**ForTran**

**OUTPUT**

**ENCRYPT**

Please input e if encrypting, d for decrypting, or s for solving:

e

Message to encrypt:

Hello world

Shift value:

5

Encrypted = Mjqqt btwqi

Press RETURN to close window . . .

**DECRYPT**

Please input e if encrypting, d for decrypting, or s for solving:

d

Message to decrypt:

Mjqqt btwqi

Shift value:

5

Decrypted = Hello world

Press RETURN to close window . . .

**SOLVE**

Please input e if encrypting, d for for decrypting, or s for solving:

s

Ecrypted message to solve for:

HAL

Max shift value:

26

Cipher: 26

Decrypted = HAL

Cipher: 25

Decrypted = GZK

Cipher: 24

Decrypted = FYJ

Cipher: 23

Decrypted = EXI

Cipher: 22

Decrypted = DWH

Cipher: 21

Decrypted = CVG

Cipher: 20

Decrypted = BUF

Cipher: 19

Decrypted = ATE

Cipher: 18

Decrypted = ZSD

Cipher: 17

Decrypted = YRC

Cipher: 16

Decrypted = XQB

Cipher: 15

Decrypted = WPA

Cipher: 14

Decrypted = VOZ

Cipher: 13

Decrypted = UNY

Cipher: 12

Decrypted = TMX

Cipher: 11

Decrypted = SLW

Cipher: 10

Decrypted = RKV

Cipher: 9

Decrypted = QJU

Cipher: 8

Decrypted = PIT

Cipher: 7

Decrypted = OHS

Cipher: 6

Decrypted = NGR

Cipher: 5

Decrypted = MFQ

Cipher: 4

Decrypted = LEP

Cipher: 3

Decrypted = KDO

Cipher: 2

Decrypted = JCN

Cipher: 1

Decrypted = IBM

Cipher: 0

Decrypted = HAL

Press RETURN to close window . . .

**SOURCE**

program Caesar\_Cipher

implicit none

!Creating variables

character :: sel

character (LEN=100) :: input

character (LEN=100) :: solve

integer :: cipher

integer :: i

!Intialize characters, otherwise issues with computation

solve = ''

!READ CASE SELECTION

print \*, "Please input e if encrypting, d for decrypting, or s for solving:"

read \*, sel

select case(sel)

!BEGIN ENCRYPTION

case('e')

print \*, "Message to encrypt: "

read (\*,'(A)') input

print \*, "Shift value: "

read \*,cipher

call encrypt(input,cipher)

!BEGIN DECRYPTION

case('d')

print \*, "Message to decrypt: "

read (\*,'(A)') input

print \*, "Shift value: "

read \*,cipher

call decrypt(input,cipher)

!BEGIN SOLVE

case('s')

print \*, "Ecrypted message to solve for: "

read (\*,'(A)') input

print \*, "Max shift value: "

read \*, cipher

do i = cipher, 0, -1

print \*,"Cipher:",i

call decrypt(input, i - (28\*i))

end do

end select

end program Caesar\_Cipher

!-----SUBROUTINES---------------------------------------------------

!

! Encrypt and decrypt functions are down here. Take input and cipher number

!

!---------------------------------------------------------------------

SUBROUTINE encrypt(input, cipher)

IMPLICIT NONE

character (LEN=100), intent(in) :: input

character (LEN=100) :: encryptChar

integer :: i

integer :: mod

integer :: cipher

encryptChar = ''

!COMPUTATION

do i = 1, len(input)

select case (input(i:i))

case ('a' : 'z')

mod = modulo(iachar(input(i:i)) - 97 + cipher, 26)

encryptChar(i:i) = achar( mod + 97)

case('A' : 'Z')

mod = modulo(iachar(input(i:i)) - 65 + cipher, 26)

encryptChar(i:i) = achar( mod + 65)

end select

end do !END COMPUTATION

print \*,''

write (\*,"(2a)") "Encrypted = ", encryptChar

END SUBROUTINE encrypt

SUBROUTINE decrypt(input, cipher)

IMPLICIT NONE

character (LEN=100), intent(in) :: input

character (LEN=100) :: decryptChar

integer :: i

integer :: mod

integer :: cipher

decryptChar = ''

!COMPUTATION

do i = 1, len(input)

select case (input(i:i))

case ('a' : 'z')

mod = modulo(iachar(input(i:i)) - 97 - cipher, 26)

decryptChar(i:i) = achar( mod + 97)

case('A' : 'Z')

mod = modulo(iachar(input(i:i)) - 65 - cipher, 26)

decryptChar(i:i) = achar( mod + 65)

end select

end do !END COMPUTATION

write (\*,"(2a)") "Decrypted = ", decryptChar

END SUBROUTINE decrypt