Q1 Commands
5 Points

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

The commands we used were as follows:
1. go
2. read
3. enter
4. read

Q2 Cryptosystem
5 Points

What cryptosystem was used in this level?

The cryptosystem that we used here was substitution cipher. We used frequency analysis method here to decrypt the secret code.

Q3 Analysis
25 Points

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

We founded frequency count of all characters in the given cipher. The distribution was quite uneven like English text. Also, some words were repeating too. These observations suggested that it might be either a caesar cipher or a substitution cipher. We assumed it to be a caesar cipher first. But, none of the 26 shifts made sense. So, we tried substitution cipher next. We did their frequency analysis and substituted the characters according to their frequency count. Some repeated words were guessed by common sense like 1 letter word could be either 'A' or 'I'. We tried a lots of combinations using trial and error method along with frequency analysis to obtain the final mappings. The in depth deciphering process has been explained in the code.

Q4 Mapping
10 Points

What is the plaintext space and ciphertext space?
What is the mapping between the elements of plaintext space and the elements of ciphertext space? (Explain in less than 100 words)

Plain text space is the collection of all characters in the actual message and ciphertext space is the collection of all characters in the cipher text. We need to map these characters so that each character of plain text space is alloted a single character of cipher text space. This mapping acts as a key for encryption and decryption process. For encrypting the message, the character is looked in plain text space. Then it's mapping is found in cipher text space. Finally, the mapping character is substituted with the original character during encryption. The reverse happens during the decryption phase. Our mapped array for a-z alphabets is as follows:
['g','k','e','m','f','t','o','a','h','p','s','w','r','b','l','c','n','y','v','j','l','u','q','k','d','z']

Q5 Password
5 Points

What is the final command used to clear this level?

iRky3U5kdgt

Q6 Codes
0 Points

Upload any code that you have used to solve this level

ModCryptA1.ipynb Download

```
In [ ]: #Created a dictionary that returns count of all the alphabets in the
        given encrypted message:
        freq_all={}
        encryp="omkf pi hdn cmgef icphsck .H krg vphqkc c, fic mco kqgf ioqag
        eo qfcmckf oq ficpihdn cm .kg dcgeficu hfcm pi hdn cmklo uuncdgmc
        oqfc mc kfoq afihqfiokgq cifi cpgy cvkc yeg mfio kdck kha cokh
        kodjuck vn k fofvfo gqpojicmoqli opiyoa of kihsc nccqki oefc ynr2
        jshpck. Fi c jhkkign yok oMvr9Vix ya fiofigvfFic xvgfck. Fio kokFice"
        #Converting everything to upper case:
        encryp=encryp.upper()
        for i in encryp:
          if i in freq_all:
            freq_all[i]+=1
          else:
            freq_all[i]=1
        print(str(freq_all))
```

```
{'O': 25, 'M': 13, 'K': 27, 'F': 28, ' ': 58, 'P': 9, 'I': 22, 'H': 13, 'D
```

```
In [ ]: #Getting a better visualization of frequencies of all characters:
        dict(freq_all.items())
```

```
Out [36]: {' ': 58,
           '!': 1,
           ',': 1,
           '.': 4,
           '1': 1,
           '2': 1,
           '9': 1,
           'A': 5,
           'C': 36,
           'D': 7,
           'E': 6,
           'F': 28,
           'G': 14,
           'H': 13,
           'I': 22,
           'J': 4,
           'K': 27,
           'L': 4,
           'M': 13,
           'N': 7,
           'O': 25,
           'P': 9,
           'Q': 12,
           'R': 3,
           'S': 2,
           'U': 5,
           'V': 7,
           'X': 3,
           'Y': 6}
```

In terms of overall percentage, the data is as follows:

['A': 1.93,

'B': 0,

'C': 13.95,

'D': 2.71,

'E': 2.32,

'F': 10.85,

'G':5.42,

'H': 5.03,

'I': 8.52,

'J': 1.55,

'K': 10.46,

'L': 1.55,

'M': 5.03,

'N': 2.71,

'O': 9.68,

'P': 3.48,

'Q': 4.65,

'R': 1.16,

'S': 0.77,

'T': 0,

'U': 1.93,

'V': 2.71,

'W': 0,

'X': 1.16,

'Y': 2.32

'Z': 0}

In [ ]:
```
#Clearly, most frequent character if we ignore the space is letter
"c" with a count of 36. So, we can substitute it with "e", which is
the most frequent character in general.
encryp=encryp.replace('C','e')
print(encryp)
```

```
OMKF PI HDN eMGEF IePHSeK .H KRG VPHQKe e, FIe MeO KQSF IOQAG EO QFeMeKF OC
```

In [ ]:
```
#2nd most occuring alphabet 'F' will be replaced by 'T':
encryp=encryp.replace('F','t')
print(encryp)
```

```
OMKt PI HDN eMGEt IePHSeK .H KRG VPHQKe e, tIe MeO KQSt IOQAG EO QteMeKt OC
```

In [ ]:
```
#A single letter starting word is generally 'I' or 'A'. We tried
using 'I', but, were stuck later. So, we replace 'H' with 'A' here:
encryp=encryp.replace('H','a')
print(encryp)
```

```
OMKt PI aDN eMGEt IePaSeK .a KRG VPaQKe e, tIe MeO KQSt IOQAG EO QteMeKt OC
```

In [ ]:
```
#It can be figured out easily that the word 'TIE' must definitely be
'THE'. So, we replace 'I' by 'h'.
encryp=encryp.replace('I','h')
print(encryp)
```

```
OMKt Ph aDN eMGEt hePaSeK .a KRG VPaQKe e, the MeO KQSt hOQAG EO QteMeKt OC
```

In [ ]:
```
#The letter 'O' closely substitutes letter 'I' at many places and
makes sense. Also, letters 'OQ occur frequently. So, they must be
'IN' most probably.
encryp=encryp.replace('O','i')
encryp=encryp.replace('Q','n')
print(encryp)
```

```
iMKt Ph aDN eMGEt hePaSeK .a KRG VPanKe e, the Mei KnGt hinAG Ei nteMeKt ir
```

In [ ]:
```
#In last line 'THI K' is clarly 'THI S'. So, we replace 'K' with 'S'.
encryp=encryp.replace('K','s')
print(encryp)
```

```
iMst Ph aDN eMGEt hePaSes .a sRG VPanse e, the Mei snGt hinAG Ei nteMest ir
```

In [ ]:
```
#It seems that the last sentence continues with the first sentence.
So, we replace 'E' with 'F' and 'M' with 'R'.
encryp=encryp.replace('E','f')
encryp=encryp.replace('M','r')
print(encryp)
```

```
irst Ph aDN erGft hePaSes .a sRG VPanse e, the rei snGt hinAG fi nterest ir
```

In [ ]:
```
#'interestinA' must be 'interesting'. So, we replace 'A' by 'G'.
encryp=encryp.replace('A','g')
print(encryp)
```

```
irst Ph aDN erGft hePaSes .a sRG VPanse e, the rei snGt hingG fi nterest ir
```

In [ ]:
```
#'Gne' must be 'One'. So, we replace 'G' with 'O'.
encryp=encryp.replace('G','o')
print(encryp)
```

```
irst Ph aDN eroft hePaSes .a sRo VPanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'Uater' must be 'Later'. So, we replace 'U' with 'L'.
encryp=encryp.replace('U','l')
print(encryp)
```

```
irst Ph aDN eroft hePaSes .a sRo VPanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'Dessage' must be 'Message'. So, we replace 'D' with 'M'.
encryp=encryp.replace('D','m')
print(encryp)
```

```
irst Ph amN eroft hePaSes .a sRo VPanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'sinJle' must be 'simPle'. So, we replace 'J' by 'P'.
encryp=encryp.replace('J','p')
print(encryp)
```

```
irst Ph amN eroft hePaSes .a sRo VPanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'passLorY' must be 'passWorD'. So, we replace 'L' by 'W' and 'Y' by
'D'.
encryp=encryp.replace('L','w')
encryp=encryp.replace('Y','d')
print(encryp)
```

```
irst Ph amN eroft hePaSes .a sRo VPanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'PhamNer' must be 'ChamBer'. So, we replace 'P' by 'C' and 'N' by
'B'.
encryp=encryp.replace('P','c')
encryp=encryp.replace('N','b')
print(encryp)
```

```
irst ch amb eroft hecaSes .a sRo Vcanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'as RoV can see' must be 'as YoU can see'. So, we replace 'R' by 'Y'
and 'V' by 'U'.
encryp=encryp.replace('R','y')
encryp=encryp.replace('V','u')
print(encryp)
```

```
irst ch amb eroft hecaSes .a syo ucanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'Xuotes' must be 'Quotes'. We tried the same but the password got
incorrect. So, we figured out that it mightbe 'Kuotes' instead. So,
we replace 'X' by 'K'.
encryp=encryp.replace('X','k')
print(encryp)
```

```
irst ch amb eroft hecaSes .a syo ucanse e, the rei snot hingo fi nterest ir
```

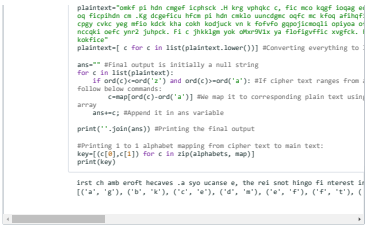
In [ ]:
```
#'caSes' must be 'caVes'. So, we replace 'S' by 'V'.
encryp=encryp.replace('S','v')
print(encryp)
```

```
irst ch amb eroft hecaves .a syo ucanse e, the rei snot hingo fi nterest ir
```

In [ ]:
```
#'B', 'T', 'W', and 'Z' occur 0 times. So, all of them can be
assigned left over letters in plain text (q,j,x,z). (W:q; T:j; W:x;
Z:z)
#Solving the space issue logically we get our final text output as
follows:
#This is the first chamber of the caves .As you can see, there is
nothing of interest in the chamber .Some of the later chambers will
be more interesting than this one!The code used for this message is a
simple substitution cipher in which digits have been shifted by 2
places. The password is iRxyOUldgt without the kuotes
# We tried this password "iRxyOUldgt" without the kuotes
#So, we figured out that "iRxyJU5kdgt" is the password. It's so
because the digits have been shifted by 2 places.
#So, the digit 2 is itself shifted by 2 places. Therefore, the
message says that the digits are shifted by 2+2= 4 places.
#9+4=3, 1+4=5. We tried with the original password but, it didn't
work. So, we read the message again and interpreted this information.
#Luckily, it worked this time.
```

Till now we manually did all the mappings to obtain the solution. To automate it using a mapping array, we can write a combined code that solves our purpose. It is given below:

In [ ]:
```
#Writing the mapping of cipher text space with original text using an array
26:
map=
['g','k','e','m','f','t','o','a','h','p','s','w','r','b','i','c','m','y','
#Creating an array of size 26 for original alphabets:
alphabets=
['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','
#Entering the cipher text provided to us in the game and storing it as pla
variable:
```



Q7 Team Name
0 Points

Noobs

Assignment 1

● GRADED

GROUP
Divyansh Bisht
Manu Shukla
Rishabh Lakhwani
View or edit group

TOTAL POINTS
38.5 / 50 pts

QUESTION 1	Commands	**5** / 5 pts
QUESTION 2	Cryptosystem	**5** / 5 pts
QUESTION 3	Analysis	R **17.5** / 25 pts
QUESTION 4	Mapping	**6** / 10 pts
QUESTION 5	Password	**5** / 5 pts
QUESTION 6	Codes	**0** / 0 pts
QUESTION 7	Team Name	**0** / 0 pts