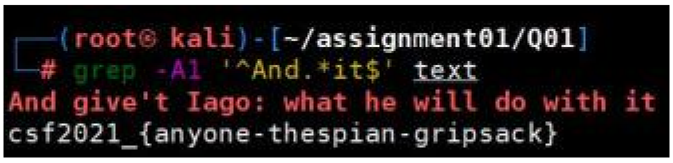
Q01

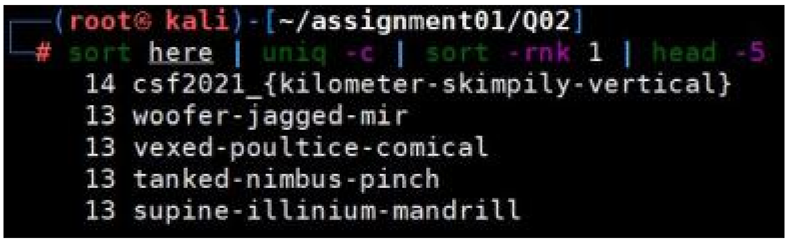
The passphrase: **csf2021_{anyone-thespian-gripsack}**



```
┌──(root㉿ kali)-[~/assignment01/Q01]
└─# grep -A1 '^And.*it$' text
And give't Iago: what he will do with it
csf2021_{anyone-thespian-gripsack}
```

Q02

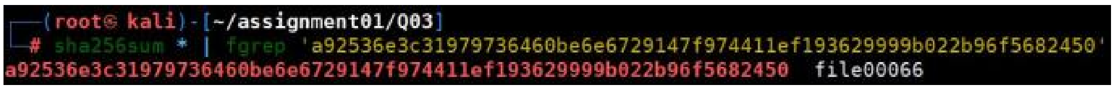
The passphare: **csf2021_{kilometer-skimpily-vertical}**



```
┌──(root㉿ kali)-[~/assignment01/Q02]
└─# sort here | uniq -c | sort -rnk 1 | head -5
     14 csf2021_{kilometer-skimpily-vertical}
     13 woofer-jagged-mir
     13 vexed-poultice-comical
     13 tanked-nimbus-pinch
     13 supine-illinium-mandrill
```

Q03

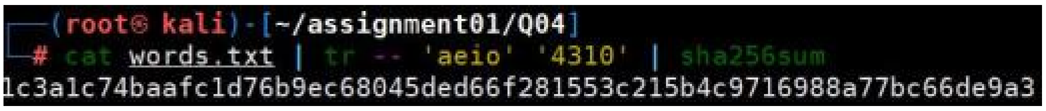
The file: **./file00066**



```
┌──(root㉿ kali)-[~/assignment01/Q03]
└─# sha256sum * | fgrep 'a92536e3c31979736460be6e6729147f974411ef193629999b022b96f5682450'
a92536e3c31979736460be6e6729147f974411ef193629999b022b96f5682450  file00066
```

Q04

Sha256: **1c3a1c74baafc1d76b9ec68045ded66f281553c215b4c9716988a77bc66de9a3**

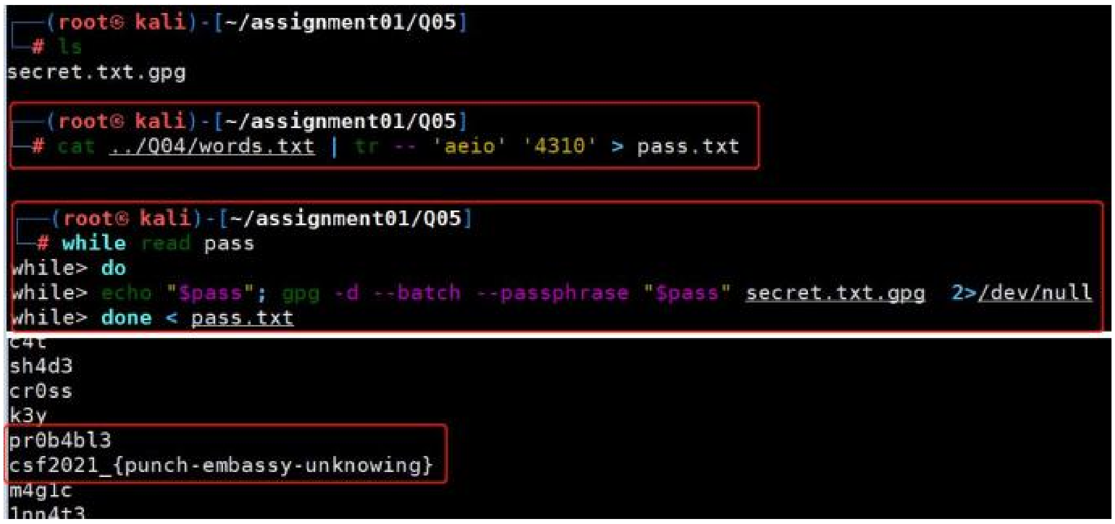


```
┌──(root㉿ kali)-[~/assignment01/Q04]
└─# cat words.txt | tr -- 'aeio' '4310' | sha256sum
1c3a1c74baafc1d76b9ec68045ded66f281553c215b4c9716988a77bc66de9a3
```

Q05

The password: **pr0b4bl3**

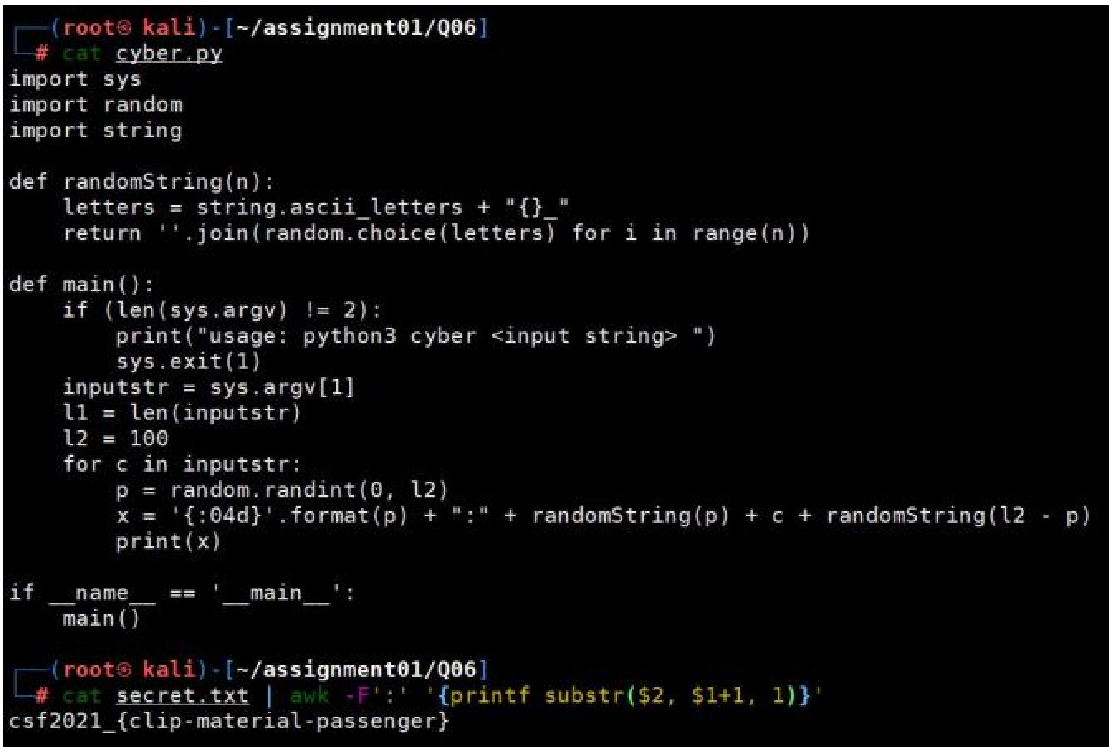
The content: **csf2021_{punch-embassy-unknowing}**



```
┌──(root㉿ kali)-[~/assignment01/Q05]
└─# ls
secret.txt.gpg

┌──(root㉿ kali)-[~/assignment01/Q05]
└─# cat ../Q04/words.txt | tr -- 'aeio' '4310' > pass.txt

┌──(root㉿ kali)-[~/assignment01/Q05]
└─# while read pass
while> do
while> echo "$pass"; gpg -d --batch --passphrase "$pass" secret.txt.gpg  2>/dev/null
while> done < pass.txt
c4t
sh4d3
cr0ss
k3y
pr0b4bl3
csf2021_{punch-embassy-unknowing}
m4g1c
1nn4t3
```

Q06

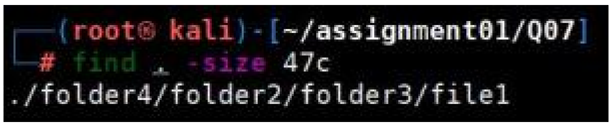
Through the encryption script I get the decryption algorithm, and then after decryption I get the secret: **csf2021_{clip-material-passenger}**



```
┌──(root㉿ kali)-[~/assignment01/Q06]
└─# cat cyber.py
import sys
import random
import string

def randomString(n):
    letters = string.ascii_letters + "{}_"
    return ''.join(random.choice(letters) for i in range(n))

def main():
    if (len(sys.argv) != 2):
        print("usage: python3 cyber <input string> ")
        sys.exit(1)
    inputstr = sys.argv[1]
    l1 = len(inputstr)
    l2 = 100
    for c in inputstr:
        p = random.randint(0, l2)
        x = '{:04d}'.format(p) + ":" + randomString(p) + c + randomString(l2 - p)
        print(x)

if __name__ == '__main__':
    main()

┌──(root㉿ kali)-[~/assignment01/Q06]
└─# cat secret.txt | awk -F':' '{printf substr($2, $1+1, 1)}'
csf2021_{clip-material-passenger}
```

Q07

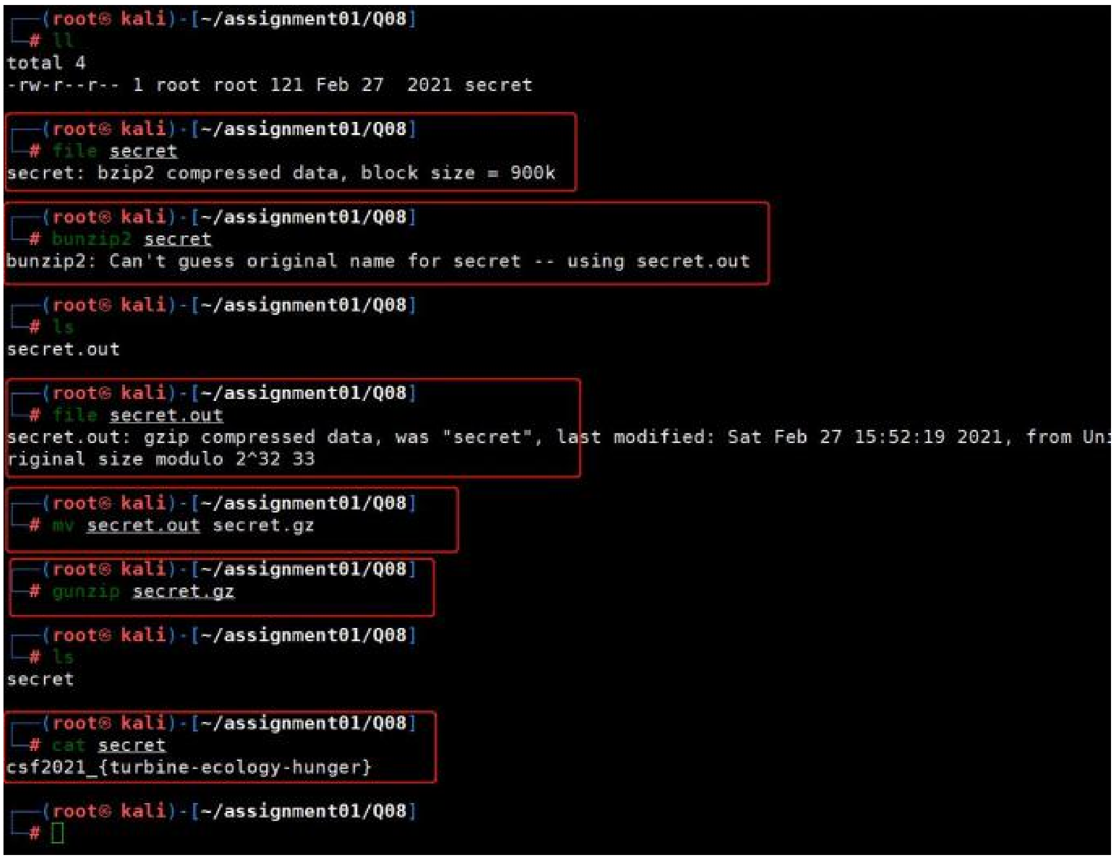
The File: **./folder4/folder2/folder3/file1**



```
┌──(root㉿ kali)-[~/assignment01/Q07]
└─# find . -size 47c
./folder4/folder2/folder3/file1
```

Q08

The secret: **csf2021_{turbine-ecology-hunger}**

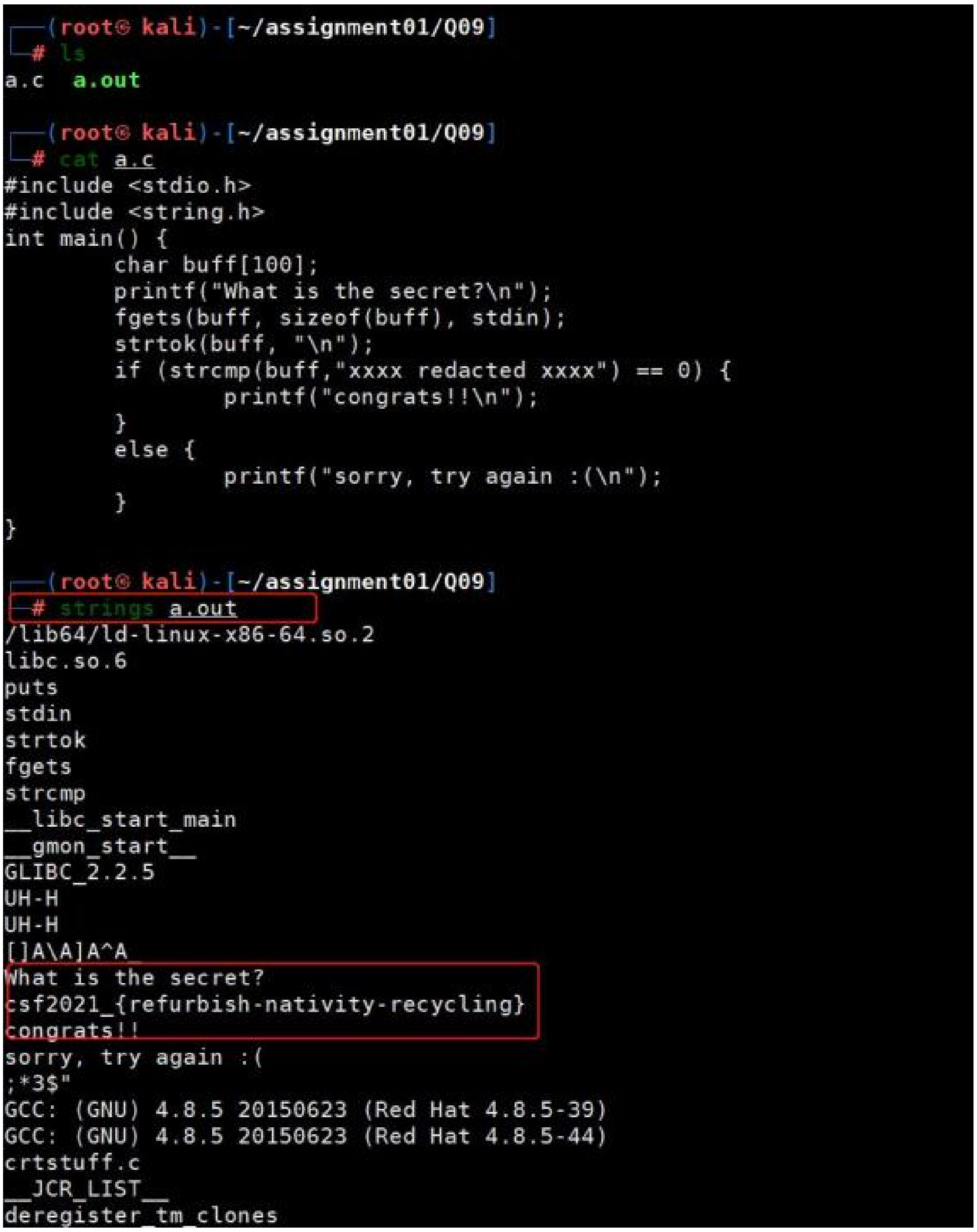


```
┌──(root㉿ kali)-[~/assignment01/Q08]
└─# ll
total 4
-rw-r--r-- 1 root root 121 Feb 27  2021 secret

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# file secret
secret: bzip2 compressed data, block size = 900k

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# bunzip2 secret
bunzip2: Can't guess original name for secret -- using secret.out

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# ls
secret.out

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# file secret.out
secret.out: gzip compressed data, was "secret", last modified: Sat Feb 27 15:52:19 2021, from Un
riginal size modulo 2^32 33

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# mv secret.out secret.gz

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# gunzip secret.gz

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# ls
secret

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# cat secret
csf2021_{turbine-ecology-hunger}

┌──(root㉿ kali)-[~/assignment01/Q08]
└─# 
```

Q09

The secret: **csf2021_{refurbish-nativity-recycling}**

The location of the password was found by looking at the source code. View the binaries directly through the strings command to get the password.

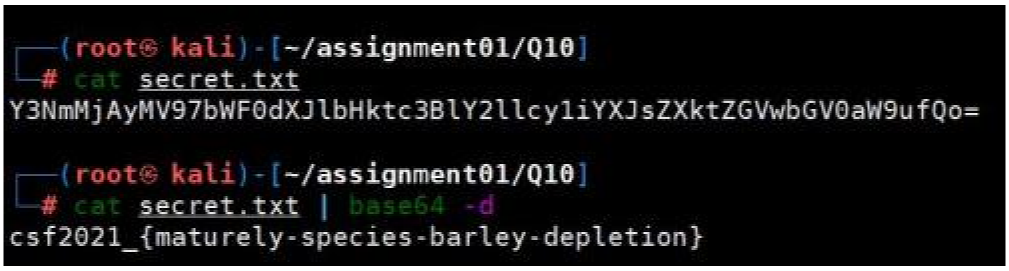


```
┌──(root㉿ kali)-[~/assignment01/Q09]
└─# ls
a.c  a.out

┌──(root㉿ kali)-[~/assignment01/Q09]
└─# cat a.c
#include <stdio.h>
#include <string.h>
int main() {
        char buff[100];
        printf("What is the secret?\n");
        fgets(buff, sizeof(buff), stdin);
        strtok(buff, "\n");
        if (strcmp(buff,"xxxx redacted xxxx") == 0) {
                printf("congrats!!\n");
        }
        else {
                printf("sorry, try again :(\n");
        }
}

┌──(root㉿ kali)-[~/assignment01/Q09]
└─# strings a.out
/lib64/ld-linux-x86-64.so.2
libc.so.6
puts
stdin
strtok
fgets
strcmp
__libc_start_main
__gmon_start__
GLIBC_2.2.5
UH-H
UH-H
[]A\A]A^A_
What is the secret?
csf2021_{refurbish-nativity-recycling}
congrats!!
sorry, try again :(
;*3$"
GCC: (GNU) 4.8.5 20150623 (Red Hat 4.8.5-39)
GCC: (GNU) 4.8.5 20150623 (Red Hat 4.8.5-44)
crtstuff.c
__JCR_LIST__
deregister_tm_clones
```

Q10

The secret: **csf2021_{maturely-species-barley-depletion}**

After seeing the contents of the file, I suspect that the text is encoded by base64.

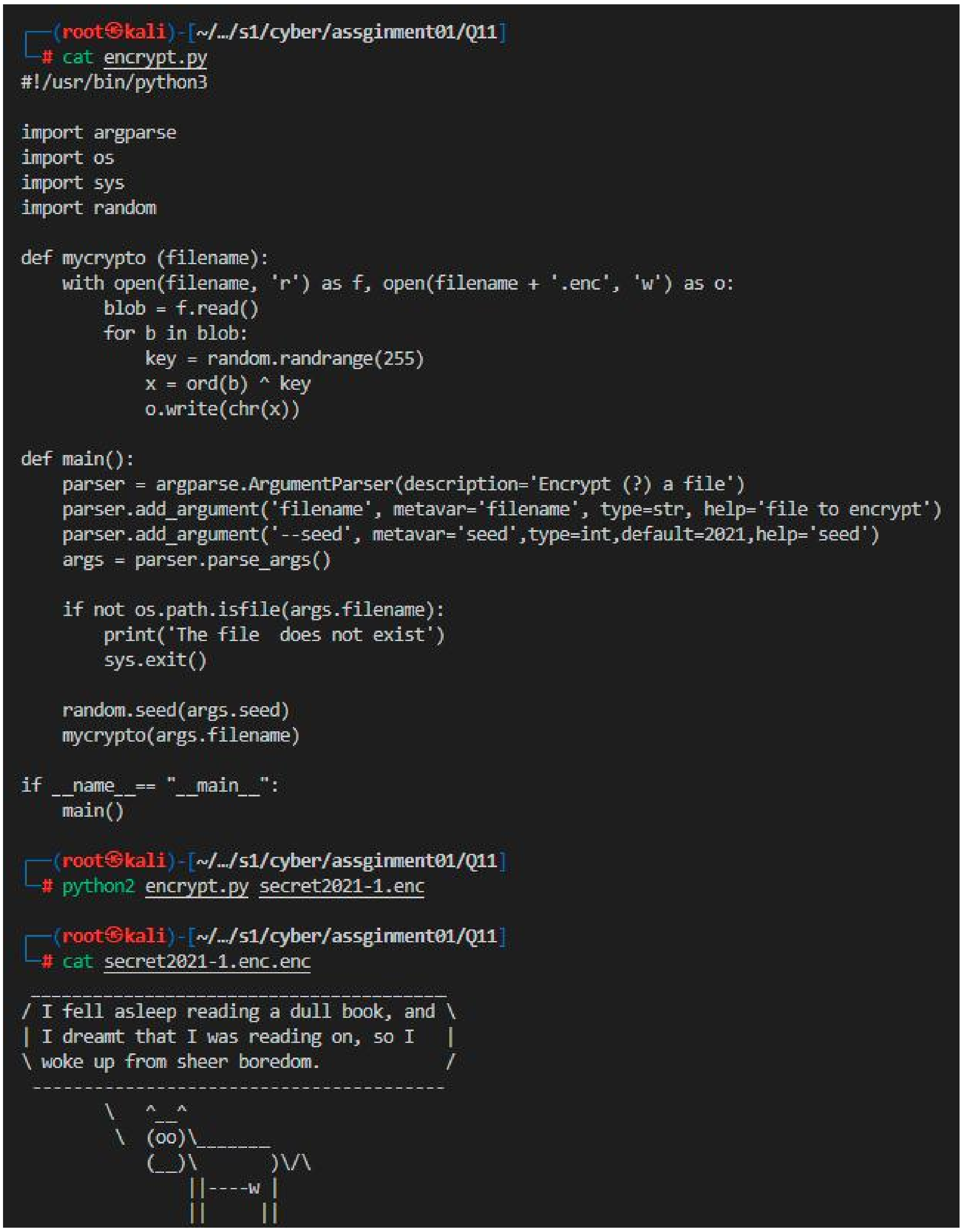


```
┌──(root㉿ kali)-[~/assignment01/Q10]
└─# cat secret.txt
Y3NmMjAyMV97bWF0dXJlbHktc3BlY2llcy1iYXJsZXktZGVwbGV0aW9ufQo=

┌──(root㉿ kali)-[~/assignment01/Q10]
└─# cat secret.txt | base64 -d
csf2021_{maturely-species-barley-depletion}
```

Q11

The secret: **I fell asleep reading a dull book, and | I dreamt that I was reading on, so I   woke up from sheer boredom.**

The script shows that the encryption uses an XOR algorithm, so the decryption method should be to encrypt the ciphertext again. In addition, the encryption and decryption also requires a seed, which I guess it is the default: 2021.

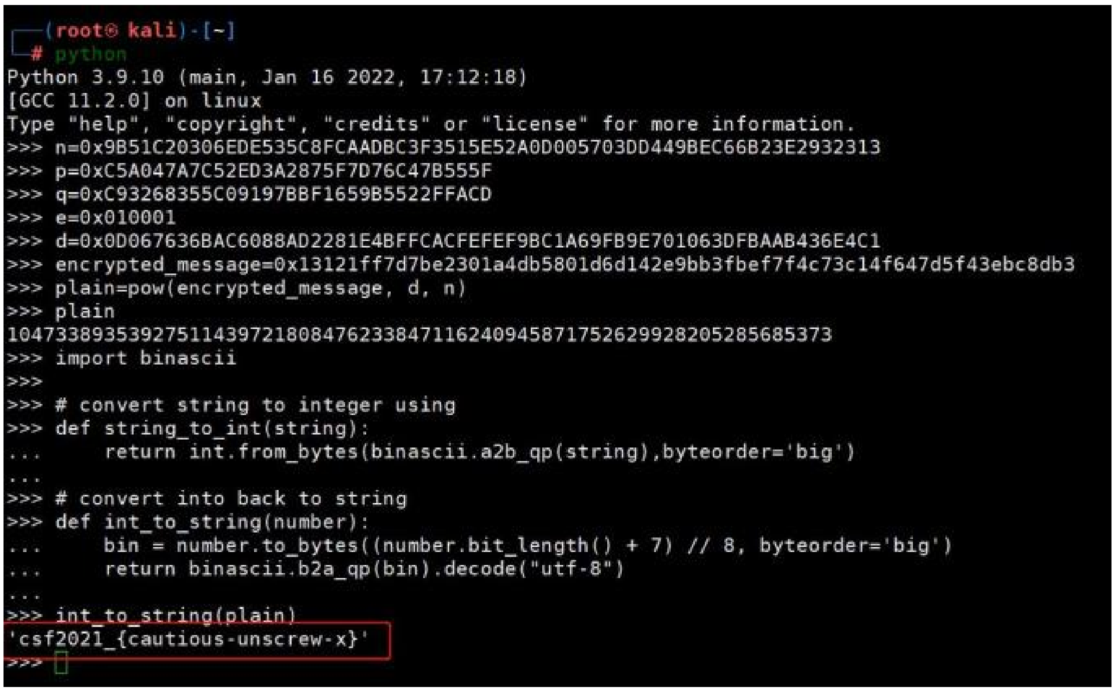


```
┌──(root㉿kali)-[~/…/s1/cyber/assginment01/Q11]
└─# cat encrypt.py
#!/usr/bin/python3

import argparse
import os
import sys
import random

def mycrypto (filename):
    with open(filename, 'r') as f, open(filename + '.enc', 'w') as o:
        blob = f.read()
        for b in blob:
            key = random.randrange(255)
            x = ord(b) ^ key
            o.write(chr(x))

def main():
    parser = argparse.ArgumentParser(description='Encrypt (?) a file')
    parser.add_argument('filename', metavar='filename', type=str, help='file to encrypt')
    parser.add_argument('--seed', metavar='seed',type=int,default=2021,help='seed')
    args = parser.parse_args()

    if not os.path.isfile(args.filename):
        print('The file  does not exist')
        sys.exit()

    random.seed(args.seed)
    mycrypto(args.filename)

if __name__== "__main__":
    main()

┌──(root㉿kali)-[~/…/s1/cyber/assginment01/Q11]
└─# python2 encrypt.py secret2021-1.enc

┌──(root㉿kali)-[~/…/s1/cyber/assginment01/Q11]
└─# cat secret2021-1.enc.enc
 ________________________________________
/ I fell asleep reading a dull book, and \
| I dreamt that I was reading on, so I   |
\ woke up from sheer boredom.            /
 ----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

Q12

The palintext: **csf2021_{cautious-unscrew-x}**

Through the encrypted script in the workshop I got the decryption method

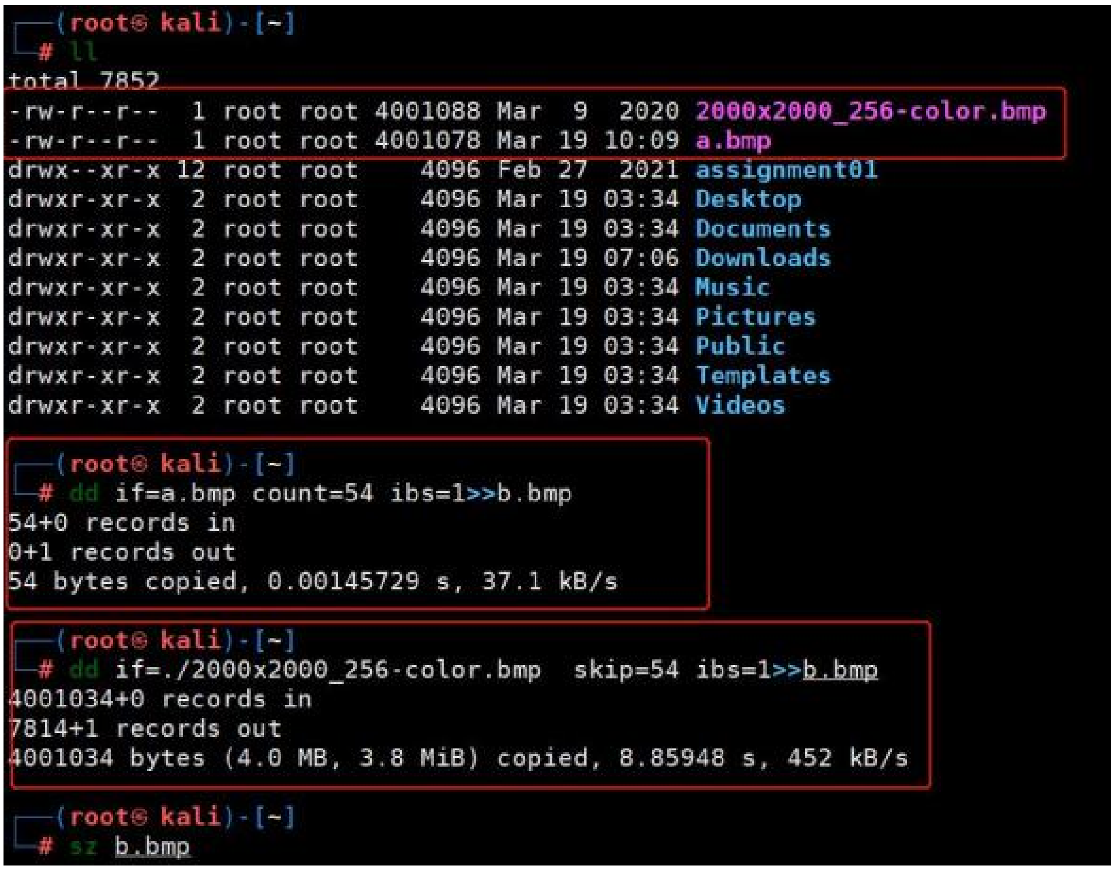


```
┌──(root㉿ kali)-[~]
└─# python
Python 3.9.10 (main, Jan 16 2022, 17:12:18)
[GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> n=0x9B51C20306EDE535C8FCAADBC3F3515E52A0D005703DD449BEC66B23E2932313
>>> p=0xC5A047A7C52ED3A2875F7D76C47B555F
>>> q=0xC93268355C09197BBF1659B5522FFACD
>>> e=0x010001
>>> d=0x0D067636BAC6088AD2281E4BFFCACFEFEF9BC1A69FB9E701063DFBAAB436E4C1
>>> encrypted_message=0x13121ff7d7be2301a4db5801d6d142e9bb3fbef7f4c73c14f647d5f43ebc8db3
>>> plain=pow(encrypted_message, d, n)
>>> plain
10473389353927511439721808476233847116240945871752629928205285685373
>>> import binascii
>>>
>>> # convert string to integer using
>>> def string_to_int(string):
...     return int.from_bytes(binascii.a2b_qp(string),byteorder='big')
...
>>> # convert into back to string
>>> def int_to_string(number):
...     bin = number.to_bytes((number.bit_length() + 7) // 8, byteorder='big')
...     return binascii.b2a_qp(bin).decode("utf-8")
...
>>> int_to_string(plain)
'csf2021_{cautious-unscrew-x}'
>>> 
```

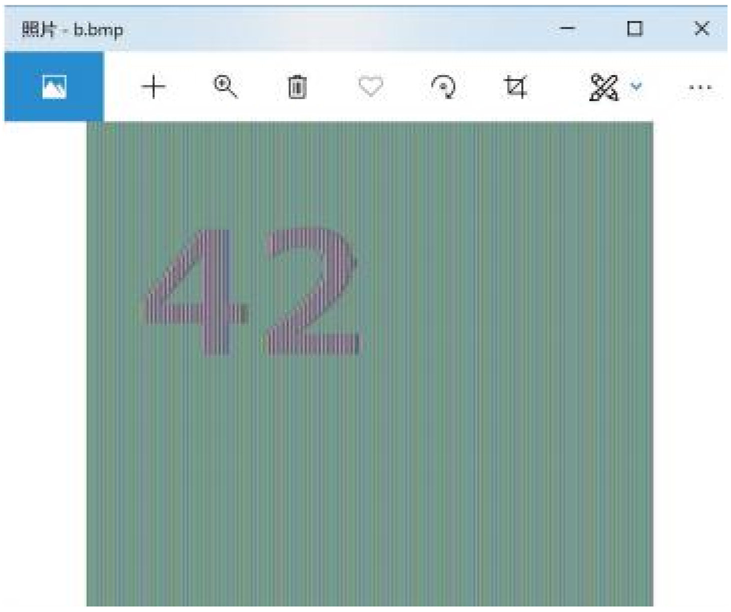
Q13

The message: **42**

According to the title, I first guessed that this bitmap was encrypted using EBC mode, and then I learned by understanding the bitmap format: the bitmap header (the first 54 bytes) records the bitmap size and color information, and the header information of different bitmaps may be different. So I first created an 2000x2000_256-color bitmap a .bmp through photoshop software, and then merged a .bmp header and encrypted bitmap into a new picture c .bmp via the dd command. Open c.bmp get the hidden information.



```
┌──(root㉿ kali)-[~]
└─# ll
total 7852
-rw-r--r--  1 root root 4001088 Mar  9  2020 2000x2000_256-color.bmp
-rw-r--r--  1 root root 4001078 Mar 19 10:09 a.bmp
drwx--xr-x 12 root root    4096 Feb 27  2021 assignment01
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Desktop
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Documents
drwxr-xr-x  2 root root    4096 Mar 19 07:06 Downloads
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Music
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Pictures
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Public
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Templates
drwxr-xr-x  2 root root    4096 Mar 19 03:34 Videos

┌──(root㉿ kali)-[~]
└─# dd if=a.bmp count=54 ibs=1>>b.bmp
54+0 records in
0+1 records out
54 bytes copied, 0.00145729 s, 37.1 kB/s

┌──(root㉿ kali)-[~]
└─# dd if=./2000x2000_256-color.bmp  skip=54 ibs=1>>b.bmp
4001034+0 records in
7814+1 records out
4001034 bytes (4.0 MB, 3.8 MiB) copied, 8.85948 s, 452 kB/s

┌──(root㉿ kali)-[~]
└─# sz b.bmp
```

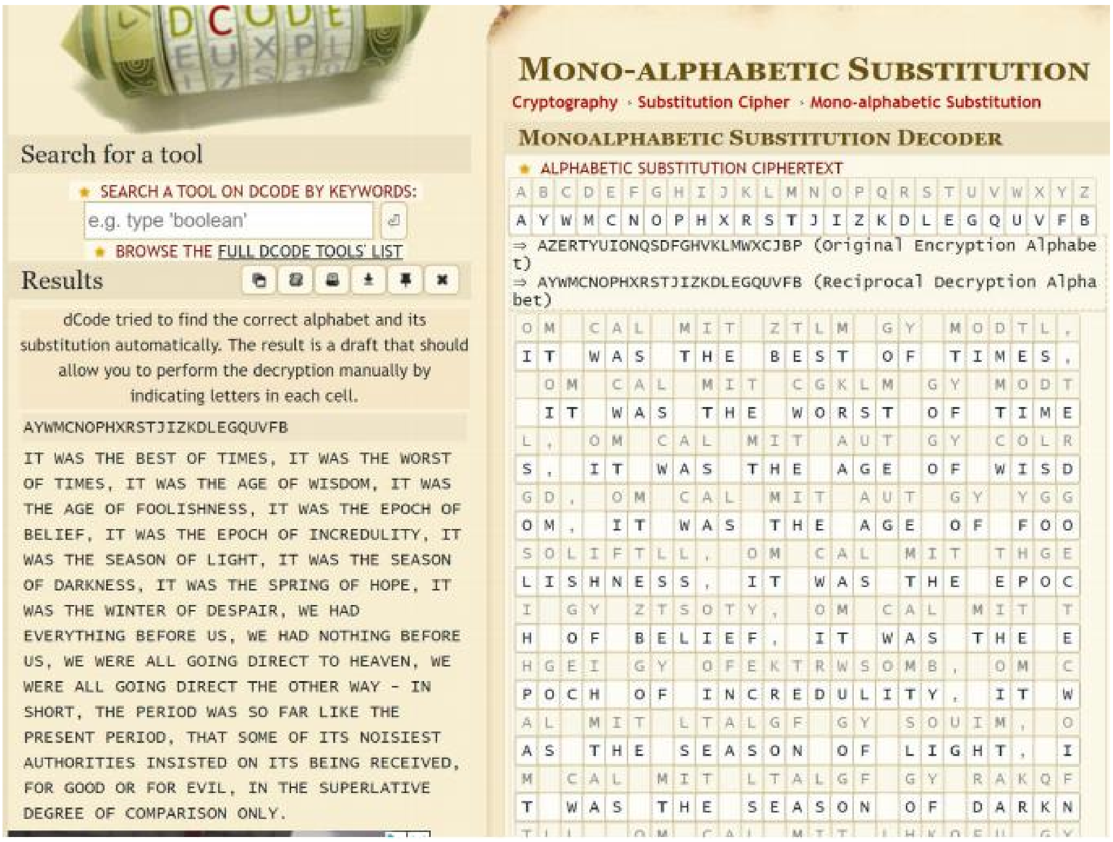


Q14

The cipher: **subsitution cipher**

The key: **AYWMCNOPHXRSTJIZKDLEGQUVFB**

Based on the information provided by the question, the first guess is that the Vigenere cipher or a subsitution cipher may be used. Secondly, observe that there are a large number of duplicate characters in the ciphertext, and the final guess should be that a substitution cipher should be used. Eventually get the ciphertext by hacking the website.



DCODE
Search for a tool
SEARCH A TOOL ON DCODE BY KEYWORDS:
e.g. type 'boolean'
BROWSE THE FULL DCODE TOOLS' LIST
Results
Cryptography › Substitution Cipher › Mono-alphabetic Substitution
ALPHABETIC SUBSTITUTION CIPHERTEXT

Mono-alphabetic Substitution

Monoalphabetic Substitution Decoder

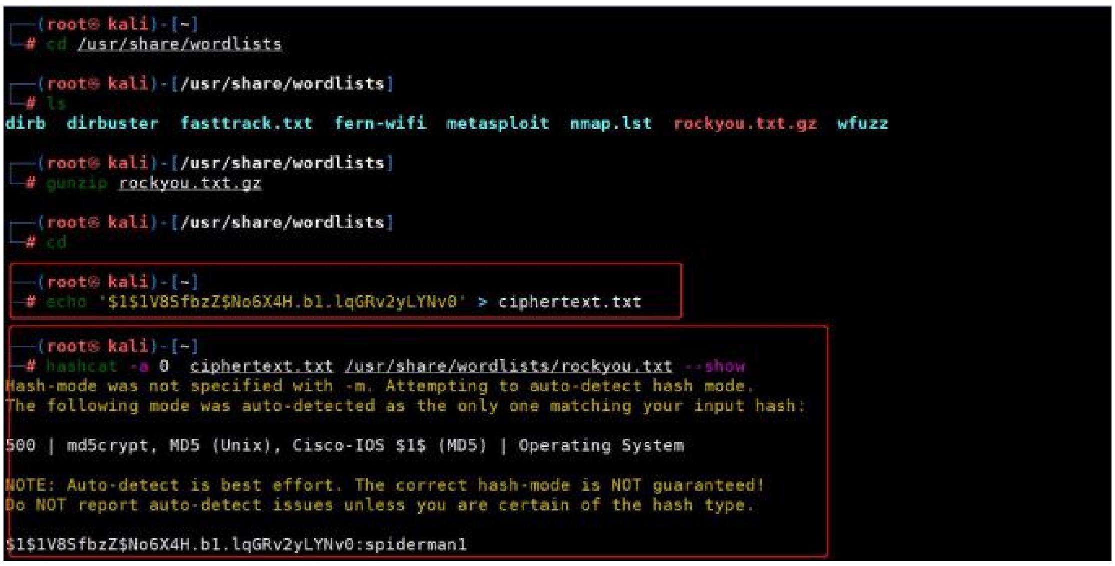
dCode tried to find the correct alphabet and its substitution automatically. The result is a draft that should allow you to perform the decryption manually by indicating letters in each cell.

```
AYWMCNOPHXRSTJIZKDLEGQUVFB

IT WAS THE BEST OF TIMES, IT WAS THE WORST
OF TIMES, IT WAS THE AGE OF WISDOM, IT WAS
THE AGE OF FOOLISHNESS, IT WAS THE EPOCH OF
BELIEF, IT WAS THE EPOCH OF INCREDULITY, IT
WAS THE SEASON OF LIGHT, IT WAS THE SEASON
OF DARKNESS, IT WAS THE SPRING OF HOPE, IT
WAS THE WINTER OF DESPAIR, WE HAD
EVERYTHING BEFORE US, WE HAD NOTHING BEFORE
US, WE WERE ALL GOING DIRECT TO HEAVEN, WE
WERE ALL GOING DIRECT THE OTHER WAY - IN
SHORT, THE PERIOD WAS SO FAR LIKE THE
PRESENT PERIOD, THAT SOME OF ITS NOISIEST
AUTHORITIES INSISTED ON ITS BEING RECEIVED,
FOR GOOD OR FOR EVIL, IN THE SUPERLATIVE
DEGREE OF COMPARISON ONLY.
```

```
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
A Y W M C N O P H X R S T J I Z K D L E G Q U V F B
⇒ AZERTYUIONQSDFGHVKLMWXCJBP (Original Encryption Alphabe
t)
⇒ AYWMCNOPHXRSTJIZKDLEGQUVFB (Reciprocal Decryption Alpha
bet)

OM CAL MIT ZTLM GY MODTL,
IT WAS THE BEST OF TIMES,
 OM CAL MIT CGKLM GY MODT
 IT WAS THE WORST OF TIME
L, OM CAL MIT AUT GY COLR
S, IT WAS THE AGE OF WISD
GD, OM CAL MIT AUT GY YGG
OM, IT WAS THE AGE OF FOO
SOLIFTLL, OM CAL MIT THGE
LISHNESS, IT WAS THE EPOC
I GY ZTSOTY, OM CAL MIT T
H OF BELIEF, IT WAS THE E
HGEI GY OFEKTRWSOMB, OM C
POCH OF INCREDULITY, IT W
AL MIT LTALGF GY SOUIM, O
AS THE SEASON OF LIGHT, I
M CAL MIT LTALGF GY RAKQF
T WAS THE SEASON OF DARKN
```

Q15

The password: **spiderman1**



```
┌──(root㉿ kali)-[~]
└─# cd /usr/share/wordlists

┌──(root㉿ kali)-[/usr/share/wordlists]
└─# ls
dirb  dirbuster  fasttrack.txt  fern-wifi  metasploit  nmap.lst  rockyou.txt.gz  wfuzz

┌──(root㉿ kali)-[/usr/share/wordlists]
└─# gunzip rockyou.txt.gz

┌──(root㉿ kali)-[/usr/share/wordlists]
└─# cd

┌──(root㉿ kali)-[~]
└─# echo '$1$1V8SfbzZ$No6X4H.b1.lqGRv2yLYNv0' > ciphertext.txt

┌──(root㉿ kali)-[~]
└─# hashcat -a 0  ciphertext.txt /usr/share/wordlists/rockyou.txt --show
Hash-mode was not specified with -m. Attempting to auto-detect hash mode.
The following mode was auto-detected as the only one matching your input hash:

500 | md5crypt, MD5 (Unix), Cisco-IOS $1$ (MD5) | Operating System

NOTE: Auto-detect is best effort. The correct hash-mode is NOT guaranteed!
Do NOT report auto-detect issues unless you are certain of the hash type.

$1$1V8SfbzZ$No6X4H.b1.lqGRv2yLYNv0:spiderman1
```