**SCALA**

**OUTPUT**

Lets encrypt Hello World by 5

MJQQT BTWQI

Lets decrypt that with 5 :

HELLO WORLD

Lets solve for HAL with max

Cipher 0 : HAL

Cipher 1 : IBM

Cipher 2 : JCN

Cipher 3 : KDO

Cipher 4 : LEP

Cipher 5 : MFQ

Cipher 6 : NGR

Cipher 7 : OHS

Cipher 8 : PIT

Cipher 9 : QJU

Cipher 10 : RKV

Cipher 11 : SLW

Cipher 12 : TMX

Cipher 13 : UNY

Cipher 14 : VOZ

Cipher 15 : WPA

Cipher 16 : XQB

Cipher 17 : YRC

Cipher 18 : ZSD

Cipher 19 : ATE

Cipher 20 : BUF

Cipher 21 : CVG

Cipher 22 : DWH

Cipher 23 : EXI

Cipher 24 : FYJ

Cipher 25 : GZK

Cipher 26 : HAL

()

Use encrypt(message, cipher)

or decrypt(message,cipher)

or solve(message,ciphermax)

**SOURCE**

object CaesarCipher{

val upper = 'A' to 'Z'

val lower = 'a' to 'z'

def main(args: Array[String]) {

println("Lets encrypt Hello World by 5 :")

println(encrypt("HELLO WORLD", 5))

println()

println("Lets decrypt that with 5 :")

println(decrypt("MJQQT BTWQI", 5))

println()

println("Lets solve for HAL with max 26: ")

println(solve("HAL", 26))

println()

println("Use encrypt(message, cipher) \n or decrypt(message,cipher) \n or solve(message,ciphermax)")

}

def shift(c:Char, cipher:Int, mod:IndexedSeq[Char])=

{

mod( (c - mod.head + cipher + mod.size) % mod.size)

}

def encrypt(input:String, cipher:Int) =

input.map

{

case c if lower.contains(c) => shift(c, cipher, lower)

case c if upper.contains(c) => shift(c, cipher, upper)

case c => c

}

def decrypt(input:String, cipher:Int) =

{ encrypt(input, (-cipher)) }

def solve(input:String, cipher:Int) =

{

var i = 0

while (i <= cipher)

{ println("Cipher " + i + " : " + decrypt(input, i - (28 \* i)))

i = i + 1

}

}

}