One of the biggest goals in programming is making code that’s readable and concise. The reason programming techniques like procedural programming and object-oriented programming are so useful is because they make it easier to achieve this goal. Despite being such amazing techniques, it is important to know how and when to use them. If used incorrectly, both procedural programming and OOP can actually make your code less readable and concise than the non-procedural variant.

Program 1 tackles the problem of printing the two lines “Hello, Virtual World!” and “It is a great day for programming” using non-procedural programming. It creates the class, then the main method, and then just runs the two System.out.println methods. As it is, it meets both of the goals mentioned earlier. It’s readable and concise as the program just consists of the bare necessities of a java program along with the two println statements.

Program 2 attempts to rewrite Program 1 using procedural programming. Even though Program 2 replaces code in the main method using a new method, it uses procedural programming inappropriately resulting in the code being less readable and concise. The main issue with the code is that the programmer creates a function with very little reusability. Good procedural programming is where you take code that is repeated often with slight tweaks and turn it into a method with parameters to account for the slight tweaks. By doing this, programmers are able to make their code more readable as methods can explain what they do by their name, and more concise as there isn’t all that repeated code anymore. Program 2 creates a method that can only be used once in the program, which makes the program longer and harder to read as you have to first read the method call and then go up to read the method itself.

Program 3 takes the mistake Program 2 made even further. It attempts to apply OOP to this simple program which ends up make the code even less readable and concise. First, the programmer makes the same mistake of creating a method that can only be used once. But after that, the programmer continues to create an object of the class he’s currently in, just so he can call the method that he already has access to. When another programmer reads this code, he’ll see the creation of a new HelloWorldV3 object, look for the HelloWorldV3 class, read the line that runs the printTwoLines function from the new object, and then read the printTwoLines function itself. Program 3 is a horrible example of OOP that unnecessarily creates objects and creates functions that can only be used once resulting in a program that is long and confusing.

Altogether, these 3 programs show that although advanced techniques like procedural programming and OOP are useful, when used inappropriately they can seriously hurt your code. In the end, Program 1, which used basic non-procedural programming, was the most readable and concise and would be the program written by a professional programmer to accomplish this task. On the other hand, Programs 2 and 3 which used more advanced techniques ended up being long and confusing.