AI vs. Human Code: The Bulldog Experiment

Author:
Michael Yattaw

February 11, 2025

When working on the Bulldog program without AI, I had to plan out the structure of the main class piece by piece, coming up with code and methods to organize actions inputted by the user. However, BulldogAI generated all of the code within the main method instead of creating extra methods. Additionally, the AI used numbers to select players rather than string inputs, and it introduced a strange feature where the default player was set to "RandomPlayer" if the input for selecting players was invalid. Interestingly, the AI didn't seem to make any mistakes in designing the program. For example, I overlooked printing the current round scores after each round, but the AI implemented this correctly without needing a specific prompt from me.

Overall, the DeepSeek LLM followed a waterfall process model, successfully referring to the documentation and producing a properly functioning codebase on the first attempt. Debugging was minimal, if there at all, making the process for BulldogAI much shorter. The time it took to complete these program assignments was vastly different. For Bulldog without AI, I spent around 2-3 hours total coding and debugging. However, for BulldogAI, I spent around 30 minutes prompting and debugging the code, and amazingly, it worked almost instantly, making the process about four times faster and much simpler.

Both Bulldog and BulldogAI had their strengths and weaknesses. The biggest weakness I faced with Bulldog was not fully following the documentation, whereas this was not an issue for BulldogAI. However, BulldogAI simplified the main class code to the point where significant refactoring would be needed if we wanted to expand the Bulldog project further. In terms of user experience, both programs performed well, aside from the minor bug in my version where round scores were not printed. However, from a developer's perspective, I am convinced that my version is more maintainable because the AI's approach lacked structure and did not use a constant for the round score. This meant that if someone needed to debug the program, they would have to search through the code for the value "104" and modify it manually.

One major takeaway from this assignment is the importance of carefully reading all the instructions multiple times when working on a programming project, regardless of how simple it seems. I overlooked basic documentation requirements, such as header comments and round score output, simply because they were easy to implement, which led me to forget them entirely. However, I found it fascinating that the AI model was able to correctly write a program, meet all the requirements, and keep the code as simple as possible. While I personally like having extra functions for creating users and beginning the game, they were not necessarily essential for this project, and the LLM seems to understand this, selecting for a simplified but perfectly working alternative. Lastly, if I were to redo this assignment with the expectation of continuing development, I would have prompted the AI to create a more flexible codebase for my own benefit.