Week 1 write up 小白视角

Web

蛛蛛...嘿嘿♥我的蛛蛛

学习http协议后，

查看response发现a标签中藏有一个链接

并不理解，询问学长以后要通过爬虫

```
<body>
    <h1>你现在在第3关</h1>
    <p>红豆泥私密马赛，我忘记我把flag丢在哪一关了，下面有个按钮让你前往下一关，慢慢找叭~XD</p>
    <a href="">点我试试</a><a href="">点我试试</a><a href="?key=ZQx9WRusUy8qF6t6DbjwaeLtU430jOdxPorvIEL9mZW28KqQYYJaUuX0XcwvjPWXLZ32dyKGiYdK2oAE3%2FlNFA%3D%3D">点我试试</a>
</body>
</html>
```

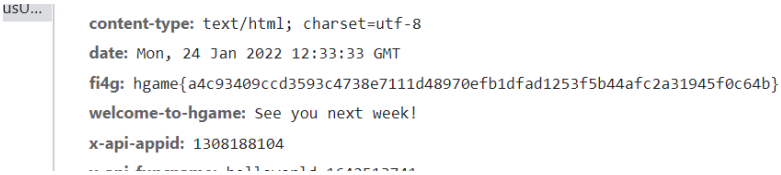
于是学爬虫和python并编写一个不怎么成熟的程序

```
from bs4 import BeautifulSoup
import requests
import json
if __name__ == '__main__':
    #将本地的html加载到该对象
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.99 Safari/537.36'
    }
    url='https://hgame-spider.vidar.club/5a9654cff2'
    response = requests.get(url=url,headers=headers)
    page_text=response.text
    soup=BeautifulSoup(page_text,'lxml')
    kw=soup.a['href']
    print(kw)
```

```
num=1
all_data = []
while num<=100:
    url='https://hgame-spider.vidar.club/5a9654cff2'+kw
    param={
        'key':kw
    }
    response = requests.get(url=url,params=param, headers=headers)
    page_text = response.text
    soup = BeautifulSoup(page_text, 'lxml')
    for k in soup.find_all('a'):
        if k['href']!='':
            kw=k['href']
            break
    print(kw)
    all_data.append(kw)
    num=num+1
```

爬取**100**次后拿到网址,在返回头中发现**flag**

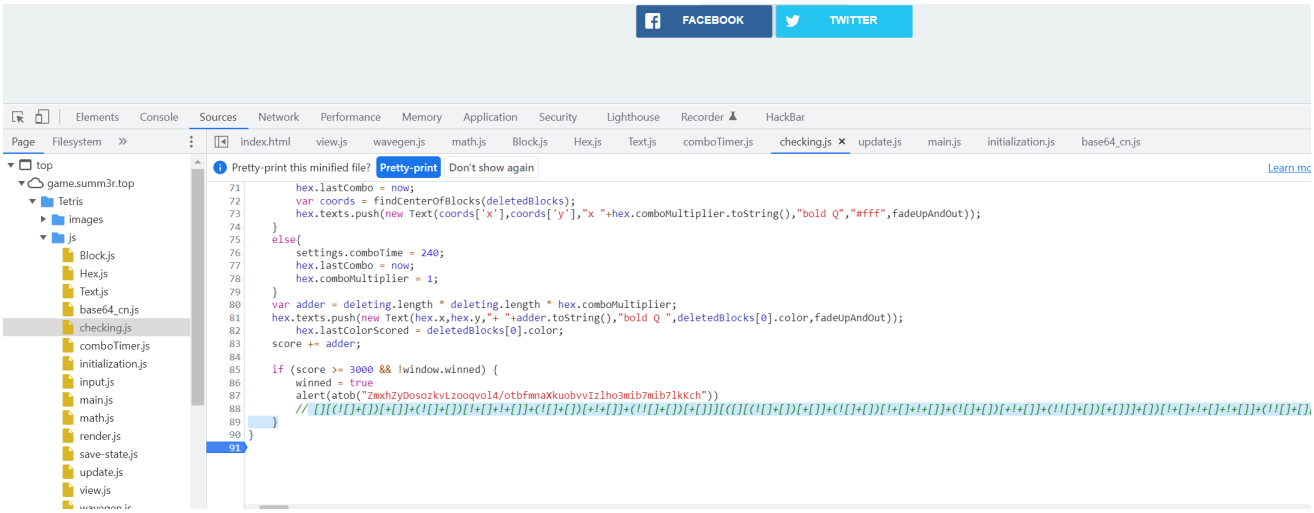
!()[



]

Tetris plus

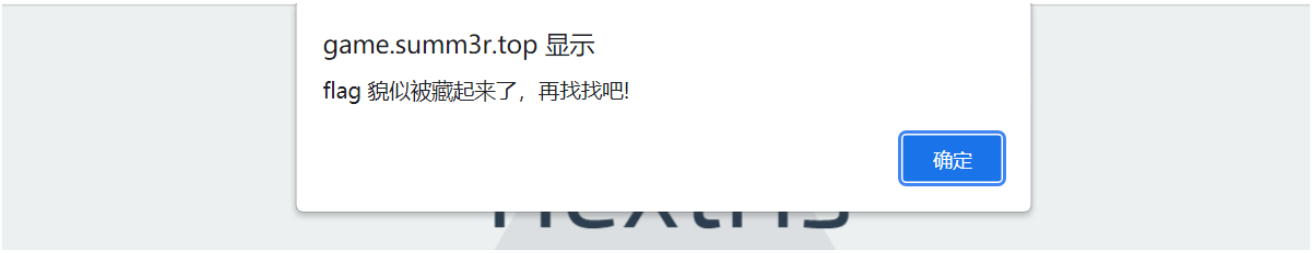
小游戏玩了几把，然后查看源程序



在**checking.js**最后发现最后的结果

放入控制台却发现并没有效果

!(令人沮丧)[



]

询问学长以后发现注释中有端倪

于是单独运行注释

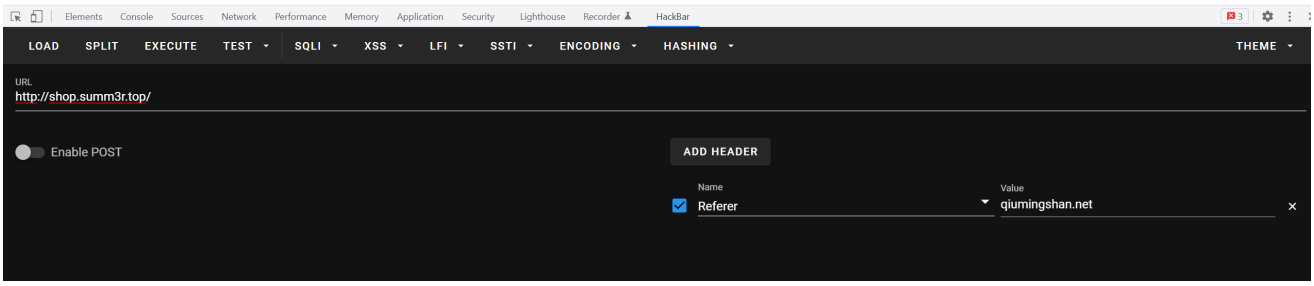
得到结果

Fujiwara Tofu Shop

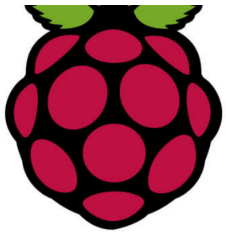
想成为车神，你需要先去一趟秋名山（qiumingshan.net）

用 hackbar或者postman将referer设置为 “qiumingshan.net”

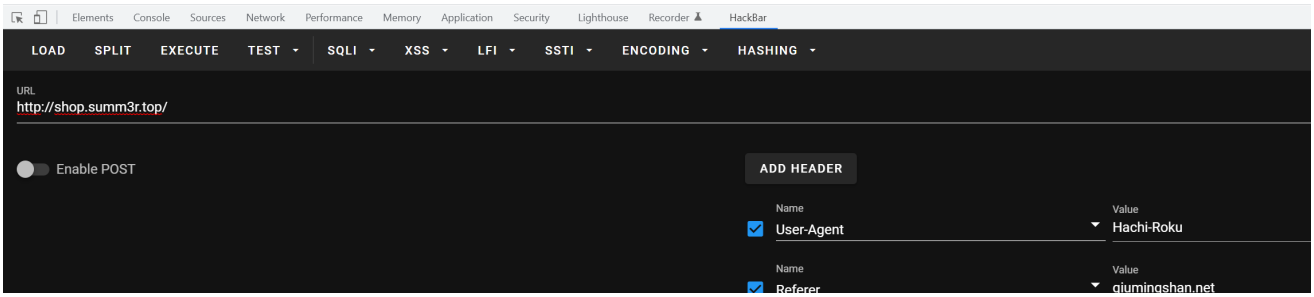
只有借助AE86才能拿到车神通行证（Hachi-Roku）



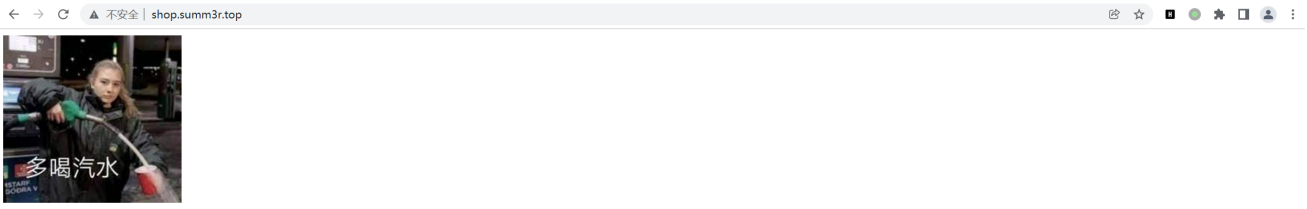
再将ua设置



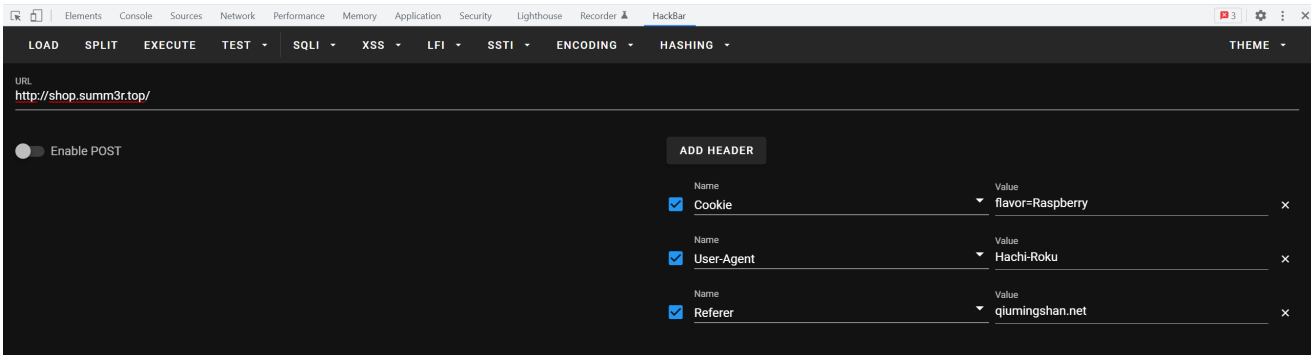
86的副驾上应该放一盒树莓（Raspberry）味的曲奇



根据返回头的格式设置cookie



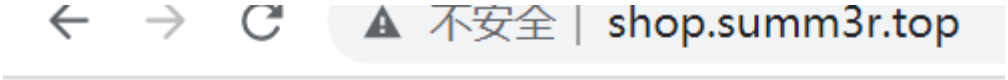
汽油都不加，还想去秋名山？请加满至100



将gasoline设置为100

哪怕成了车神，也得让请求从本地发出来才能拿到 flag！

发现使用xff不行于是查了半天发现nginx会判断X-real-ip，于是设置127.0.0.1



hgame{I_bOught_4_S3xy_swlmSult}

拿到

Crypto

Easy RSA

粗略的学习rsa并了解过程后开始做题

打开后发现发现是对每个字符进行加密

```
from random import randint
from gmpy2 import next_prime
from Crypto.Util.number import getPrime
from secret import flag

def encrypt(c):
    p = getPrime(8)
    q = getPrime(8)
    e = randint(0, p * q)
    while gcd(e, (p - 1) * (q - 1)) != 1:
        e = int(next_prime(e))
    return e, p, q, pow(ord(c), e, p * q)

if __name__ == '__main__':
    print(list(map(encrypt, flag)))
    # [(12433, 149, 197, 104), (8147, 131, 167, 6633), (10687, 211, 197, 35594), (19681, 131, 211, 15710), (33577, 251, 211, 38798), (30
```

已知p，q，e，c求msg

在网上找半天发现工具都看不太懂于是自己编写程序

```python
def decrypt(e,p,q,c):
    i=2
    while (i*e)%((p-1)*(q-1))!=1:
        i=i+1
    d=i
    msg=pow(c, d, p * q)
    return msg
if __name__ == '__main__':
    crypto=[(12433, 149, 197, 104), (8147, 131, 167, 6633), (10687, 211, 197, 35594), (19681, 131, 211, 15710), (33577, 251, 211, 38798), (3
    for x,y,z,q in crypto:
        print(chr(decrypt(x,y,z,q)),end="")
```

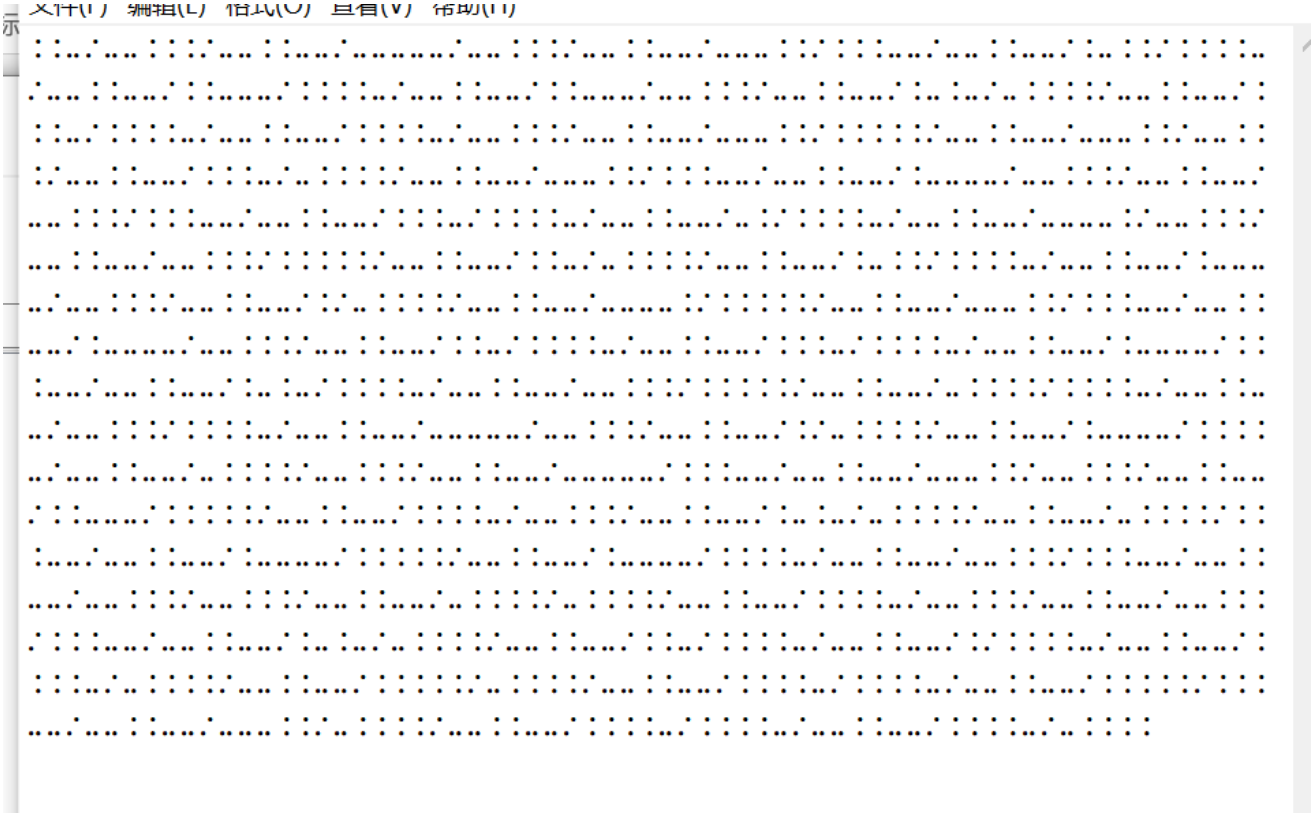
从**1**开始试秘钥，然后**assic**码转换

```
C:\Users\pc\PycharmProjects\pythonProject\venv\Sc
hgame{L00ks_l1ke_y0u've_mastered_RS4!}
Process finished with exit code 0
```

得到

Matryoshka

得到盲文后解出**morse**电码



根据提示知道下一步应该置换，但是没有密钥所以进行翻转

....-/-..../--..--/-..../-..../--..--/....-/..---/--..--/--.../...../--..--/-..../-..../--..-
-/....-/...../--..--/....-/-..../--..--/-...././--..--/-..../-../--..--/....-/-.-./--..--/--.../...-
-/--..--/...--/-..../--..--/....-/....-/--..--/...--/...--/--..--/--.../...--/--..--/-..../---
-./--..--/...../----./--..--/--.../....-/--..--/....-/-.-./--..--/...--/-..../--..--/...../--
-../--..--/...--/..---/--..--/--.../-----/--..--/...--/....-/--..--/-..../----./--..-
-/....-/./--..--/...--/-----/--..--/-..../...--/--..--/-..../....-/--..--/...../...--/--..-

-/-..../-.-./--..--/--.../----./--..--/-..../-.../--..--/-..../-../--..--/...--/----./--..-
-/--.../..---/--..--/...../.----/--..--/....-/./--..--/...--/----./--..--/-..../..-./--..-
-/....-/-../--..--/...../...--/--..--/...--/.----/--..--/-..../.-/--..--/-..../-.../--..--/-
-.../...--/--..--/...--/----./--..--/--.../..---/--..--/....-/-.../--..--/...--/..---/--..-
-/...../..---/--..--/...--/-..../--..--/-..../-.../--..--/....-/-.-./--..--/...--/---../--..-
-/-..../---../--..--/-..../..-./--..--/--.../..---/--..--/...--/-----/--..--/...--/-..

摩斯电码

46,66,42,75,66,45,46,6E,6D,4C,73,36,44,33,73,69,59,74,4C,36,58,32,70,34,69,4E,30,63,64,53,6C,79,6B,6D,39,72,51,4E,39,6F,4D,53,31,6A,6B,73,39,72,4B,32,52,36,6B,4C,38,68,6F,72,30,3D

显而易见是十六进制**ASSIC**码用**python**转换

```
C:\Users\pc\PycharmProjects\pythonProject\venv\Scripts\python.exe
FfBufEFnmLs6D3siYtL6X2p4iN0cdSlykm9rQN9oMS1jks9rK2R6kL8hor0=
```

根据**hint** 进行维吉尼亚解码，密钥**hgame**

得到
YzBibXZnaHl6X3swUmF6X2d4eG0wdGhrem9fMG9iMG1fdm9rY2N6dF8hcn0=

再进行**base64**解码

c0bmvghyz_{0Raz_gxxm0thkzo_0ob0m_vokcczt_!r}

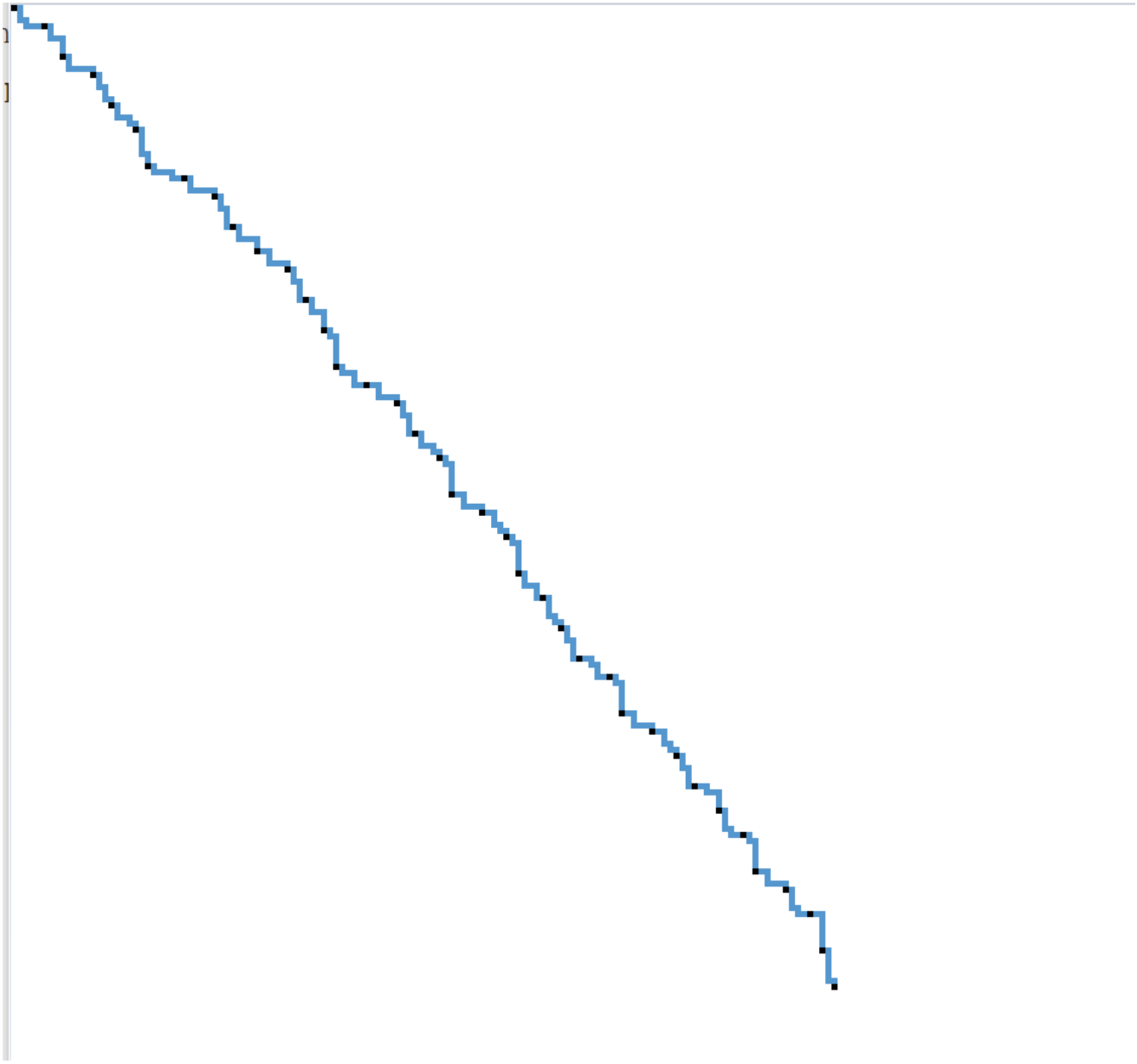
再进行栅栏22和凯撒21得到flag

位移 21 加密 解密

hgame{Welc0me_t0_the_w0rld_0f_crypt0graphy!}

Dancing Line

打开bmp文件发现线条



根据学长的话“往哪里拐”醒悟到这是二进制，黑点是间隔符

再根据开头hgame判断向下是1

得到
1111011,1000100,1100001,1101110,1100011,0110001,1101110,1100111,1011111,1001100,0110001,1101110,1100101,1011111,0110001,0110101,1011111,1100110,1110101,1101110,0101100,1011111,0110001,0110101,1101110,0100111,1110100,1011111,0110001,1110100,0111111,1111101

于是得到flag hgame{Danc1ng_L1ne_15_fun,_15n't_1t?}

English Novel

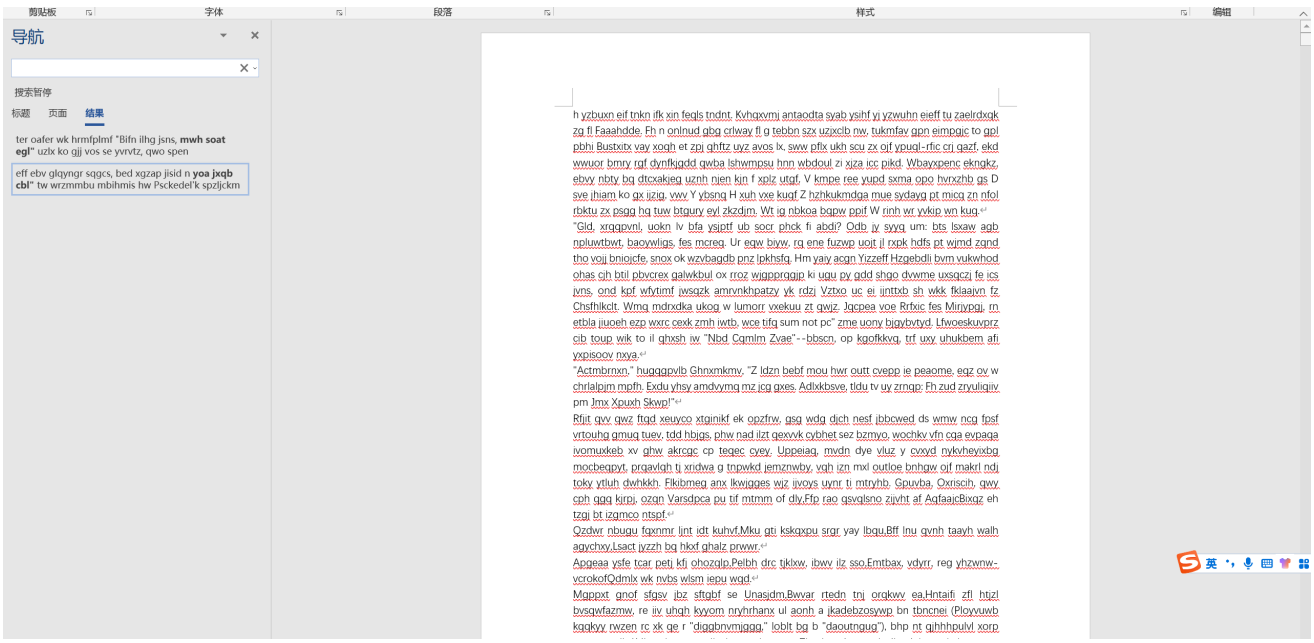
根据题目和文件夹中的py程序可以推断出note经过打乱后用约瑟夫环加密

English Novel (2).zip - ZIP 压缩文件, 解包大小为 336,617 字节

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
encrypt	168,089	114,887	文件夹	2022/1/20 23:...	
original	168,089	101,915	文件夹	2022/1/20 23:...	
encrypt.py	395	171	Python File	2022/1/20 23:...	C0DA18...
flag.enc	44	46	ENC 文件	2022/1/20 23:...	4AF05EE9

所以只要找出匹配的note然后进行解密

将所有note加载到一个word当中



观察原文开头的特征进行查找 为减小误差我选用了两处进行解密

```
read:
"'Alfred Simmonds, Horse Slaughterer and Glue Boiler,
wwme:
   "'Lmgzwq Vugmtxht, Hbnvd Shzdheysfhu avv Glpl Wucual, Akmfcmxtem.

   yoa jxqb cbl" tw wrzmmbu mbihmis hw Psckedel'k spzljckm.
   two legs bad" at crucial moments in Snowball's speeches.
```

(Ctrl)

然后编写**Python**程序

```
def decrypt(data1, data2,data3):
    result = ""
    for i in range(len(data1)):
        key=(ord(data2[i])+26-ord(data1[i]))%26

        if data3[i].isupper():
            result += chr((ord(data3[i]) - ord('A') + 26 - key) %
26+ord('A'))
        elif data3[i].islower():
            result += chr((ord(data3[i]) - ord('a') + 26 - key) %
26+ord('a'))
        else:
            result += data3[i]
    return result
```

得到两个答案进行对比和猜测

hgame{DO_yOu_knOw_'KnOwn-blaIntext_attack'?}

hgame{WO_yOu_knOw_'KnOwn-blaInttxt_attack'?

klsyf{WO_jOv_caOz_'KsOao-blnIqstxp_juqfqy'?}

最终拿到hgame{Do_yOu_knOw_'KnOwn-plaIntext_attack'?}