19F-0285

Assignment 4

Question 1:

Alphabets = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T',

'U', 'V', 'W', 'X', 'Y', 'Z']

def checkNumber(text):

i = 0

number = []

while i < len(text):

j = 0

while j < len(Alphabets):

if text[i] == Alphabets[j]:

number.append(j)

j += 1

i += 1

return number

def encrptText(textNumber,A,B):

i = 0

encrptTextNumber = []

while i < len(textNumber):

result = (((textNumber[i]\*A)%26)+B)%26

encrptTextNumber.append(result)

i+=1

encrptedText = []

i = 0

while i < len(encrptTextNumber):

j = 0

while j < len(Alphabets):

if encrptTextNumber[i] == j:

encrptedText.append(Alphabets[j])

j+=1

i+=1

text = ""

for i in encrptedText:

text += i

print("Encrypted Text = " , text)

return text

def findInverse(A):

i = 0

while i < len(Alphabets):

res = A\*i%26

if res == 1:

return i

i += 1

def decryptText(textNumber , A , B):

i = 0

decryptTextNumber = []

while i < len(textNumber):

result = (((textNumber[i] - B) % 26) \* A) % 26

decryptTextNumber.append(result)

i += 1

decryptedText = []

i = 0

while i < len(decryptTextNumber):

j = 0

while j < len(Alphabets):

if decryptTextNumber[i] == j:

decryptedText.append(Alphabets[j])

j += 1

i += 1

text = ""

for i in decryptedText:

text += i

print("Decrypted Text = ", text)

return text

Plain\_Text = input("Enter Text = ")

print("Plain Text: ", Plain\_Text)

A = 7

B = 2

TextNumber = checkNumber(Plain\_Text)

encrptedText = encrptText(TextNumber,A,B)

TextNumber = checkNumber(encrptedText)

decryptedText = decryptText(TextNumber, findInverse(A),B)

Graphical user interface, text, application

Description automatically generated

Question 2:

rows, cols = (5, 5)

def print\_matrix(solution):

for row in solution:

for val in row:

print(val, " ", end=" ")

print(" ")

def fillMatrix(matrix , key):

added = []

Alphabets = ['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z']

i = 0

j = 0

k = 0

aplha = 0

while i < rows:

while j < cols:

if k < len(key):

if key[k] not in added:

added.append(key[k])

matrix[i][j] = key[k]

j += 1

k += 1

else:

if aplha < len(Alphabets):

if Alphabets[aplha] not in added:

added.append(Alphabets[aplha])

if Alphabets[aplha] == "I":

matrix[i][j] = Alphabets[aplha] + Alphabets[aplha+1]

aplha += 1

else:

matrix[i][j] = Alphabets[aplha]

j += 1

aplha += 1

else:

j += 1

j = 0

i += 1

print()

print\_matrix(matrix)

return matrix

def plainTextCombination(plaintext):

i = 0

newPlainText = []

while i < len(plaintext) - 1:

if plaintext[i] == plaintext[i+1]:

newPlainText.append(plaintext[i])

newPlainText.append('X')

else:

newPlainText.append(plaintext[i])

if (i+1) == len(plaintext) - 1:

newPlainText.append(plaintext[i+1])

i += 1

print(plaintext)

if (len(newPlainText) % 2) != 0:

newPlainText.append('X')

text = ""

for i in newPlainText:

text += i

print("Text after: " , text)

return text

def findPostion(matrix, f,s):

i = 0

j = 0

while i < rows:

while j < cols:

val = matrix[i][j]

if val == "IJ":

if val[0] == s:

sIndex = (i, j)

#print(s, ": ", sIndex)

if val[1] == s:

sIndex = (i, j)

# print(s, ": ", sIndex)

if val[0] == f:

fIndex = (i, j)

# print(f, ": ", fIndex)

if val[1] == f:

fIndex = (i, j)

# print(f, ": ", fIndex)

if val == f:

fIndex = (i,j)

#print(f , ": ",fIndex)

if val == s:

sIndex = (i,j)

#print(s , ": ", sIndex)

j+=1

j = 0

i+=1

# print(f, ": ", fIndex)

# print(s, ": ", sIndex)

return fIndex,sIndex

def EncryptText(matrix,text):

i = 0

eccrypt = []

while i < len(text) :

first = text[i]

second = text[i+1]

findex , sindex = findPostion(matrix,first,second)

if findex[0] == sindex[0]: # for row check

r = findex[0]

c1 = findex[1] + 1

c2 = sindex[1] + 1

if c1 > 4:

c1 = 0

if c2 > 4:

c2 = 0

eccrypt.append(matrix[r][c1])

eccrypt.append(matrix[r][c2])

if findex[1] == sindex[1]: # for col check

c = findex[1]

r1 = findex[0] + 1

r2 = sindex[0] + 1

if r1 > 4:

r1 = 0

if r2 > 4:

r2 = 0

eccrypt.append(matrix[r1][c])

eccrypt.append(matrix[r2][c])

if (findex[0] < sindex[0]) and (findex[1] < sindex[1]): #moving right direction when First row < second row

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

if (findex[0] > sindex[0]) and (findex[1] > sindex[1]): #moving left direction

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

if (findex[0] > sindex[0]) and (findex[1] < sindex[1]): #moving right direction when First row < second row

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

if (findex[0] < sindex[0]) and (findex[1] > sindex[1]): # moving LEFT direction when First row < second row AND SEC COL < FIRST

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

i += 2

encryptText = ""

for i in eccrypt:

encryptText += i

print("Encrpted Text = ", encryptText)

return encryptText

def decryptText(matrix,text):

i = 0

eccrypt = []

while i < len(text):

first = text[i]

second = text[i+1]

findex , sindex = findPostion(matrix,first,second)

if findex[0] == sindex[0]: # for row check

r = findex[0]

c1 = findex[1] - 1

c2 = sindex[1] - 1

if c1 > 4:

c1 = 0

if c2 > 4:

c2 = 0

eccrypt.append(matrix[r][c1])

eccrypt.append(matrix[r][c2])

if findex[1] == sindex[1]: # for col check

c = findex[1]

r1 = findex[0] - 1

r2 = sindex[0] - 1

if r1 > 4:

r1 = 0

if r2 > 4:

r2 = 0

eccrypt.append(matrix[r1][c])

eccrypt.append(matrix[r2][c])

if (findex[0] < sindex[0]) and (findex[1] < sindex[1]): #moving right direction when First row < second row

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

if (findex[0] > sindex[0]) and (findex[1] > sindex[1]): #moving left direction

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

if (findex[0] > sindex[0]) and (findex[1] < sindex[1]): #moving right direction when First row < second row

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

if (findex[0] < sindex[0]) and (findex[1] > sindex[1]): # moving LEFT direction when First row < second row AND SEC COL < FIRST

newfindex = (findex[0], sindex[1])

newSindex = (sindex[0], findex[1])

eccrypt.append(matrix[newfindex[0]][newfindex[1]])

eccrypt.append(matrix[newSindex[0]][newSindex[1]])

i += 2

encryptText = ""

for i in eccrypt:

encryptText += i

print("Decrypted Text = ", encryptText)

key = "FIFAWORLDCUP"

matrix = []

for i in range(rows):

matrix.append([0 for i in range(cols)])

matric = fillMatrix(matrix,key)

Plain\_Text = "GERMNAYGONNAWIN"

text = plainTextCombination(Plain\_Text)

eText = EncryptText(matrix,text)

dText = decryptText(matrix,eTet)

Text

Description automatically generated

Question 3:

Alphabets = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',

'V', 'W', 'X', 'Y', 'Z']

def print\_key(key):

for row in key:

for val in row:

print(val, " ", end=" ")

print(" ")

def inputKey(key, r, c):

i = 0

while i < r:

j = 0

while j < c:

key[i][j] = int(input("Enter value at matix = "))

j += 1

i += 1

return key

def inputKey2D(key):

i = 0

while i < 2:

j = 0

while j < 2:

key[i][j] = int(input("Enter value at matix = "))

j += 1

i += 1

return key

def checkNumber(text):

i = 0

number = []

while i < len(text):

j = 0

while j < len(Alphabets):

if text[i] == Alphabets[j]:

number.append(j)

j += 1

i += 1

return number

def vectorInput(text, r):

i = 0

allVectors = []

numberVectors = []

number = checkNumber(text)

while i < len(text):

vector = []

vector.append(text[i])

vector.append(text[i + 1])

allVectors.append(vector)

j = 0

Vectors = []

while j < r:

Vectors.append(number[i + j])

j += 1

numberVectors.append(Vectors)

i += 2

return numberVectors

def multiplication(m1, m2, r, c):

i = 0

result = []

while i < r:

j = 0

res = 0

while j < c:

res += m1[i][j] \* m2[j]

j += 1

i += 1

result.append(res % 26)

return result

def encryptText(key, vectors, r, c):

i = 0

encryptedText = []

while i < len(vectors):

ecryptedNumber = []

vector = multiplication(key, vectors[i], r, c)

j = 0

while j < len(vector):

k = 0

while k < len(Alphabets):

if vector[j] == k:

encryptedText.append(Alphabets[k])

k += 1

j += 1

i += 1

text = ""

for i in encryptedText:

text += i

return text

def findDeterminant(matrix):

res = (matrix[0][0] \* matrix[1][1]) - (matrix[0][1] \* matrix[1][0])

if res < 0:

res = res \* -1

return res

def ThreeD\_Determinant(matrix):

row = int(input('Enter row fo determinant'))

col = int(input('Enter col fo determinant'))

def findInverse(A):

i = 0

while i < len(Alphabets):

res = A \* i % 26

if res == 1:

return i

i += 1

def findAdj(matrix):

matrix[0][1] = -matrix[0][1]

if matrix[0][1] < 0:

matrix[0][1] = matrix[0][1] % 26

matrix[1][0] = -matrix[1][0]

if matrix[1][0] < 0:

matrix[1][0] = matrix[1][0] % 26

temp = matrix[0][0]

matrix[0][0] = matrix[1][1]

matrix[1][1] = temp

# print\_key(matrix)

return matrix

def decryptText(key, encryptVectors):

adjoint = []

for i in range(2):

adjoint.append([0 for i in range(2)])

invVal = findInverse(findDeterminant(key))

adjoint = findAdj(key)

kinverse = []

for i in range(2):

kinverse.append([0 for i in range(2)])

# kInverse

i = 0

while i < 2:

j = 0

while j < 2:

kinverse[i][j] = (invVal \* adjoint[i][j]) % 26

j += 1

i += 1

# print\_key(kinverse)

print("Decrypted Text: ", encryptText(kinverse, encryptVectors,2,2))

def checklen(text):

if len(text)%2 == 1:

text+= 'X';

return text

plainText = "ATTACK"

plainText = checklen(plainText)

print("Plain\_Text = ", plainText)

TwoDkey = []

ThreeDkey = []

for i in range(2):

TwoDkey.append([0 for i in range(2)])

for i in range(3):

ThreeDkey.append([0 for i in range(3)])

TwoDkey = inputKey(TwoDkey,2,2)

print\_key(TwoDkey)

plainvectors = vectorInput(plainText,2)

print(plainvectors)

encryptedText = encryptText(TwoDkey,plainvectors,2,2)

print("Encrypted Text = ", encryptedText)

encryptVectors = vectorInput(encryptedText,2)

decryptedText = decryptText(TwoDkey, encryptVectors)

Text

Description automatically generated