From this article, I learned that functional programming is different from iterative programming in that it is highly reusable, treats functions as values, and is more readable. This programming style encourages creating functions for each action of a larger task. They can be reused over again in different parts of your code or in different projects. Additionally, you can pass and return functions as values (since functions are first-class objects). In the article, getItem returned a function, which was later passed into the map function with the input 'temperatures.' AddNumbers was another example of a function being used as a parameter. Being able to return unevaluated functions is useful for generalizing code, similar to making templates. Lastly, with functional programming, the code is more readable in the end. In this article's example, all the procedures were wrapped and consolidated into a single line of functional code in the end!