenance? Give an example.
      1. Adaptive maintenance is when something in the architecture of your project changes and you have to adapt. For example a framework no longer has LTS so you change frameworks
   4. What is preventative maintenance? Give an example.
      1. Preventative maintenance is when you know something is going to change or go wrong so you take measures to prevent it. For example you know that a new dataset will break certain functions so you change the function beforehand
2. Name and explain 2 challenges associated with each of the types of maintenance.
   1. Corrective maintenance:
      1. Reproducing a problem
      2. Fixing without introducing new bugs
   2. Perfective maintenance:
      1. Scaling
      2. Managing user expectations
   3. Adaptive maintenance:
      1. User does not see direct changes
      2. Design may not adapt well
   4. Preventative maintenance:
      1. Hard to get approved
      2. Requires careful regression testing
3. What is refactoring? Give 3 examples of changes that can be considered refactoring.
   1. Refactoring is taking something that already exists and remaking it to make it better
      1. Removing duplication
      2. Making the code more modular
      3. Making code less complex
4. Explain why testing is so important for successful refactoring. How would the absence of existing test assets affect the effort required?
   1. Testing and having tests already in place because ideally the code already works so when you refactor you want to make sure that the code still works
5. Explain the term technical debt. How can it lead to the need to refactor code?
   1. Technical debt is when a task takes longer to do because of code being too complex / bad. This leads to needing to refactor code because for example if you have 5 hours of technical debt per sprint then it is heavily impeding your efficiency
6. Refer to the following link: [Software maintenance - Wikipedia](https://en.wikipedia.org/wiki/Software_maintenance). Explain Maintenance debt. How does obsolescence factor into maintenance debt? Is it possible for Maintenance debt to make it impossible to support a product?
   1. Maintenance debt is accumulation of issues that come from not addressing technical debt and maintenance tasks.
   2. Not doing updates to obsolete things could lead to compatibility issues
   3. yes
7. Use the website <http://jamesshore.com/Agile-Book/refactoring.html> to answer the following questions.
   1. In a couple of sentences describe, in your own words, *reflective design*.
      1. Reflective design takes the understanding the design of existing code and using refactoring and code smells to improve on that design
   2. Describe 4 examples of *code smells* that the author describes.
      1. Divergent change and shotgun surgery
         1. Unrelated changes affect the same class or multiple classes need to be changed for a single idea
      2. Primitive obsession and data clumps
         1. Using primitive data types such as decimal and having several primitive types represent one idea
      3. Data class and wannabee static class
         1. Having data and code in separate classes and classes that contain methods but nothing meaningful to that specific object
      4. Coddling nulls
         1. Methods that check for null references and then return null themselves
8. Read [Coding conventions - Wikipedia](https://en.wikipedia.org/wiki/Coding_conventions) and answer the following questions.
9. What are Code Conventions?
   1. Code conventions are a set of guidelines that coders should follow so that code is much more legible and understandable when a project is being worked on by another developer
10. Why are they important for software maintenance?
    1. They are important because you want other programmers to be able to understand your code quickly when it comes to maintenance or bug fixes. Not following coding conventions could also lead to technical debt

1. What does LDAP stand for? How is it used?
   1. Light weight directory access protocol
   2. It makes it possible for applications to query user information rapidly
2. Explain what an IIS AppPool is and describe 2 benefits.
   1. IIS application pool is a collection of one or more websites running a single w3wp process with a single identity
      1. Benefits
         1. Encapsulation among hosted web apps
         2. Apps running outside a given pool cannot affect the applications in the pool

Mark Breakdown:

|  |  |
| --- | --- |
| **Question** | **Mark** |
| 1. Types of Maintenance | 8 |
| 1. Challenges for each | 8 |
| 1. Refactoring, examples | 4 |
| 1. Testing and refactoring | 4 |
| 1. Technical debt | 4 |
| 1. Maintenance debt | 4 |
| 1. Reflective design and code smells | 6 |
| 1. Coding conventions | 4 |
| 1. LDAP | 2 |
| 10. IIS AppPool | 2 |
| Organization, English, handed in properly | 1 |
| Total | 47 |

**To submit**

When you have completed the lab save the file and upload the following to Moodle: YourUserName\_K30\_L03\_Maintenance.docx.