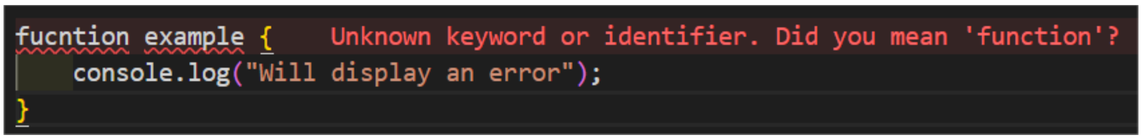
Student: Jorge Alvarez Grana (#249179033)
Class: WDD330
Term: Fall 2021
Subject: W05 Readings

Ch 10: Testing and Debugging

I've learned that there are "errors", "warnings" and "exceptions". All are indications that something is wrong in the code, but they are of different degree of importance. A warning indicates that something is wrong but won't crash the program. An exception is "an error that produces a return value that can then be used by the program to deal with the error." But an error is when something goes wrong in the program there are different three different types of errors:

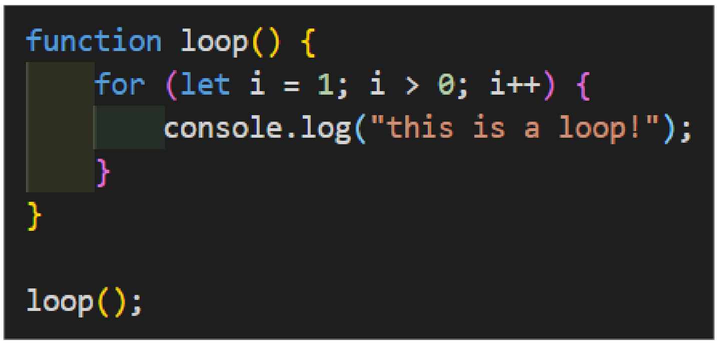
1. syntax errors that prevent the program from running,
2. runtime errors where the code has an unexpected behavior, and
3. logical errors where your the doesn't do what you intended.

Syntax errors are easiest to indentify with modern IDEs. For example, when typing on *Visual Studio Code* at times I see messages indicating that I am doing something wrong because the IDE recognizes lack of code integrity. They are the type of this example where I type the reseverd word *function* as *fuction* and I receive a warning:



```
fucntion example {    Unknown keyword or identifier. Did you mean 'function'?
    console.log("Will display an error");
}
```

The other two are not that easy to catch and they are usually detected as the program is running. For example, when running an infinite loop. An example of this kind may be:



```
function loop() {
    for (let i = 1; i > 0; i++) {
        console.log("this is a loop!");
    }
}

loop();
```

And a semantic error is often detected when analyzing the output because something didn't come out as expected. It can be as simple as when trying to multiply two numbers we realize that the output is not the expected number because we used the "+" operator instead the "*".

When debbuging we are trying to understand why the code is behaving the way it is. Usually the best tool will be Chrome or Firefox console that will help us trace our error. However, it seemed me to be an technique that can be learned but that it also requires time and practice.

Other debugging techniques might imply breaking points, isolating the code, etc. (See an interesting article here). But in the end the best tools seems to be using *console.log* to see the output in the console before going any further; testing the code by blocks while you go along coding, so you can see that up to that line all was fine. Verify that there are uniformity on variable-types (in having 7 and 3 and the first is a number and the second a string, the output will be 73 not 10); search for unclosed parenthesis or problems with quotes (when using works like *don't* in English); making sure we use equality and not equal operators in *if-else* statements; making sure we call our functions; and in case of using constants use *const* and not *let* so we make sure we don't asign different values by error.