Pick one of the exercises below and solve it in the language of your choice. The exercises are intended to take about two hours but feel free to put in as much effort as you feel is appropriate. Put the results on GitHub. We are going to run your solution, so it shouldn’t be an essay, a diagram, or pseudocode.

Please email me when you’ve completed the exercise and we’ll schedule a time to review your solution together. Feel free to email me with any questions.

```
------------
-Exercise 1-
------------
```

Write some code that will accept an amount and convert it to the appropriate string representation.
Example:
Convert 2523.04
     to "Two thousand five hundred twenty-three and 04/100 dollars"

```
------------
-Exercise 2-
------------
```

Write some code that will evaluate a poker hand and determine its rank.
Example:
Hand: Ah As 10c 7d 6s (Pair of Aces)
Hand: Kh Kc 3s 3h 2d (2 Pair)
Hand: Kh Qh 6h 2h 9h (Flush)

```
------------
-Exercise 3-
------------
```

Write some code that accepts an integer and prints the integers from 0 to that input integer in a spiral format.

For example, if I supplied 24 the output would be:

```
20 21 21 23 24
19  6  7  8  9
18  5  0  1 10
17  4  3  2 11
16 15 14 13 12
```

```
------------
-Exercise 4-
------------
```

Write some code that evolves generations through the "game of life".

The input will be a game board of cells, either alive (1) or dead (0).

The code should take this board and create a new board for the next generation based on the following rules:

1) Any live cell with fewer than two live neighbours dies (under-population)
2) Any live cell with two or three live neighbours lives on to the next generation (survival)
3) Any live cell with more than three live neighbours dies (overcrowding)
4) Any dead cell with exactly three live neighbours becomes a live cell (reproduction)

As an example, this game board as input:

```
0 1 0 0 0
1 0 0 1 1
1 1 0 0 1
0 1 0 0 0
1 0 0 0 1
```

Will have a subsequent generation of:

```
0 0 0 0 0
1 0 1 1 1
1 1 1 1 1
0 1 0 0 0
0 0 0 0 0
```

-Exercise 5-

Write some code that can be used in a templating engine.

This should take a map of variables ("day" => "Thursday", "name" => "Billy") as well as a string
template ("${name} has an appointment on ${Thursday}") and perform substitution as needed.

This needs to be somewhat robust, throwing some kind of error if a template uses a variable that has not
been assigned, as well as provide a way to escape the strings ("hello ${${name}}" -> "hello ${Billy}")

Good Luck!

Don’t hesitate to email me with any questions.