

### Q1 Commands

5 Points

List the commands was used in this level?

go, enter, pluck, back, give, back,  
back, thrnxtzy, read

### Q2 Cryptosystem

10 Points

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

A monoalphabetic substitution permutation cipher is the cryptosystem that is being employed.

The entire plaintext is first separated into blocks of a specifically fixed length (here,5), and each block is then used to encrypt a single plaintext in this cryptosystem. The characters in this block are then all subjected to substitution, which changes each character into a different one (which may or may not be identical to the original character before replacement, but the mapping between the original character and substituted character is one-to-one). A block of five characters results from substitution. This is then given to the permutation operation, which applies a permutation (shuffling) operation to the block. A block of five characters is again the result of permutation. The operations must be reversed in order to conduct decryption. To put it another way, we must first undo the permutation and then the substitution.

### Q3 Analysis

30 Points

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

As the word lengths in the ciphertext match, we deduced that the first four words must be "breaker of this code" based on previous rounds. The third character in the ciphertext would be the same as the fifth character if the cipher was a monoalphabetic substitution cipher. Monoalphabetic substitution cipher was disregarded because this is not the case. So did bigram and trigram frequency analysis on cipher text and found the following result.

since bigrams and trigrams frequency is very less then there must be Transposition present.

We also attempted to use the Vigenere cipher with the wording "thrnxtzy" provided by the spirit, but the plaintext that was produced was erroneous.

Thus, we assumed the plaintext had been subjected to a monoalphabetic substitution-permutation network. Monoalphabetic substitution-permutation was tested once. This means that in order to accomplish decryption, we must first undo the permutation. Next, we must decode the ciphertext that results as a substitution cipher in order to obtain the plaintext.

Permutation analysis: Block lengths 2 and 3 were not tried because they would provide very little protection.

First, we tested a block length of 4, but the results were not as expected.

Next, we attempted a block length of 5. We decoded the first four words and discovered that the blocks sanvw and ewcfl, which correspond to the letters eroft and hisco, only share the character w. (corresponds to o). As a result, we discovered that in permutation, the 2nd position maps to the 5th position and the 5th position to the 3rd position.

We now employ the first two blocks, qmnjv (which corresponds to the break) and sanvw (which corresponds to eroft). Two frequently used characters—n and v—could stand in for the letters e or r. Due to its higher frequency in the ciphertext's single-character frequency analysis, we assumed that v corresponded to e. This also implies that n and r are equivalent. In the block sanvw, where position v corresponds to eroft, we learned that position 4th maps to position 1st, and in the block qmnjv, where position n corresponds to break, we learned that position 3rd maps

to position 2nd. The only remaining mapping is that 1st place corresponds to 4th place. The permutation is therefore discovered to be 12345 -> 43512.

Using single character, bigram, and trigram frequency analysis to study monoalphabetic substitution  
We calculated the frequencies of single characters, bigrams, and trigrams in the ciphertext after applying this permutation.

The most frequent bigrams were af and fv, and the most frequent trigram was afv. The most frequent trigram and bigram in the English language are "the" and "he," respectively. This provided a clear indication that afv was mapped to "the," with af to "th" and fv to "he." So, the mapping a->t, f->h, and v->e.

As the first four words are "cracker of this code," the frequently occurring bigram ws was known to be "of," yielding the mapping w->o, s->f.

Letters were matched as follows since the initial word, jnvqmvn, means "breaker":

q->a, m->k, j->b

In the English language, the often occurring trigram 'and' was mapped to the frequently occurring trigram qyp, whose initial letter q is known to be mapped to a. The mappings y->n and p->d are provided.

The word afcl, in which af is known to be "th" according to the bigram analysis, was mapped to "this" ('that' is not possible because I can't map to t because an already maps to t). This translated c->i to l->s.

Bigram jx, a two-letter word that appears twice, is assumed to be the word "by" because j is known to map to b. Hence, x->y.

The commonly occurring three-letter word xwd (which appears three times) is mapped to the often occurring three-letter word "you" in the English language. Hence, d->u. In the three-letter word fcu, where fc is known to map to "hi," "his" was translated to "him" (because I already maps to "s"). Hence, u->m.

The four-letter word vbct, where vbc is known to map to the word "evi," really reads "evil." Hence, t->l

Four-letter word rcaf is mapped to "with" even though caf

is known to transfer to "ith." Hence, r->w.  
 These several characters' mappings provided sufficient hints for understanding the remaining plaintext. Every punctuation mark, including the full stop, comma, exclamation point, and underscore, corresponds to itself in the plaintext. In other words, .->., ,->, , -> ,!->!

#### 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.

▼ decrypted.py

Download

```


1  #!/usr/bin/env python
2  # coding: utf-8
3
4  # In[16]:
5
6
7  from collections import Counter
8  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 cqsqs. lhf avawnc cv eas fuqb qvq tc yllrqx xxwa
   cfy. psdc uqf avrqc gefq pyat trac xwv taa wwd dv eas
   flcbq. vd trawm vupq quw x decgqwt, yq yaf1 vlqs
   yqklhq! snafq vml lhvqpawr nqg_vfusr_ec_wawy qp fn
   wgawdgf"
9
10 # Join all lines into a single string
11 text = ''.join(filter(str.isalpha, ciphertext)).lower()
12
13 #####
14
15 # Count the frequency of each letter in the ciphertext
16 frequency = Counter(text)
17
```

```
18 # Print the top 5 most frequent letters in the
    ciphertext
19 for letter, count in frequency.most_common(5):
20     print(f"{letter}: {count}")
21
22 #####
23
24
25 # Undo the permutation operation
26 permuted = ''
27 for i in range(0, len(text), 5):
28     permuted += text[i+2] + text[i+0] + text[i+3] +
    text[i] + text[i+1]
29
30 # Replace spaces at their original positions
31 spaces = []
32 for i, c in enumerate(text):
33     if c == ' ':
34         spaces.append(i)
35 permuted_list = list(permuted)
36 for i in spaces:
37     permuted_list[i] = ' '
38 permuted = ''.join(permuted_list)
39
40 # Count the frequency of bigrams and trigrams
41 trigrams = {}
42 bigrams = {}
43 words = permuted.split()
44 for w in words:
45     for i in range(len(w) - 1):
46         bigram = w[i:i+2]
47         bigrams[bigram] = bigrams.get(bigram, 0) + 1
48     for i in range(len(w) - 2):
49         trigram = w[i:i+3]
50         trigrams[trigram] = trigrams.get(trigram, 0) +
    1
51
52 # Sort and print the frequency of bigrams and trigrams
53 sorted_bigrams = sorted(bigrams.items(), key=lambda x:
    x[1], reverse=True)
54 sorted_trigrams = sorted(trigrams.items(), key=lambda
    x: x[1], reverse=True)
55 print(sorted_bigrams)
56 print(sorted_trigrams)
57
58
59 # In[ ]:
60
61
62
63
```

Q6 Group name

0 Points

da\_vinci

|   |             |
|---|-------------|
| Assignment 3  | ● Graded    |
| Group   |             |
| ANSHUL SHARMA   |             |
| PRADEEP CHALOTRA  |             |
| SUMIT KUMAR CHAUDHARY   |             |
|  <a href="#">View or edit group</a> |             |
| Total Points  |             |
| 46 / 50 pts   |             |
| Question 1  |             |
| <a href="#">Commands</a>  | 5 / 5 pts   |
| Question 2  |             |
| <a href="#">Cryptosystem</a>  | 10 / 10 pts |
| Question 3  |             |
| <a href="#">Analysis</a>  | 26 / 30 pts |
| Question 4  |             |
| <a href="#">Password</a>  | 5 / 5 pts   |
| Question 5  |             |
| <a href="#">Codes</a>   | 0 / 0 pts   |
| Question 6  |             |
| <a href="#">Group name</a>  | 0 / 0 pts   |