**LISP**

**OUTPUT**

[1]> (load "ceaser")

;; Loading file C:\Program Files (x86)\clisp-2.49\ceaser.lsp ...

;; Loaded file C:\Program Files (x86)\clisp-2.49\ceaser.lsp

T

[2]> (encrypt "Hello World" 5)

Encrypted:

"MJQQT BTWQI"

[3]> (decrypt "Mjqqt Btwqi" 5)

Decrypted:

"HELLO WORLD"

[4]> (solve "HAL" 26)

Cipher 26: "HAL"

Cipher 25: "GZK"

Cipher 24: "FYJ"

Cipher 23: "EXI"

Cipher 22: "DWH"

Cipher 21: "CVG"

Cipher 20: "BUF"

Cipher 19: "ATE"

Cipher 18: "ZSD"

Cipher 17: "YRC"

Cipher 16: "XQB"

Cipher 15: "WPA"

Cipher 14: "VOZ"

Cipher 13: "UNY"

Cipher 12: "TMX"

Cipher 11: "SLW"

Cipher 10: "RKV"

Cipher 9: "QJU"

Cipher 8: "PIT"

Cipher 7: "OHS"

Cipher 6: "NGR"

Cipher 5: "MFQ"

Cipher 4: "LEP"

Cipher 3: "KDO"

Cipher 2: "JCN"

Cipher 1: "IBM"

Cipher 0: "HAL"

NIL

**SOURCE**

#|Converts the character based on a check

against the uppercase/lowercase value|#

(defun convert (char cipher)

(let\*

(

#|An implementation using ASCII character encoding

might return these values:

(char-code #\$) => 36

(char-code #\a) => 97 |#

(myChar (char-code char))

(fLetter

(cond

#|Try swapping these arround? |#

((and (>= myChar (char-code #\A))

(<= myChar (char-code #\Z))) (char-code #\A)))

)

)

#|This is.... ridiculous.

In order to get the notation right,

I have to space it out like this.

The addition is just confusing without the spacing to me.

|#

(if fLetter (code-char

(+

(mod

(+

(- myChar fLetter )

cipher) 26)

fLetter))

char)

)

)

;;This function does the actual encryption call

(defun encryptFun (input cipher)

(map 'string #'

(lambda (char)

(convert char cipher))

(string-upcase input))

)

;;Made this a seperate function to look pretty

(defun encrypt (input cipher)

(format t "Encrypted:")

(encryptFun input cipher)

)

;;Calls encryption but with a negative key

(defun decrypt (input cipher)

(format t "Decrypted:")

(encryptFun input (- cipher))

)

;;Calls the encryptFun rather than the

;;decrypt fun for neatness

(defun solve (input maxCipher)

(loop for i from maxCipher downto 0 do

(format t "Cipher ~s" i)

(format t ": ~s~%" (encryptFun input i))

)

)