The Wrong Comparison Operator

Format: Common App Personal Essay | Level: College Application | ~650 words

The error had no business being there. I had reviewed the code three times, and every time I read it, the logic looked correct. The program was supposed to convert a list of student scores into letter grades and output a report, and instead it was silently dropping every score below 60 and reporting a perfect class average. It took me two days and a rubber duck, an actual rubber duck propped against my monitor, to realize I had written the wrong comparison operator.

The rubber duck is a real debugging technique. You explain your code, line by line, to an inanimate object. The theory is that articulating your assumptions forces you to notice when they are wrong. The duck cannot respond, which is the point; you are not looking for advice, you are looking for the moment when you hear yourself say something that cannot possibly be true.

I had written score > 60 when I meant score >= 60. Every student with exactly a 60 was being classified as failing. In my class data, that was three students, and their absence from the passing category shifted the average enough to eliminate the apparent failures entirely. The program was not outputting errors because it was not making an error by its own logic. It was doing exactly what I told it to do. I had told it the wrong thing.

The two days I spent looking for that bug were not wasted. I learned more about my own habits of reasoning in those two days than I had in the preceding semester of coursework. I learned that I read code the way I meant to write it, not the way I actually wrote it. I learned that when something is almost working, the instinct to keep reading the same section of code is almost always wrong; the error is usually somewhere else, in an assumption so obvious that I never thought to test it.

I also learned something about the relationship between precision and intention. In ordinary language, greater than and greater than or equal to are nearly synonymous; context makes the distinction clear. In code, they are categorically different, and no amount of context compensates for the wrong symbol. The program does not infer what you meant. It does what you said.

I have thought about that bug many times since, not because the mistake was significant in any objective sense but because the process of finding it taught me something I keep applying in contexts that have nothing to do with programming. When an argument is not working, I now look first at the assumption I have not examined rather than the conclusion I have. When a plan is failing, I ask what I told myself was obviously true without checking. The rubber duck on my desk is a reminder that the most important debugging tool is the willingness to explain your logic to something that cannot politely agree with you.