Q1 Team name
0 Points

CryptoCreeks

Q2 Commands
10 Points

List the commands used in the game to reach the ciphertext.

Commands to use the hint and then go to the ciphertext : (go ,back, read) OR (go, go ,read) (Both works) and command to directly reach to the ciphertext: read

Q3 Cryptosystem
10 Points

What cryptosystem was used in this level?

Playfair Cipher (Morse Code for decoding the pattern to obtain the key for playfair cipher)

Q4 Analysis
20 Points

What tools and observations were used to figure out the cryptosystem? (Explain in less than 100 words)

On entering the "go" command, when we came close to the boulder, we saw some string consisting of "." and "-", which appeared to be something similar to the Morse Code. On decoding the code written on the boulder using the Morse Code,

we found that it says "SECURITY". Then in the next paragraph, it was also mentioned to "PLAY FAIR" in capitals. From there we got an idea that we should use play-fair cipher for the decryption of some ciphertext. Now, in order to use a play-fair cipher, we would also need some key. We decided to go back using the "back" command and then typed the "read" command to read what was written on the glass panel. Here, we came across a ciphertext which needed to be decrypted. Now in order to decrypt this text, we knew that we have to use the play-fair cipher. The key required for the same will be what was obtained by decoding the string using Morse Code i.e. "SECURITY".

Q5 Decryption algorithm

15 Points

Briefly describe the decryption algorithm used. Also, mention the plaintext you deciphered. (Use less than 250 words)

First, we prepared the key-square of 5X5 using the key "SECURITY". The key-square was first filled up using the unique characters of the key in order of occurrence, followed by the remaining letters of the alphabet. As per the rules, any one of the letters "I" or "J" is used in the key-square, and since our key contains the letter "I", we will not include "J" in the table.
Then the ciphertext is divided into pairs of two letters, called digraphs.
We then decrypted the ciphertext with the help of this key-square using the following rules :
a) If both the letters are in the same column, we will choose the letters in the row above them in the same column. In case any letter is present in the first row, choose from the same column and last row.
b) If both the letters are in the same row, we will choose the letter in the same row and column left to it. If case any letter is present in the first column, choose the letter in the same row and the last column.
c) If the letters are neither in the same row or column, and if the first letter is at (i,j), and the second one is at (m,n), then we will choose letter at (i,n) for the first one, and letter at (m,j) for the

second one.

After decrypting the ciphertext, we got the following plaintext:

BE WARY OF THE NEXT CHAMBER, THERE IS VERY LITTLE IOY THERE. SPEAK OUT XTHE PASSWORD "OPEN_SESAME" TO GO THROUGH. MAY XYOU HAVE THE STRENGTH FOR THE NEXT CHAMBER. TO FIND THE EXIT YOU FIRST WILXL NEXED TO UTTER MAGIC WORDS THERE.

As we know, that during encryption, if a digraph contains two same letters, then a 'X' is inserted after the first letter and then the new digraphs are encrypted. So we will remove those extra 'X's from the obtained plaintext wherever it is between 2 same letters. For example in WILXL, we will remove this X such that word becomes WILL. Also, as we know that we use any one of 'I' and 'J' during key square generation (at the time of encryption) , and since they are positioned in the same space, so in plaintext we can check which one of 'I' and 'J' forms a suitable word for that sentence. For ex: 'I' can be replaced with 'J' in IOY to form a meaningful word JOY.

The plaintext obtained from the decryption is as follows::

BE WARY OF THE NEXT CHAMBER, THERE IS VERY LITTLE JOY THERE. SPEAK OUT THE PASSWORD "OPEN_SESAME" TO GO THROUGH. MAY YOU HAVE THE STRENGTH FOR THE NEXT CHAMBER. TO FIND THE EXIT YOU FIRST WILL NEED TO UTTER MAGIC WORDS THERE.

Q6 Password

10 Points

What was the final command used to clear this level?

OPEN_SESAME

Q7 Code

0 Points

Upload any code that you have used to solve this level.

PlayFair.ipynb Download

In [1]:
```
import numpy as np
```

In [2]:
```
morse_code_chart = {".-":'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',
                    ".----" : '1',
                    "..---" : '2',
                    "...--" : '3',
                    "....-" : '4',
                    "....." : '5',
                    "-...." : '6',
                    "--..." : '7',
                    "---.." : '8',
                    "----." : '9',
                    "-----" : '0'
                   }
```

In [3]:
```
x = "... . -.-. ..- .-. .. - -.--"
cipher_text = 'TR XYCB MH AFC MUVY EOHPTCS,
AFCSS TE QCSI NTYIMS TNA AFCSC. EMRBH XAA
VAFR MIUCQPUH "LMRL_CCETOT" FN HM AKUXAHK.
OTA WANA OTXT FFU EISCWNAF HME BFU MCVA
UGTOTRE. BM HYLF IFU UVTY ANE HBSEI QYOQM
OUVSF AM EAFTE PYHYS XNSKE IFUSC.'
```

In [4]:
```
def get_key(pattern):
    s_list = pattern.split(" ")
    code = ""
    for symbol in s_list:
        code += morse_code_chart[symbol]
    return code
```

In [5]:
```
def generate_key_square(key):
    sq = list()
    i = 0
    for k in key:
        sq.append(k)
        i+=1
    for c in "ABCDEFGHIKLMNOPQRSTUVWXYZ":
        if c not in sq:
            sq.append(c)
    return np.array(sq).reshape(5,5)
```

In [6]:
```
def get_digraphs(text):
    d_list = list()
    count = 0
    s= ""
    for i,c in enumerate(text):
        if c.isalpha():
            if count == 0:
                s = c
                count = 1
            elif count == 1:
                s += c
                count = 0
                d_list.append(s)
    return d_list
```

In [11]:
```
def decrypt_text(sq):
    d_list = get_digraphs(cipher_text)
    p_list = list()
    for x in d_list:
        row1 = np.where(sq==x[0])[0][0]
        col1 = np.where(sq==x[0])[1][0]
        row2 = np.where(sq==x[1])[0][0]
        col2 = np.where(sq== x[1])[1][0]

        #same row
        if row1 == row2:
            temp = sq[row1][col1-1] +
sq[row2][col2-1]

        # same column
        elif col1 == col2:
            temp = sq[row1-1][col1] +
sq[row2-1][col2]

        else:
            temp = sq[row1][col2] + sq[row2]
[col1]

        p_list.append(temp)
```

```
    e_text = "".join(p_list)
    message = ""
    i=0
    for c in cipher_text:
        if c.isalpha():
            if e_text[i] == "X" and i!=0 and
e_text[i-1]==e_text[i+1]:
                i += 1
            else:
                message += e_text[i]
                i += 1
        else:
            message += c
    return message
```

In [12]:
```
key = get_key(x)
print(key)
```

```
SECURITY
```

In [13]:
```
sq=generate_key_square(key)
print(sq)
```

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

In [14]:
```
decrypt_text(sq)
```

Out [14]:
```
'BE WARY OF THE NEXT CHAMBER, THERE IS VERY LITTLE
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Assignment 2

GRADED

GROUP
SHRUTI WASNIK
PRADEEP KUMAR TIWARI
TANISHA RASTOGI
View or edit group

TOTAL POINTS
65 / 65 pts

QUESTION 1
Team name — **0** / 0 pts

QUESTION 2
Commands — **10** / 10 pts

QUESTION 3
Cryptosystem — **10** / 10 pts

QUESTION 4
Analysis — **20** / 20 pts

QUESTION 5
Decryption algorithm — **15** / 15 pts

QUESTION 6
Password — **10** / 10 pts

QUESTION 7
Code — **0** / 0 pts