Functional Programming with JavaScript

Student's Name

Department, Institutional Affiliation

Course Number and Name

Professor's Name

Due Date

Functional Programming with JavaScript

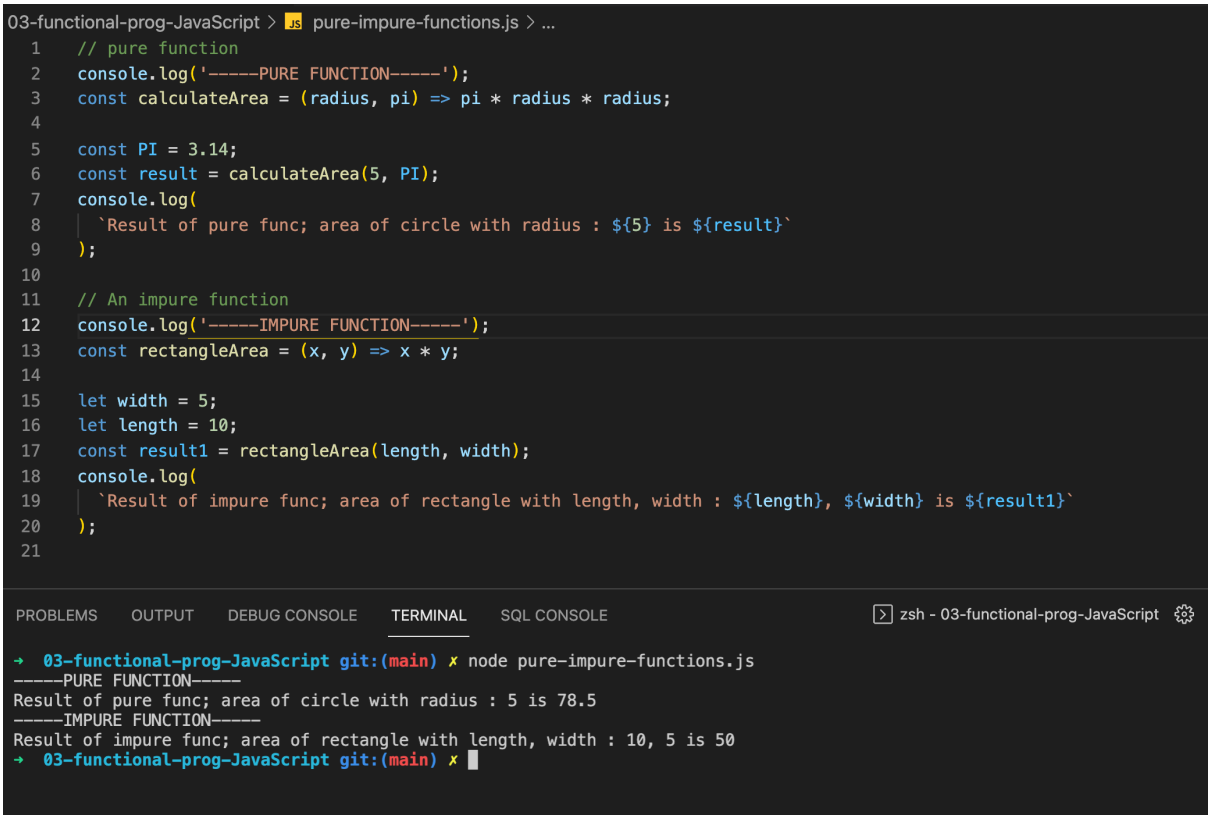
JavaScript is a multi-paradigm language. It allows the use of imperative or procedural, object-oriented, and functional programming simultaneously (Maldonado, 2021). Functional programming (FP) is a paradigm that enforces the use of expressions and functions without mutating the state or data, and functions must remain true and pure to their expression (Recio, 2019). This paradigm has a few fundamental concepts: first-class objects, pure functions, higher-order functions, composition, and immutability (Maldonado, 2021). It also ensures that the code written is error-free, easily readable, maintainable, and testable. This paper focuses on pure functions.

Understanding functional programming requires understanding its fundamental concepts. If given the same arguments, a pure function returns the same result and does not cause any observable side effects (Banz, 2017). An impure function has side effects, such as writing to a file or console and making input/output reads (Banz, 2017). Figure 1 shows an example of pure and impure functions in JavaScript and their outputs. *calculateArea* is a pure function, since it does not make use of any global variable; hence, it will always return the same value given the same arguments. *rectangleArea* is an impure function, since it makes use of global parameters, *width,* and *length*. Therefore, the return value is not guaranteed to be the same for every call, since changing a global variable does not notify the calling method. Return values from pure functions can be memoized (cached). Since they have no external dependencies, pure functions are easier to read, debug, and test. Consequently, programmers should utilize pure functions, if possible.

The functional programming paradigm helps developers write bug-free code. It allows them to write mocking-free tests. Therefore, if a problem can be solved with FP, programmers should prioritize using it.

Figure 1

Pure and Impure Functions in JavaScript



References

Banz, M. (2017, June 27). *An introduction to functional programming in JavaScript.* Opensource.com. https://opensource.com/article/17/6/functional-javascript

Maldonado, L. (2021, July 16). *Functional programming with JavaScript.* Progress Telerik. https://www.telerik.com/blogs/functional-programming-javascript

Recio, S. (2019, August 21). *Functional programming in JavaScript: How and why.* Bits and Pieces. https://blog.bitsrc.io/functional-programming-in-javascript-how-and-why-94e7a97343b