Python Screening Task 2: Write a Prompt for an AI Debugging Assistant

Prompt:

"Imagine you are a professional Python programming tutor. I will provide written student Python code that has errors or could be improved. Your role is to look at the code and identify possible problems, but you cannot provide a correct solution. Instead, you will provide useful suggestions or advice that will help the student find the problems without you giving the answer. You need to be professional and make your remarks clear enough that the student will understand what type of error or inefficiency is involved without seeing the answer. Consider the various types of common errors in Python: improper use of functions or libraries, overall errors in the logic of the code, or errors in syntax in the code. For example, if the student has included a syntax error that has to do with parentheses or indentation, you might suggest the student think about what they might have left out. If the logic is wrong, you might suggest they think about their algorithm or the order of their code."

My Design Choices:

I designed the prompt to be simple and kind, as in a friendly tutor or teacher style. My overall goal was to establish clear rules from the outset, so that it is even more clear to the AI what it is supposed to do. I gave the AI a clear identity so it could stay in a friendly and supportive tone right from the start.

A very key aspect of my design is that we never give you the answer. I implemented this as a key design rule by saying the AI is unable to "identify any possible problems without providing any solution options". This design rule focuses the AI on guiding the student to find their own solution, which is what good tutoring is all about! I also asked the AI to focus on the "main mistake or misunderstanding", which is to encourage the AI to point out a conceptual issue rather than a known programming error or triviality. Finally, I was sure to instruct a positive ending to the prompt. This is important as it provides valuable confidence boosts to the student and makes them feel inclined to practice more.

My Reasoning:

1) What kind of voice should the AI use?

The AI should sound like a super-duper, friendly and caring peer or a relaxing, but effective, tutor you would actually want to chill with. It should be nice and clear, and conversational; at

no point should it make you feel bad for messing up - just keep it friendly and casual as you review the variables. There really shouldn't be a lot of technical jargon either. Instead of saying something like, "You've made a scoping error," its message should say, "Hey, check out that variable you created inside the function. When the function ends, what happens to that variable?" That response is useful!

2) How should the AI balance finding bugs and helping the student?

I see the AI as a resource for debugging. The AI doesn't just fix the issue for you, it should give you insights. This AI should show you where the problem might be and allow you to figure it out. Rather than saying, "You have a bug on line 7," it should say, "Take a look at line 7 again, your adding something to your list, but what kind of data is that variable holding?" It offers a hint without giving you the answer.

3) How would you change this for beginners versus advanced students?

For beginner prompts, I would simplify even further. I would have the AI just use very simple words and break down the issues into smaller parts. I would have the AI just go with the very basics.

For advanced prompts, I would alter the prompt to emphasize that the person writing the prompt knows the basics already. I would have the AI focus on more advanced concepts like how to make the code run faster or to design it better. At that point, all the code needs is some tweaks to make it truly awesome, not fix a bug.