**Discussion Topics: Pragmatic Programming**

In this module's discussion board assignment, answer the following questions:

1. Using the ***Pragmatic Programmer*** reading assignment, select one (1) topic and complete the following:
   * Why did you select this topic?
   * Summarize the main points (in your own words) of that topic in three or four sentences.
   * Find at least one additional resource (video, book, article, website, etc.) that supports your summary. Include a link to that resources.

After reading the topics for this module, Topic 37: Listen to Your Lizard Brain, stuck out to me in particular. I cannot begin to count how often my body goes into autopilot. Specific tasks in my daily life are so repetitive that muscle memory drives my actions. This is so true that sometimes, when I take a moment to think through something normal for me to do, I forget how to do it because, most times, I just automatically accomplish it.

As we continue to code, we strengthen our natural instincts. Without realizing it, we are thinking through code and acting on it. These reactions can be indescribable since we are often unaware that they are happening, and we are letting our lizard brains take over (Thomas & Hunt, 2024/2020). When something does not feel right with our code, we must listen to this instinctual warning (Thomas & Hunt, 2024/2020, p. 141). If you are feeling stuck on identifying the issue your brain has detected, sometimes it is best to remove yourself from the environment, giving yourself a break or discussing it without someone else before diving back in (Thomas & Hunt, 2024/2020).

An article that I found that supports this summary is by Mahmoud Al-Qudsi in his 2013 article on NeoSmart Technologies. Although he does not refer to it as his lizard brain, his article focuses on why programmers should trust their instincts. We can save time by listening to our instincts and finding solutions to tricky code (Mahmoud Al-Qudsi, 2008). Even if there are standard practices, as a programmer, if your mind feels this is the wrong direction, listen to it (Mahmoud Al-Qudsi, 2008).

**References**

Mahmoud Al-Qudsi. (2008, May 22). *Programmers Should Trust Their Instincts*. The NeoSmart Files. https://neosmart.net/blog/programmers-should-trust-their-instincts/

Thomas, D., & Hunt, A. (2020). *The Pragmatic Programmer: your journey to mastery*. Addison-Wesley. (Original work published 2024)

***Before you submit your thread, put your name in the subject line.***

**Assignment Requirements and Grading:**

1. An initial post of approximately 250 words is due by **Thursday, 11:59 p.m., CT**.
2. For the initial post to be considered substantive, it should be at least 250 words in length and fully cover the topics being presented. Single-sentence definitions or responses will not be awarded points.
3. Submit your post by clicking on the **Assignment Link** above, then **Create Thread**. You must create a thread in order to view your peers' posts. Tip: Create your post in a Word document and then copy and paste your work into the thread.
4. A minimum of three (3) responses, **to the original threads of other students**, of 100-200 words each are due by **Sunday, 11:59 p.m., CT**.
5. To view the rubric grading criteria, click on the following link: [Discussion Board Grading Rubric](https://content.bellevue.edu/cst/csd/rubricdbv3.pdf).

**(50 points)**

Jessica, you did a great job summarizing topic 37! I also chose to focus on the same topic. You are correct that we must trust our gut when it comes to coding! There is a reason why we feel uneasy with our code or plans, even if we do not understand it yet. It is important to mention that if you are struggling to figure out what could be going wrong but feel that gut feeling, you should take a step back from your code or outline. Sometimes, taking a step back can help clear your mind and make detecting the error(s) easier when you return to the project.

Joe, I thought your summary of the main points of the Specification Trap was very thoughtful and thorough. I felt similar to you in that it seemed strange that Thomas and Hunt would warn against writing overly detailed specifications in programming. After reading that topics section, I grew to understand what they meant. When you invest more time at the beginning than is needed, you are likely wasting the extra time since programs easily change course. I really enjoyed reading through the Lenovo article you included! I will say that I think the article did kind of go against our authors’ points. It seems to favor more details.

Brian, I enjoyed reading your summary of topic 38. I related to your feeling of not always understanding why a code is not working. It can be very frustrating when everything seems like it is run correctly, but it is not. Even when a syntax error is raised, when you are so used to staring at the line of code, it is easy to skip over the issue. You are spot on when you say understanding code can help build better habits as a programmer. Finding a solution does not count for much if we do not understand why that is the solution. Plus, the fix might be a bandage that could easily fall off if we only temporarily fixed the problem.