**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.

After completing the assigned reading in the *Pragmatic Programmer,* I focused on Topic 17: Shell Games.No matter the application or program, I love to customize the appearance to fit my preferences. This is part of why I selected this topic since our book mentions how much time will be spent using the command line.

Creating a personalized workstation has many benefits. It is similar to adding pictures to a work desk or buying computer accessories or items that improve your workflow. We experience a lack of personalization when we are temporarily separated from our familiar work setting. Often, I find myself surprised to remember my settings and layout are not the default when working at a different location. There are also options to change the prompt store commands under “simple aliases” and “command completion” (Thomas & Hunt, 2024/2020, p. 68). This all aids in constructing an environment that fuels progress. This will make you more comfortable in your setting, making it easier to learn and complete projects.

I was actually surprised that I never really thought or considered that there would be an option to customize my terminal as well! I had already explored this within my IDE VSCode, but it never occurred to me that I could do the same with my terminal. Especially not the idea that the prompt line could be changed! Of course, it does make sense, but for some reason, since it looks so standard, it was hard to imagine it any other way.

In his 2023 article, Mohammed Omer discusses personalizing your command terminal. Specifically, macOS can improve productivity and provide step-by-step instructions on alternatives to your own with plug-ins and more (Omer, 2024).

**References**

Omer, M. (2024). *How to make your boring macOS terminal look so much better*. Medium.com. https://medium.com/@mdomer19967/how-to-make-your-boring-macos-terminal-look-so-much-better-dd7f80ffeedc

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 the following link: [Discussion Board Grading Rubric](https://content.bellevue.edu/cst/csd/rubricdbv3.pdf).

**(50 points)**

Samir, I can relate to your student debugging experience and how solving problems and improving code is challenging and rewarding. So often, I feel defeated when small details within my code cause errors, causing the code not to execute correctly. However, I am thrilled to watch my program successfully run once I can overcome these hurdles. I have felt overwhelmed before when receiving multiple syntax errors, so I agree that isolating the problem makes it more bearable since you handle it one error at a time. You mentioned not to be afraid to look for outside resources if you need help, and I agree! Do you have any resources you recommend when you are feeling stuck?

Miles, you did an excellent job summarizing topic 16! I always find it hard to choose which topic to discuss each week since they all stand out for different reasons. I really like how you were able to connect this topic back to reversibility. You are right that plain text remains the same even as programming continues to grow and adapt. Plain text provides insights that supersede the changes with time, making it easier for any developer to understand what the original coder intended. It was interesting learning more from NinjaOne in the article that you found. Knowing that plain text is versatile is beneficial and is further reasoning for utilizing it in programs.

Adrian, nice job on your post! I also ended up choosing shell games to discuss in my post this week. After implementing these changes earlier this week, doing my later assignments for this week had an entirely different and better feel to it! Automating and prompting the terminal is a game changer. Aside from simple aliases and command completion, the terminal can also be customized appearance wise to fit everyone’s tastes. I really enjoyed reading more about the resources that you include! I definitely think I will be playing within my terminal even more to continue to find ways to improve it to my liking!