**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 resource.

I selected Topic 27: Don’t Outrun Your Headlights for this week’s discussion. Often, creating a new program can feel somewhat intimidating at first; this is why this section stuck out to me. I am admittedly a planner, so not being able to see around the corner is something that I hate to face.

Throughout programming, if we are not careful, we can sabotage ourselves by going faster or further than we should. Our authors compare this to outrunning our headlights since low-beam lights are not long-range, giving us a limited view ahead of us (Thomas & Hunt, 2024/2020, p. 99). Taking the process step by step is helpful when programming. During each step, ensure that everything works correctly and is purposeful (Thomas & Hunt, 2024/2020, p. 99). To help this, gather feedback from others, error codes, unit tests, and trial runs by reading, evaluating, and more. It is essential to distinguish what we know versus what guesses we use to predict future outcomes. There will be circumstances where we cannot avoid speculation since client expectations must be set, but it is best to break it down as much as possible. Sometimes, we encounter black swans, which throw us off our rails due to their unpredictability. We can find ways to advance and improve our ability to see further. In the topic, the authors may consider this similar to turning on the high beams or brights to expand the change.

Black Swan theory dates back to 1697 but has been revised to fit today’s use and understanding (GeeksforGeeks, 2024). The theory goes beyond just coding, reaching other aspects of life (GeeksforGeeks, 2024). Black Swan events can have a negative impact on stakeholders.

**References**

GeeksforGeeks. (2024, January 8). *Black Swan Theory: History, Example, Benefits & Effects*. GeeksforGeeks. https://www.geeksforgeeks.org/black-swan-theory/

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)**

Cindy, I think you did an excellent job on your post for this week’s module! You did a really good job of summarizing the topic and explaining it further. Decoupling is smart when integrated into the code. It makes sense since the heavy connection between the components would mean adjustments must be made to both. When errors occur, altering code for multiple entities is more time-consuming. I am glad you mentioned that it can help with the scale because you are correct. It is interesting to note that when referring to other subjects, decoupling is not always a good thing, but it is in coding.

Brett, you did a very nice job addressing your topic. I also focused on the same topic for this week. Similar to you, I was also drawn in by the analogy used to open the topic because I was able to picture the scene clearly. I am a pretty visual learner, so by painting that paint, I am more likely to remember the concept when completing other assignments, while programming, and in general. I think it can be easy to get swept away in the complexities of coding, but taking baby steps can help set ourselves up for a successful outcome.

Joe, you did a good job on your post, and it is interesting to read more about why metaprogramming stuck out to you. I like how you were able to connect it to what our futures as programmers will look like. By utilizing metaprogramming, we can build off of existing programs, altering them in ways that can improve performance. I am intrigued by trying out Enterprise Java Beans since I have not worked with Java at all thus far in my programming journey. Do you have any experience with Java or JavaScript? I like how you sought out an additional resource to help provide your clarity.