Scala

Commentary

For this one, I didn't have to change to much. I started with my program from last time and reorganized the encrypt to be more like a functional manner. Decrypt didn't have to change at all, sweet. For solve, I googled if there was a .foreach function, and then when I learned it did, I looked up a way to generate a sequence. It easily gave me those answers given Scala's above average documentation and community and boom, I was done.

Google Searches

- scala .foreach
- scala generate sequence of int

Caesar Implementation

```
// run with scala caesar.scala
object Caesar extends App {

    println(encrypt("ATTACK AT ONCE", 4))
    println(decrypt("EXXEGO EX SRGI", 4))

    solve("abcdeFGHIJKLmnopqrstuvwxyz ,?;{[()]}", 26)

    def encrypt(input: String, shift: Int): String = {
        input
            .toUpperCase()
            .map(chr =>
                if (chr.isUpper)
                    ((chr - 65 + shift) % 26 + 65).toChar
                else
                    chr
            )
    }

    def decrypt(input: String, shift: Int): String = {
        encrypt(input, 26-shift)
    }

    def solve(input: String, maxShift: Int): Unit = {
        List.tabulate(maxShift)(_+0).foreach(i => println("Shift: " + i +
"\tResult: " + decrypt(input, i)))
    }
}
```

Output

```
EXXEGO EX SRGI
ATTACK AT ONCE
Shift: 0        Result: ABCDEFGHIJKLMNOPQRSTUVWXYZ ,?;{[()]}
Shift: 1        Result: ZABCDEFGHIJKLMNOPQRSTUVWXY ,?;{[()]}
Shift: 2        Result: YZABCDEFGHIJKLMNOPQRSTUVWX ,?;{[()]}
Shift: 3        Result: XYZABCDEFGHIJKLMNOPQRSTUVW ,?;{[()]}
Shift: 4        Result: WXYZABCDEFGHIJKLMNOPQRSTUV ,?;{[()]}
Shift: 5        Result: VWXYZABCDEFGHIJKLMNOPQRSTU ,?;{[()]}
Shift: 6        Result: UVWXYZABCDEFGHIJKLMNOPQRST ,?;{[()]}
Shift: 7        Result: TUVWXYZABCDEFGHIJKLMNOPQRS ,?;{[()]}
Shift: 8        Result: STUVWXYZABCDEFGHIJKLMNOPQR ,?;{[()]}
Shift: 9        Result: RSTUVWXYZABCDEFGHIJKLMNOPQ ,?;{[()]}
Shift: 10       Result: QRSTUVWXYZABCDEFGHIJKLMNOP ,?;{[()]}
Shift: 11       Result: PQRSTUVWXYZABCDEFGHIJKLMNO ,?;{[()]}
Shift: 12       Result: OPQRSTUVWXYZABCDEFGHIJKLMN ,?;{[()]}
Shift: 13       Result: NOPQRSTUVWXYZABCDEFGHIJKLM ,?;{[()]}
Shift: 14       Result: MNOPQRSTUVWXYZABCDEFGHIJKL ,?;{[()]}
Shift: 15       Result: LMNOPQRSTUVWXYZABCDEFGHIJK ,?;{[()]}
Shift: 16       Result: KLMNOPQRSTUVWXYZABCDEFGHIJ ,?;{[()]}
Shift: 17       Result: JKLMNOPQRSTUVWXYZABCDEFGHI ,?;{[()]}
Shift: 18       Result: IJKLMNOPQRSTUVWXYZABCDEFGH ,?;{[()]}
Shift: 19       Result: HIJKLMNOPQRSTUVWXYZABCDEFG ,?;{[()]}
Shift: 20       Result: GHIJKLMNOPQRSTUVWXYZABCDEF ,?;{[()]}
Shift: 21       Result: FGHIJKLMNOPQRSTUVWXYZABCDE ,?;{[()]}
Shift: 22       Result: EFGHIJKLMNOPQRSTUVWXYZABCD ,?;{[()]}
Shift: 23       Result: DEFGHIJKLMNOPQRSTUVWXYZABC ,?;{[()]}
Shift: 24       Result: CDEFGHIJKLMNOPQRSTUVWXYZAB ,?;{[()]}
Shift: 25       Result: BCDEFGHIJKLMNOPQRSTUVWXYZA ,?;{[()]}
```

Log

Estimate: 1 hour

Date	Hours Spent	Accomplishments
4/13	.25	Re-install scala (i'm a minimalist, sorry)
4/13	.25	Re-organize encrypt, decrypt, solve

Discrepancy of time

I was expecting to have to change the program more. I also didn't expect to have to re-install scala, I had thought I didn't uninstall it. Ultimately, I ended up saving a lot of time on this one.

Overall Review

Scala is a pretty good language overall. Functionally, it is a good choice also. Similarly, to javascript, I think it hits a good combination of being capable as both. Very rarely would the best way to design an application be purely functional or imperative. Working with arrays makes a lot of sense with map functions. Although, I prefer JS out of comfort, I have to give the objective edge to scala here. The JS approach definitely involved some "hacky" techniques but that is kinda par for the course in a JS world. Scala gets the edge for readability and loses for writability because this program was much more verbose than the JS version I wrote.

Ratings

Readability: 8/10

Writability: 7/10

Ranking: 2/5

Rankings

1. JavaScript
2. Scala
3. ML
4. Erlang
5. Lisp