Q1 Commands
5 Points

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

list
climb
read
enter
read

Q2 Cryptosystem
5 Points

What cryptosystem was used at this level?

At this level, the cryptosystem involves substitution cipher in which every letter is substituted by another letter from the English Alphabet according to some one-one map. Caesar cipher (substitution cipher) is used for numbers in which every digit is shifted by some fixed amount.

Q3 Analysis
25 Points

What tools and observations were used to figure out the cryptosystem?

NOTE: Failing to provide proper analysis would result in zero marks for this assignment.

First and foremost, the observation is that certain alphabet blocks, like "mewa" and "mey", appear multiple times in the ciphertext. It hints at the cryptosystem being a substitution cipher.
Now, upon frequency analysis by freq_analysis.py, the highest frequency is of "y". So, replace it with the highest frequency alphabet in normal English, "e" by replace.py.
There is only one unigram in ciphertext, "p", which appears in the middle of a sentence. The only unigrams in English are I and A. Since it is in the middle of the sentence without a comma before it, it is possibly "a" and not "i".
The most common bigram is "th". In ciphertext, "me" occurs the most.
There was "ayy" in ciphertext, which is now "_ee" after the above changes. The only common possibilities are bee and see. Considering "a" to be "s".
The next word, "meysy", is now "the_e". Possibilities are "these" or "there". "s" is already assigned to "a". So, replacing "s" with "r" and the decrypted word becomes "there".
The next word, "wa", is now "_s". "wa" occurs quite frequently, hence it could be "is" or "as". "There is" is more sensible than "There as". So, "w" is "i".
A few words are partially complete and identifiable, like "_irst", "i_terest", "i_terestin_", "ha_e _een shifte_", and "n_thin_". These could be "first", "interest", "interesting", "have been shifted", and "nothing".
More words like this, "s_bstit_tion", "_ass_ord", "_essage", and "_itho_t". These could be "substitution", "password", "message", and "without".
Lastly, "_hamber", "_ou _an see", "simp_e", and "_uotes". These could be "chamber", "you can see", "simple", and "quotes".
These were also verified by the common ciphered alphabets in them.

The decrypted text now reveals that the ciphered numbers are the case of substitution cipher, in which digits are increased by some fixed amount. Consider that to be x. The text itself shows that the digits are shifted by x places. When we would encrypt this message, it will become x+x.

This gives x equals 4. Now to decrypt "03", we will have to inverse shift it by 4, that is, shift it by 6.

The password is, thus, "tyRgU69diqq".

Tools used : To figure out, most common unigrams, bigrams and trigrams, online searches were used.

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)

Reference from Introduction to Modern Cryptography, by Katz and Lindell
Plaintext space is set of all messages supported by our encryption algorithm.
Similarly, ciphertext space is set of all messages that can be deciphered using the decryption algorithm.
The map of given ciphertext is (Note that both upper case and lower case are similar in the given encryption)
M = {a,b,d,e,f,g,h,i,j,k,m,n,o,p,r,s,t,u,v,w,x,y,0,1,2,3,4,5,6,7,8,9}
C = {s,v,q,h,p,o,n,c,m,l,t,u,b,a,g,r,f,d,w,i,y,e,6,7,8,9,0,1,2,3,4,5}
This is the respective one to one map between them. For example, a becomes s, b becomes v, d becomes q, and so on.
The collection of all strings spanned by M is plaintext space.
The collection of all strings spanned by C is ciphertext space,

Q5 Password
5 Points

What is the final command used to clear this level?

tyRgU69diqq

Q6 Codes
0 Points

Upload any code that you have used to solve this level

replace.py Download

```
#replacing some alphabet/number of the whole text

import random

alphabet = 'abcdefghijklmnopqrstuvwxyz0123456789'
alphabet1=alphabet.upper()
alphabet = alphabet+alphabet1
key = 'svjqhponcmlktubaxgrfdwiyez6789012345'
key1=key.upper()
key=key+key1
ciphertext = "Mewa wa mey twsam iepjoys gt mey ipbya. Pa xgn iph ayy, meysy wa hgmewhr gt whmysyam wh mey iepjoys. Agjy gt mey kpmys iepjoysa vwkk oy jgsy whmysyamwhr meph mewa ghy! Mey iguy nayu tgs mewa jyaapry wa p awjfky anoamwmnmwgh iwfeys wh vewie uwrwma epby oyyh aewtmyu ox 8 fkpiya. Mey fpaavgsu wa \"mxSrN03uwdd\" vwmegnm mey dngmya."

def decrypt(ciphertext, key, alphabet):
    keyMap = dict(zip(alphabet, key))
```

```
    return ''.join(keyMap.get(c, c) for c in ciphertext)

plaintext = decrypt(ciphertext, key, alphabet)

print(plaintext)
```

freq_analysis.py Download

```
#frequency analysis of given plaintext

import operator

ciphertext = "Mewa wa mey twsam iepjoys gt mey ipbya. Pa xgn iph ayy, meysy wa hgmewhr gt whmysyam wh mey iepjoys. Agjy gt mey kpmys iepjoysa vwkk oy jgsy whmysyamwhr meph mewa ghy! Mey iguy nayu tgs mewa jyaapry wa p awjfky anoamwmnmwgh iwfeys wh vewie uwrwma epby oyyh aewtmyu ox 8 fkpiya. Mey fpaavgsu wa \"mxSrN03uwdd\" vwmegnm mey dngmya."
temp={}
for char in ciphertext:
    if char.isalpha():
        temp[char.lower()]=temp.get(char.lower(),0)+1
res=dict(sorted(temp.items(), key=operator.itemgetter(1),reverse=True))
for key,value in res.items():
    print("{} -> {:.1f}%".format(key, value/len(ciphertext)*100))


```

Q7 Team Name

0 Points

ela