Q1 Commands
5 Points

List the commands was used in this level?

enter, enter, pluck, back, give, back, back, thrnxxtzy, read, the_magic_of_wand , c, c

Q2 Cryptosystem
10 Points

What cryptosystem was used in the game to reach the password?

Permutation(Columnar Transposition) + Substitution Cipher(Mono-Alphabetic) using frequency analysis. We have used block length for permutation as 5.

Q3 Analysis
30 Points

What tools and observations were used to figure out the cryptosystem and the password? (Explain in less than 1000 lines)

After entering the above command we reached the ciphertext below:

"qmnjvsa nv wewc flct vprj tj tvvplvl fv xja vqildhc xmlnvc nacyclpa fc gyt vfvw. fv wgqyp, pqq pqcs y wsq rx qmnjvafy cgv tlvhf cw tyl aeuq fv xja tkbv cqnsqs. lhf avawnc cv eas fuqb qvq tc yllrqr xxwa cfy . psdc uqf avrqc gefq pyat trac xmv taa wwd dv eas flcbq. vd trawm vupq quw x decgqcwt, yq yafl vlqs yqklhq! snafq vml lhvqpawr nqg_vfusr_ec__wawy qp fn wgawdgf "

The ciphertext is acombination of lowercase alphabets, spaces and punctutation marks.

Baesd on previous experience we ignored all the spaces and punctuation marks and try to solve without them.

After getting the ciphertext we tried to first solve it by frequency analysis and got the following stats:
Total Alphabets=284
Q 30, V 29, A 23, C 22, W 19, F 19, L 17 , T 13, Y 13, S 11, P 11, N 10, R 9, X 8, G 8, E 7, D 7, J 6, U 6, M 5, H 5, B 3, K 2, I 1

We tried solving by looking at bigrams :-
nv,tj,fv,fc,rx,cw,cv,tc,dv,vd,yq,ec,
and trigrams:-
xja,gyt,pqq,wsq,cgv,tyl,xja,lhf,eas,qvq,cfy,uqf,xmv,taa,wwd,eas, quw,vml,nqg
After trying for commonly use words like
the,an,be,an,of,by,go etc. we failed to simplify the ciphertext and thus concluded that it is not using frequency analysis cipher.

Next we tried to look if whether it was using Gronsfeld or Vignere substitution cipher as previous level ciphertext . For that we need keys and so we did some investigation.
Th spirit to which we gave the mushroom told us magic words "thrnxxtzy" . But that words were used to reach to the ciphertext itself unlike in previous level where the numbers were the key of ciphertext. Anyway thinking this be the key we tried to solve the ciphertext.
With "thrnxxtzy" as key we solved for one line of ciphertext and the output came as
"xfwwyvh ox dxfp ioju xwks gm wcwrsou sy aqb xxbuqkf"
which did not make any sense and thus we concluded that it is not using vignere sbstitution cipher.

Thus we figured that there has to be transposition present.

Also looking at the ciphertext and based on previous experiences we knew that "nqg_vfusr_ec_wawy" must be the password.
Also the words "snafq vml lhvqpawr" must be "speak the

password" and thus "l " must somehow become "s" as 'ss' of 'password' .
So we start our work keeping in focus the above line. After ignoring all the spaces and punctuation marks the total words came to be 284. As the transposition has to be done in segments of equal sizes, the multiples of 284 closest being 4 we broke the entire ciphertext into segments of 4.
With our focus on "speak the password " the segments came as:

```
lhqs  nafq  vmll  hvqp  awrn
***s  peak  thep  assw  ord*
```

Here above the problem is that the "ss" of "password " is coming up in "hvqp" block but by common sense we know that "ll" of "vmll" will translate to "ss". But in above configuration it is not possible as transposition has to be done within block and thus the "ll " will remain in previous block away from the expected position . After transposition and substitution we cannot get "ss" in "hvqp " as two alphabets must be same . So we figure we are doing it wrong.

After failing for 4 we tried for segment length of 5 and with focus again on "speak the password " the corresponding segments came as:

```
lhqsn   afqvm   llhvq   pawrn
***sp   eakth   epass   word*
```

In the above "ll" are coming in the same segment as "ss" of "password" so after transposition
and substitution we can get "speak the password" and thus we figure it might be right.

Now for transposition and assuming monoalphabetic substitution we notice the common words in segments of both ciphertext and plaintext as below:

```
                12345  12345  12345  1 2 3 4 5
ciphertext:     lhqsn  afqvm  llhvq  p a w r n
                12345  12345  12345  1 2 3 4 5
```

plaintext : * * * s p e a k t h e p a s s w o r d *

Based on above we figure that :

1.) 3rd block has 2L as "ll" in ciphertext and 2S as "ss" in plaintext thus l = s.
2.)2nd and 3rd block have 'q','v' common in ciphertext and 'e','a' common in plaintext. Thus either" 'q=a','v=e'" or "'q=e','v=a'"
3.) Also in 3rd block only 'h' is left in ciphertext and "p" in plaintext. Thus h = p.
This is also verified by looking at 1st block.
4.) Since l = s thus looking at 1st block we figure that 1st alphabet in a block gets shifted to 4th position before substitution which is also true when we look at 3rd block.
5.)From 3rd block and 1st & 3rd point observation we figure that 2nd alphabet in a block gets shifted to 5th position before substitution.
6.)In 3rd block since h = p, thus 3rd alphabet in ciphertext gets shifted to 2nd position in plaintext before substitution .
7.)Using 6th point and from second block since 3rd alphabet in ciphertext which is q gets shifted to 2nd position before substitution we figure that q = a.
8.)From 7th and 2nd point we figure that q=a & v = e.
9.)Using v = e and 2nd block we figure that 4th position in ciphertext gets shifted to 1st position.
10.)Since we are left with 5th alphabet in ciphertext and 3rd alphabet in plaintext we figure that 5th alphabet gets shifted to 3rd position.

Thus the sequence of permutation is as follows:
ciphertext (alphabet position) : 1 2 3 4 5
ciphertext(after permutation) : 4 3 5 1 2

Applying above observations:

1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
ciphertext: l h q s n a f q v m l l h v q p a w r

n

43512 435 12 43512 435 1 2

After permutation: s q n l h v q m a f v h q l l r w n p a

Plaintext: * * * s p e a k t h e p a s s w o r d *

From above we know that:
l=s ,h = p , v=e , q=a , m=k , a=t , f=h , r=w , w=o , n=r , p=d

Applying permutatiom to the whole ciphertext in subsets of 5 alphabets we get the new ciphertext as:

" jnvqm vnwsa fclew pvrct tjvjt vllvp jxafv lidvq mxlhc ncanv lcpcy gcyaf vfwtv gwqfv qpqyp scypq rqxws jnvqm cygaf vlhvt twyfc ueqla jxafv vbctk qssqn afvlh cncaw safve qbvuq yclqt rqxlr cafxw dscyp afvuq gcerq ypafq arctt tvaxw dwdaw safve qbvlc arwdt puqmv xwdqu qgcec qyywt vllaf qykqs sqnlh vqmaf vhqll rwnpa fvuqg cewsr qypaw gwafn fg%wd"

This can be solved by using the frequency analysis result that we tried first and substituting the alphabets that we got from above.

Substituting l=s, h=p, v=e, q=a, m=k, a=t, f=h, r=w, w=o, n=r, p=d and placing the new ciphertext according the original ciphertext so that we can get to know more bigrams and trigrams we get :

"#reaker o# th#s #ode w### #e ##essed ## the s#eak# sp#r#t res#d### ## the ho#e. #o ahead a#d ###d a wa# o# #reak### th e spe## o# h## #ast ## the e### #a##ar. the sp#r#t o# the ca#e#a# #s a#wa#s w#th #o#. ###d the #a### wa#d that w### #et #o# o#t o# the #a#es. #t wo##d #ake #o# a #a####a# #o #ess tha# #a##ar! speak the password the_#a###_o#_wa#d to #othr h# o# "

From the above half decoded plaintext, looking at possibe bigrams and trigrams and the sense made we can tell about other alphabets as follows:
1.First word must be"breaker" thus j=b
2. Second word will be "of" thus s=f
3.Third word will be "this " thus c= i
4.In fourth word after putting c = i we get "witt" thus for the word to make sense "t = l"

Similarly repeating for other we get the dictionary for other alphabets as
j=b, s=f, c=i,e=c,t=l, x=y,y=n,g=g,u=m,d=u

After putting above value the modified plaintext we get is is:

"breaker of this code will be blessed by the s##eaky spirit residing in the hole. go ahead and find a way of breaking the spell on him cast by the evil #affar. the spirit of the caveman is always with you . find the magic wand that will let you out of the caves. it would make you a magician no less than #affar! speak the password the_magic_of_wand to go through."

From above the words that are left are s#ueaky, #affar and the alphabets that are left to be replaced are k,i, and choices that these can taken are j,z,q.

Thus we figure that k=j,i=q. Putting in ciphertext we get:
""breaker of this code will be blessed by the squeaky spirit residing in the hole. go ahead and find a way of breaking the spell on him cast by the evil jaffar. the spirit of the caveman is always with you . find the magic wand that will let you out of the caves . it would make you a magician no less than jaffar! speak the password the_magic_of_wand to go through.""

The punctuation, underscore and spaces of the cipher text is considered to be as is in the decrypted text.

Password: the_magic_of_wand

The attached code was used to decode the ciphertext

Q4 Password
5 Points

What was the final command used to clear this level?

the_magic_of_wand

Q5 Codes
0 Points

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

assignment3.py Download

```
1  import numpy as np
```

```
ciphertext=" qmnjvsa nv wewc flct vprj tj tvvplvl fv xja vqildhc xmlnvc nacyclpa fc gyt vfvw. fv wgqyp, pqq pqcs y wsq rx qmnjvafy cgv tlvhf cw tyl aeuq fv xja tkbv cqnsqs. lhf avawnc cv eas fuqb qvq tc yllrqr xxwa cfy. psdc uqf avrqc gefq pyat trac xwv taa wwd dv eas flcbq. vd trawm vupq quw x decgqcwt, yq yafl vlqs yqklhq! snafq   vml lhvqpawr nqg_vfusr_ec_wawy qp fn wgawdgf."
print("The ciphertext is\n ")
print(ciphertext)
alphabets=[c for c in ciphertext if c.isalpha()]
result=''.join(alphabets)
print("\nAfter removing all spaces and punctuation marks\n")
print(result)

subsets = []

for i in range(0,len(result), 5):
    subset = result[i:i+5]
    subsets.append(subset)
print("\nDivind into subsets of 5\n")
print(subsets)
# add padding to the last subset if its length is less than 5
if len(subsets[-1]) < 5:
    subsets[-1] += '%' * (5 - len(subsets[-1]))

# concatenate the elements of the subsets in a specific order
reshuffled_subsets = []
for subset in subsets:
    reshuffled_subsets.append(subset[3] + subset[2] + subset[4] + subset[0] + subset[1])
print('\n After permutation in a specific order\n')
print(reshuffled_subsets)
#print('\n After concatenaring all the subsets\n')
#result1 = ''.join(reshuffled_subsets)
#print(result1)
# Example dictionary to map letters to new letters
dictionary = {
  'a': 't', 'b': '_', 'c': '_', 'd': '_', 'e': '_', 'f': 'h', 'g': '_', 'h': 'p',
  'i': '_', 'j': '_', 'k': '_', 'l': 's', 'm': 'k', 'n': 'r', 'o': '_', 'p': 'd',
  'q': 'a', 'r': 'w', 's': '_', 't': '_', 'u': '_', 'v': 'e', 'w': 'o', 'x': '_',
  'y': '_', 'z': '_'
}

```

```
# Create a translation table using the dictionary
translation_table = str.maketrans(dictionary)

# Translate each subset and join them together
translated_subsets = []
for subset in reshuffled_subsets:

translated_subsets.append(subset.translate(translation_tab
result = (translated_subsets)
print('\n After substituting some known letters in 1st
iteration \n')
# Print the result
print(result)

dictionary = {
  'a': 't', 'b': 'y', 'c': 'i', 'd': '_', 'e': 'c', 'f':
'h', 'g': 'g', 'h': 'p',
  'i': '_', 'j': 'b', 'k': '_', 'l': 's', 'm': 'k', 'n':
'r', 'o': '_', 'p': 'd',
  'q': 'a', 'r': 'w', 's': 'f', 't': 'l', 'u': '_', 'v':
'e', 'w': 'o', 'x': 'y',
  'y': 'n', 'z': '_'
}

# Create a translation table using the dictionary
translation_table = str.maketrans(dictionary)

# Translate each subset and join them together
translated_subsets = []
for subset in reshuffled_subsets:

translated_subsets.append(subset.translate(translation_tab
result = (translated_subsets)
print('\n After substituting some known letters in 2nd
iteration \n')
# Print the result
print(result)
dictionary = {
  'a': 't', 'b': 'v', 'c': 'i', 'd': 'u', 'e': 'c', 'f':
'h', 'g': 'g', 'h': 'p',
  'i': 'q', 'j': 'b', 'k': 'j', 'l': 's', 'm': 'k', 'n':
'r', 'o': '#', 'p': 'd',
  'q': 'a', 'r': 'w', 's': 'f', 't': 'l', 'u': 'm', 'v':
'e', 'w': 'o', 'x': 'y',
  'y': 'n', 'z': '#'
}

# Create a translation table using the dictionary
translation_table = str.maketrans(dictionary)
```

```
77
78  # Translate each subset and join them together
79  translated_subsets = []
80  for subset in reshuffled_subsets:
81
    translated_subsets.append(subset.translate(translation_tab
82  result = (translated_subsets)
83  print('\n After substituting some known letters in 3rd and
    final iteration \n')
84  # Print the result
85  print(result)
86
87  print('\n After concatenating all the subsets\n')
88  result1 = ''.join(result)
89  print(result1)
90  print("\nThe password is \n ")
```

Q6 Group name

0 Points

hustler

Assignment 3

● Graded

Group

SANDULA LAVANYA
ABHIVADAN
SHRUTIKA ANIL JADHAV
View or edit group

Total Points

46 / 50 pts

Question 1

Commands **5** / 5 pts

Question 2

Cryptosystem **10** / 10 pts

Question 3

Analysis R **26** / 30 pts

Question 4

Password **5** / 5 pts

Question 5

Codes **0** / 0 pts

Question 6

Group name **0** / 0 pts