

## Day 7 - 1/31/2024

# Lua

# Fundamentals

Single line comments are `--`, multiline are `--[==[Comments...]==]`

Not Comments where the equals can be 0 or more.

Lua is case sensitive; `abc`  $\neq$  `Abc`  $\neq$  `ABC`.

## Two kinds of string literals:

- Quoted strings: "hi"
- Multiline strings: `[===[Hello "[]" <- these are legal  
there I am a multiline string this syntax sucks]===]`

tonumber and tostring exist and functions don't use the lord's snake case :(.

Functions are declared with `function`; example:

```
function printTwice(s)
  io.write(s..s.."\\n")
end
```

```
printTwice("abc")
printTwice "abc"
printTwice(42)
```

This yields: `abcabc\nabcabc\n4242\n`

The devil once again shows his face; Lua has nil.

Lambdas exist:

```
f = function io.write("sddasfklSDKGLrkwe") end
```

Tables are initialized with {}:

```
t = { [2]="abc" }  
t[3] = "xyz"
```

```
t["abc"] = 56
print(t.abc)
```

Yields: 56

## Modules

A local variable is a variable that exist only in a function, otherwise it's always global.

```
function f()
    local n = 5 -- n = 5
    n = n + 1    -- n = 6
    local n = 2 -- a new; n = 2
    local function g()
        ..
    end
    g()
end
```

Neat trick:

```
local mm, foo -- Fixes the issue
```

```
local function mm()
    foo()
end
```

```
local function foo()
    mm()
end
```

require appends a .lua to code, treats the code as the filename of a source file, and calls the code in that file.

```
abc = require "xyz" -- Calls file xyz.lua as a function
                    -- Sets "abc" to the returned value
```

To export from a source file, don't make it local.

```
-- quark.lua
function quark.gg(...) -- gg is exported from quark.lua
```

...  
end