

CheatEngine-实验吧CTF题库writeup

原创

[0_Re5et](#) 于 2016-08-31 23:16:47 发布 5596 收藏 2

分类专栏: [网络安全 数据结构](#) 文章标签: [writeup CTF](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/kanamisama0/article/details/52390078>

版权



[网络安全](#) 同时被 2 个专栏收录

1 篇文章 0 订阅

订阅专栏



[数据结构](#)

15 篇文章 1 订阅

订阅专栏

题目网址: <http://www.shiyanbar.com/ctf/1930>

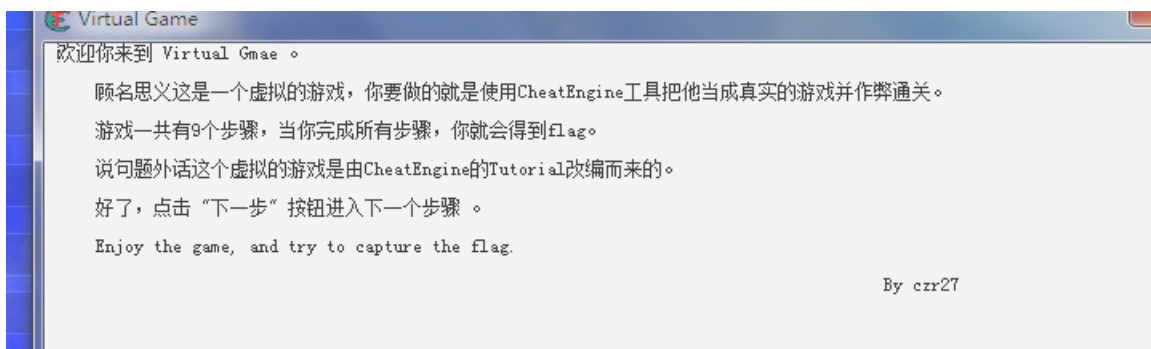
题目要求: 这是一个虚拟的游戏, 你要做的就是使用CheatEngine工具把他当成真实的游戏并作弊通关。

wp:

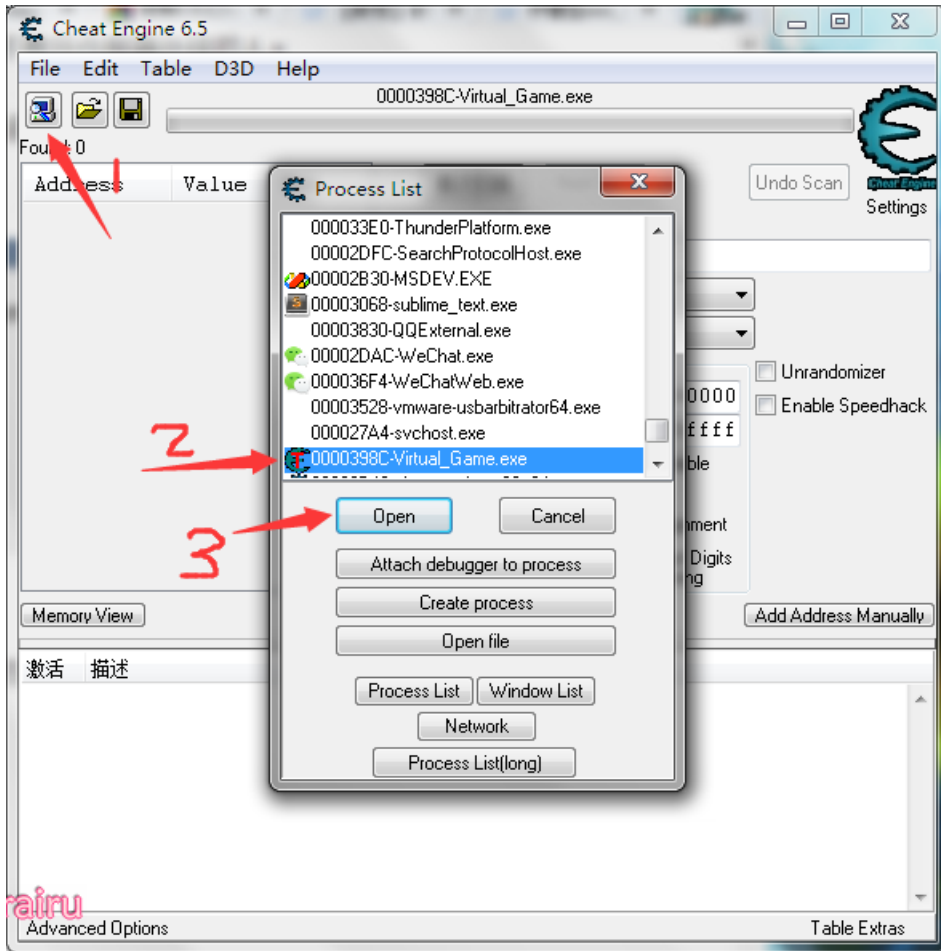
STEP1: 欢迎界面

之前我是从来没听说过CE这个软件的, 我一直是那种玩游戏都不敢开作弊器外挂的人x

这次去网上下载了CE, 先找了CE自带的游戏攻略跟着做完了一遍, 然后才做的这个题目, 其实步骤都是一样的。

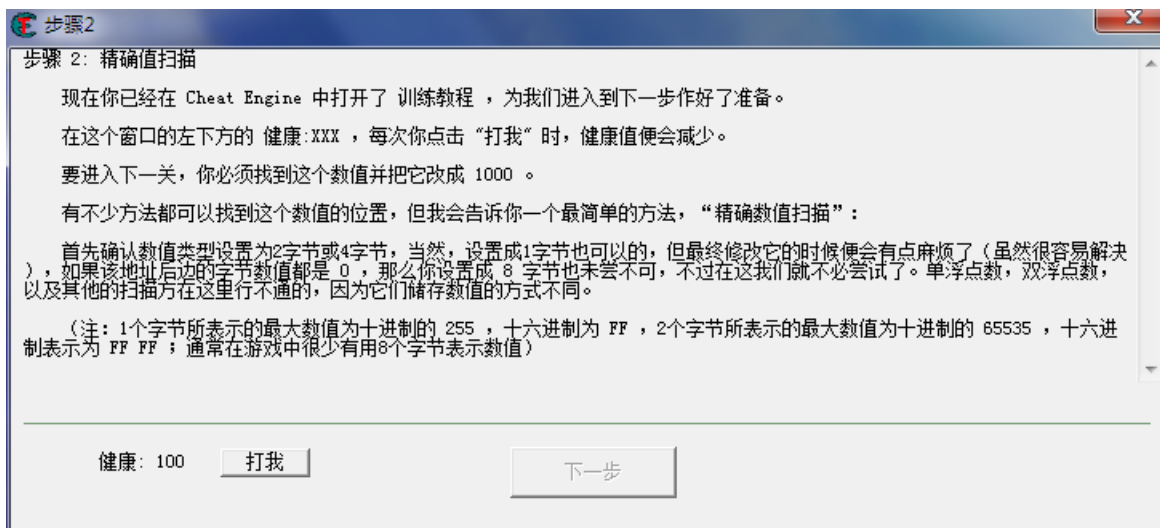


然后我们把CE打开, 用左上角的按钮打开这个游戏, 以便之后的修改



开始闯关。

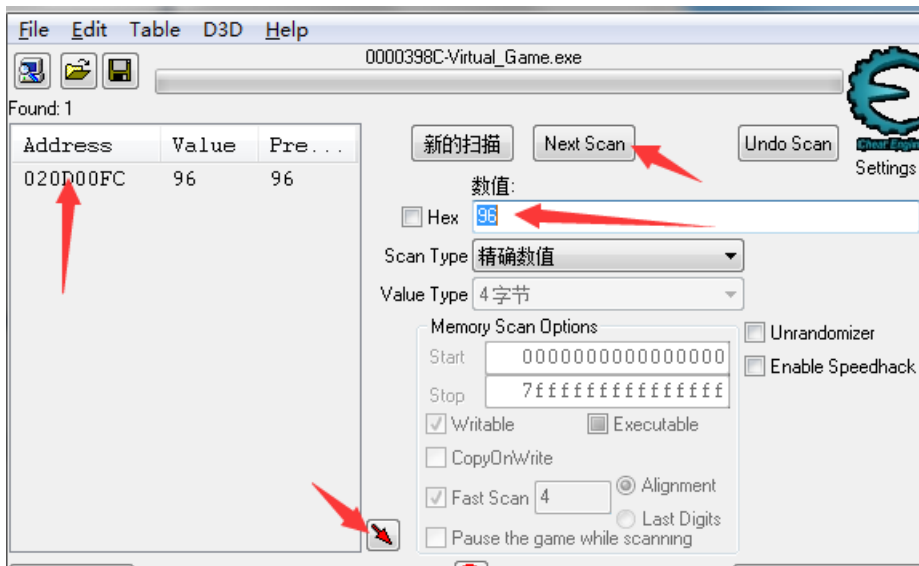
STEP2: 精确值扫描:



输入100，点击首次查询后按钮变成新的查询



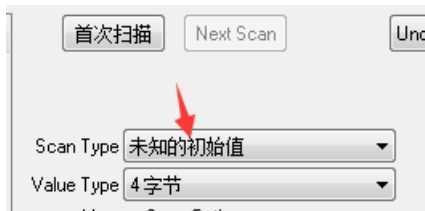
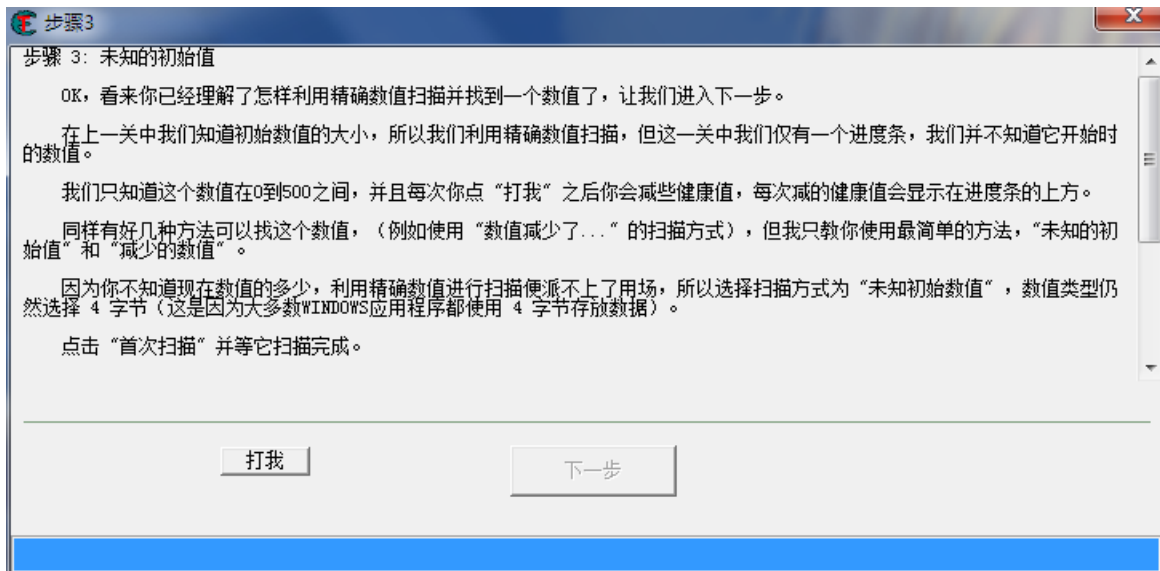
点击游戏里的打我，然后看值的改变，把改变后的值填好，点继续搜索。



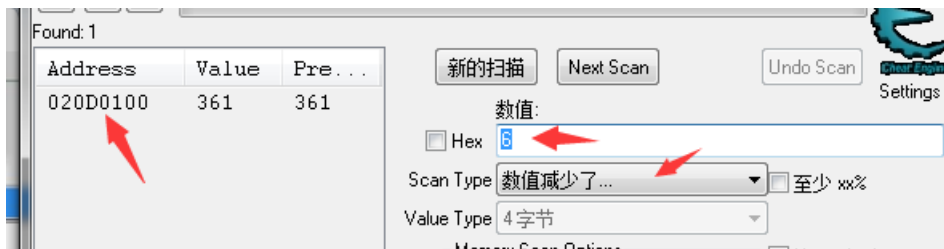
然后点中间下方的小箭头，添加到下面。把值修改为1000即可进入下一关。

激活	描述	地址	类型	数值
<input type="checkbox"/>	无描述	020D00FC	4字节	1000

STEP3: 未知的初始值

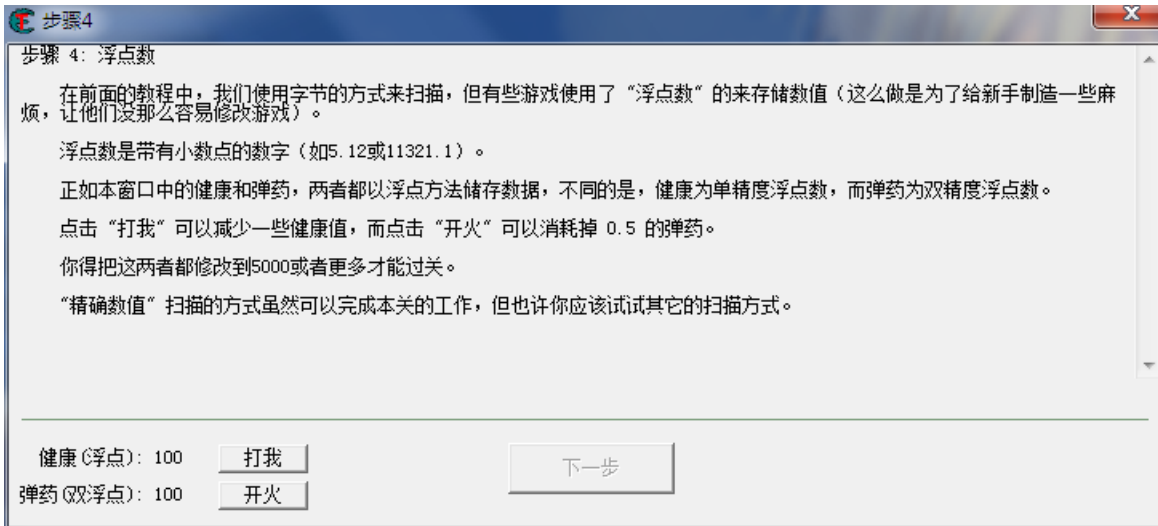


因初始值未知, 在此选择未知的初始值, 之后在游戏中选择打我, 提示减少值, 在继续扫描中选择减少了, 并填写减少值

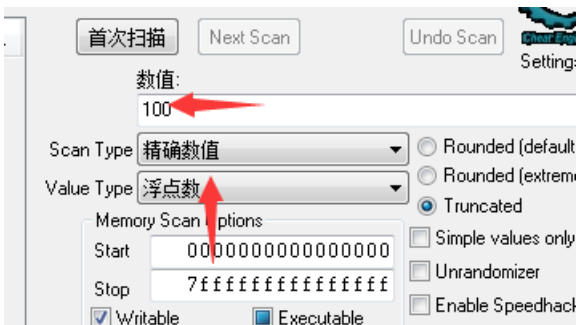


找到了地址, 添加到下方修改即可。

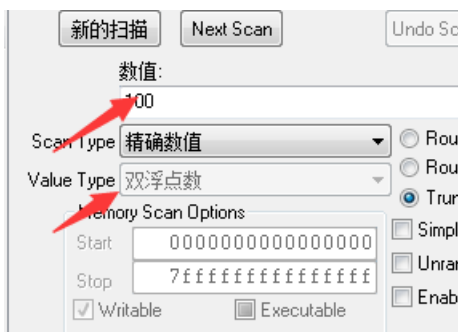
STEP4: 浮点数



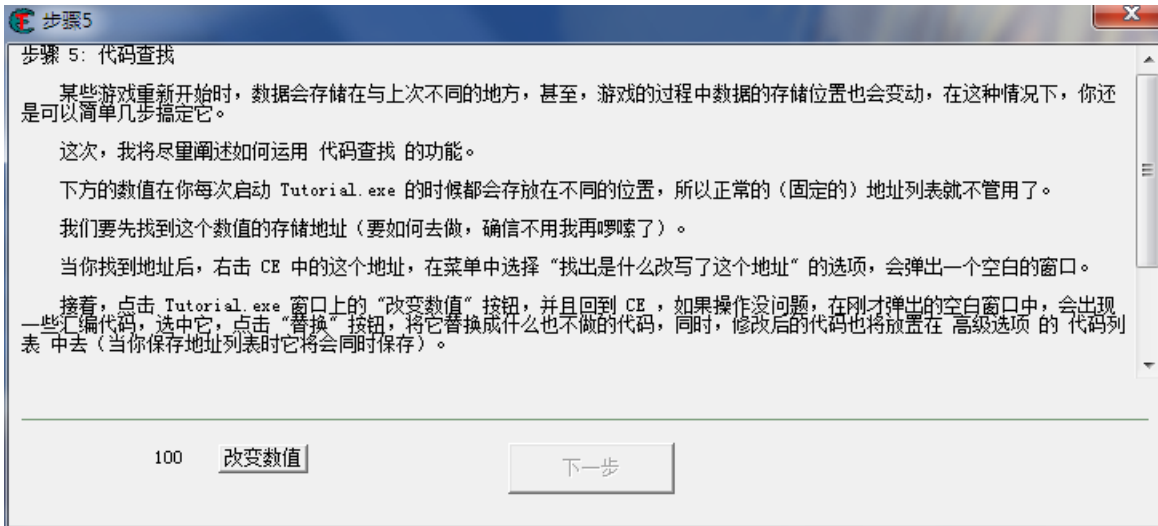
首先查找健康值的地址，题目提示为浮点数，按下图设置后点击首次扫描，再按照第一关的方式点击“打我”按钮来改变健康值从而寻找到地址。



修改火药数方法类似，两值添加到下方后把值改为5000即可过关。



STEP5: 代码查找

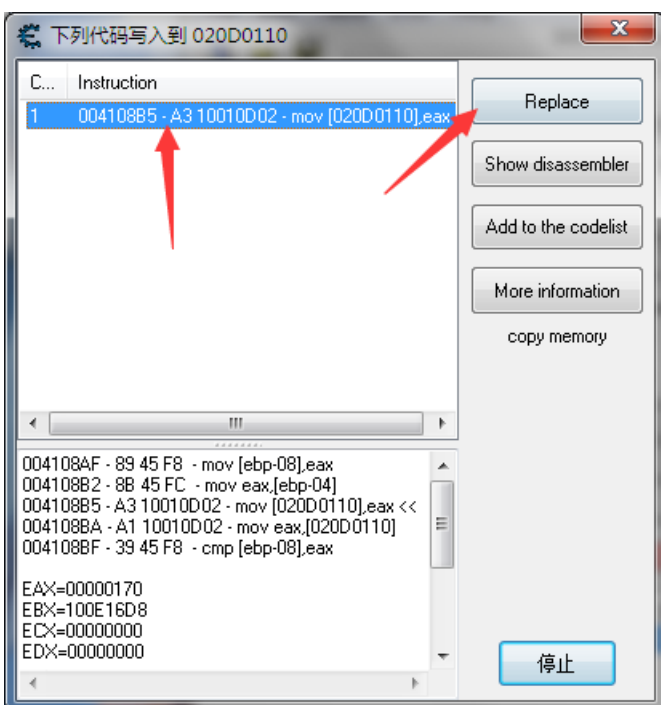


在查找100这个数值时记得把查找类型从双精度型改回4字节

数值	描述	地址	类型	数值
	无描述	020D0110	4字节	551

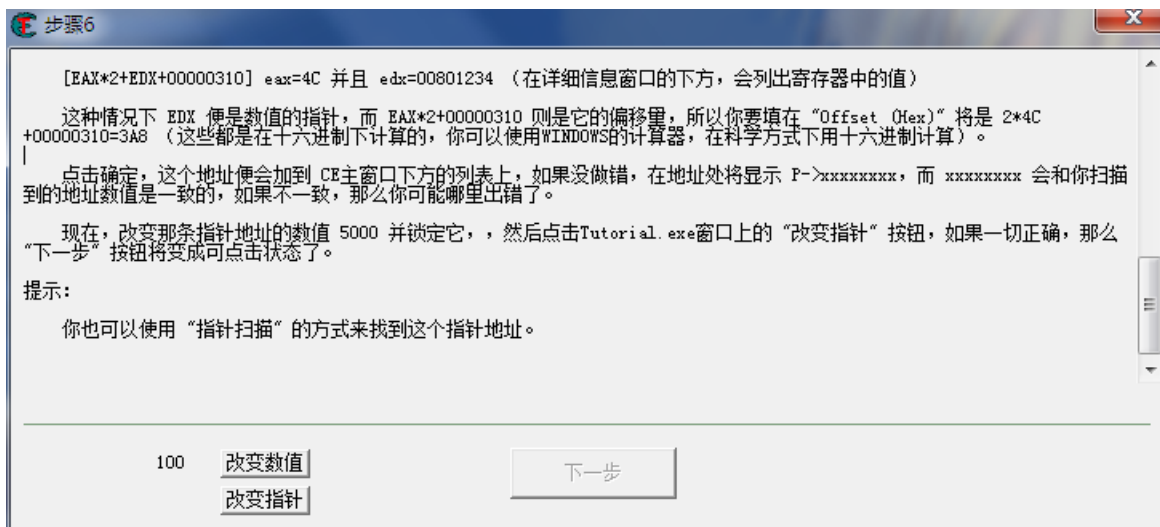
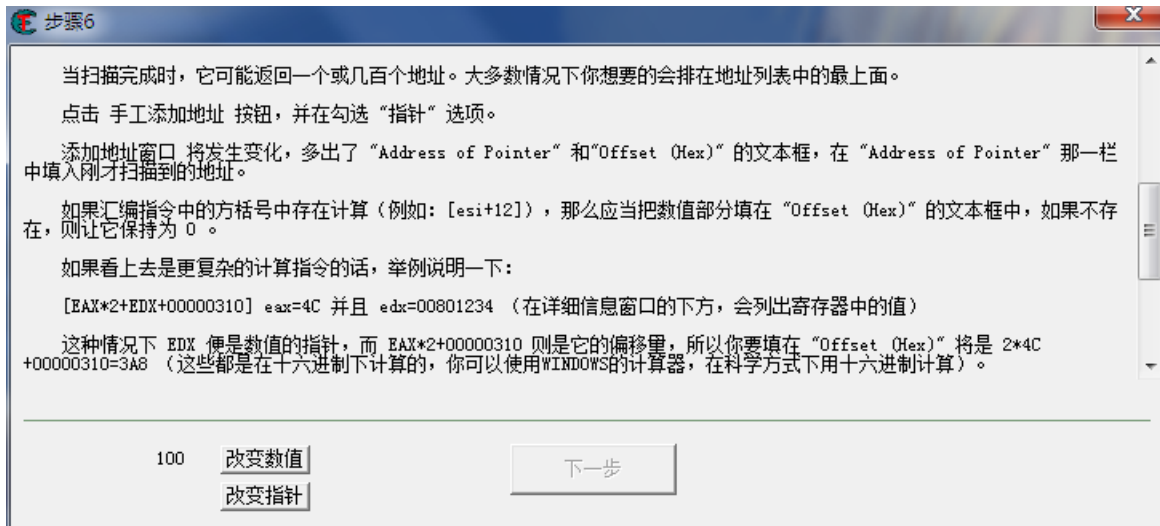
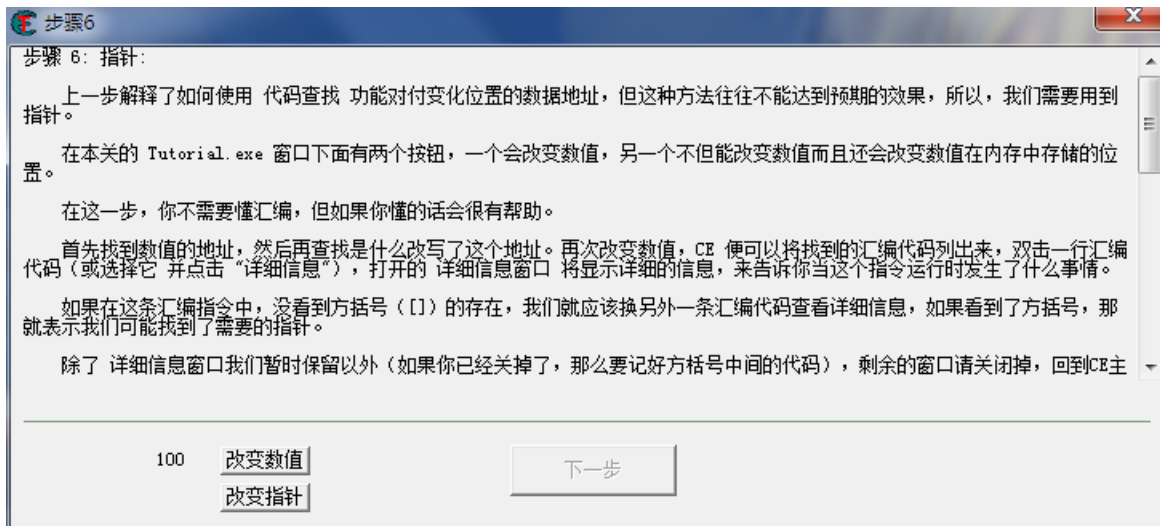
在下方选中这个地址，然后右键“find out what writes to this address”，即“找出是什么改写了这个地址”，我这里电脑截图有问题，截不到右键的东西，点击后弹出一个对话框，问是否使用ce调试器加载当前进程，选择是。

此时出现一个新的对话框，我们需要通过改变值来找到访问了这个地址的代码，所以点击游戏里的“改变数值”，然后对话框会显示出访问了这个地址的信息。如图

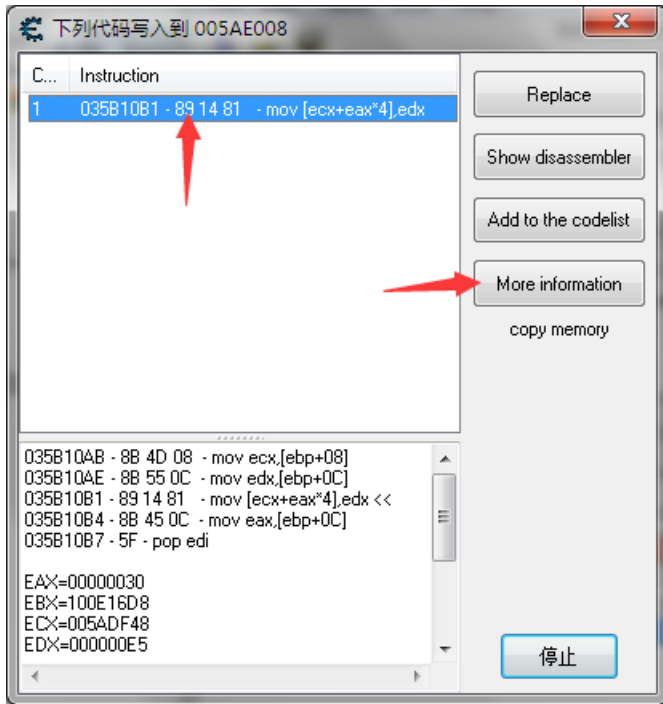


选中地址后点击右侧的replace，按题目要求把原来代码删除，改为nop，即什么都不做，然后点击右下方的停止使游戏继续下去即可。

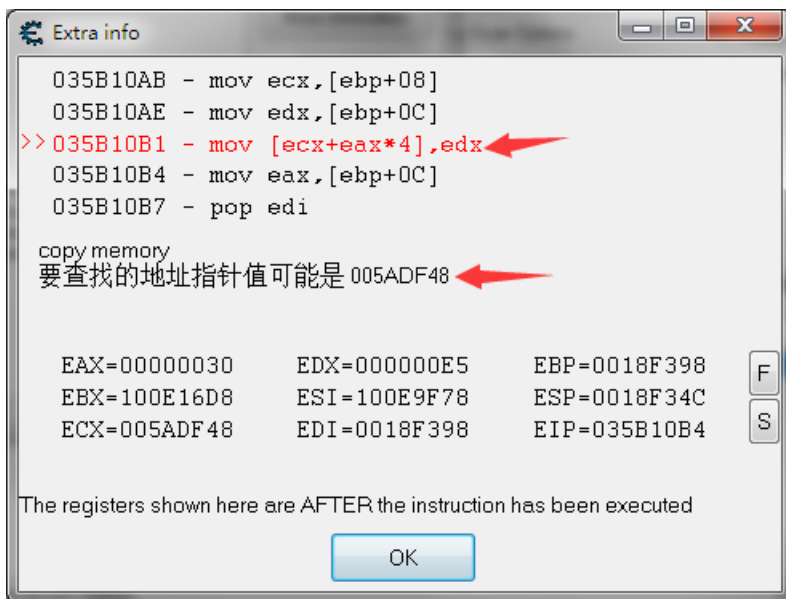
STEP6: 指针



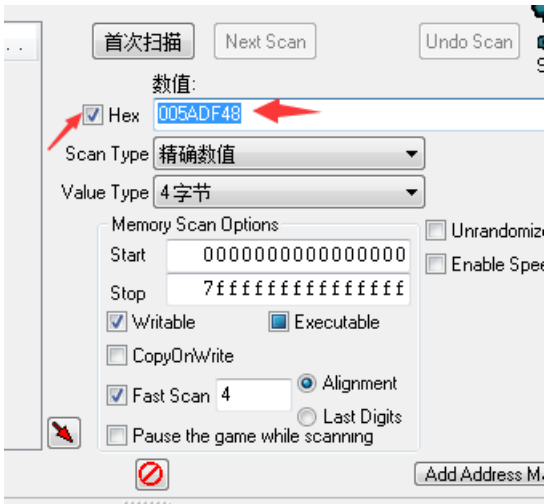
题目要求略长。首先我们按照前面的方式寻找到数值的地址添加到下面,然后按照上一步的方式启动调试器。



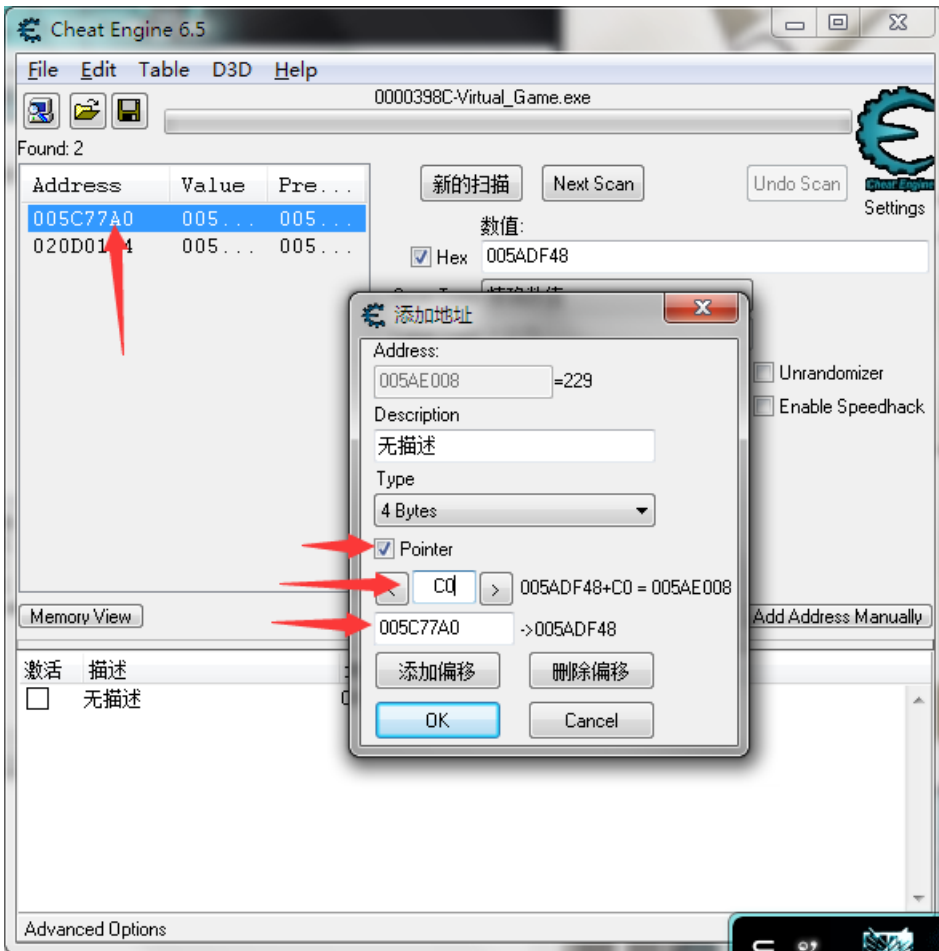
然后按照题目要求点击详细信息。



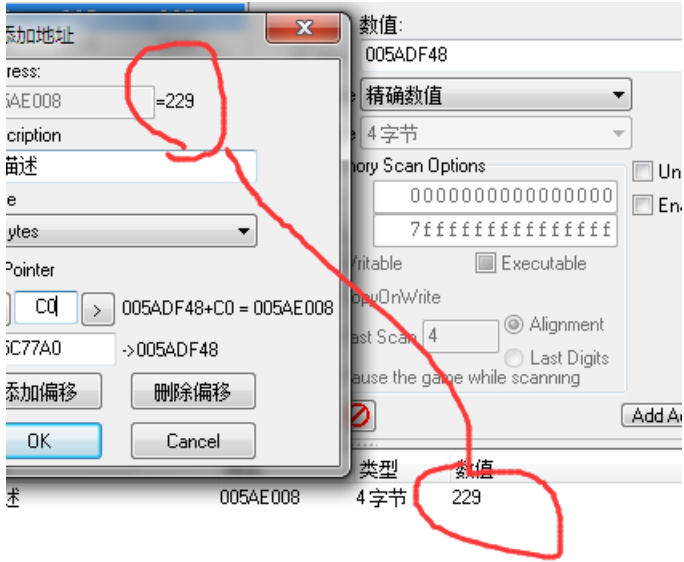
标红的部分用来算偏移量，在题目中按照规律求即可。本题中偏移量为 $eax*4$ ，使用电脑自带计算器求得偏移量为C0。要查找的地址指针值处右键，可将地址复制下来。复制之后点击OK关闭界面。将刚才的地址放到上面搜索。



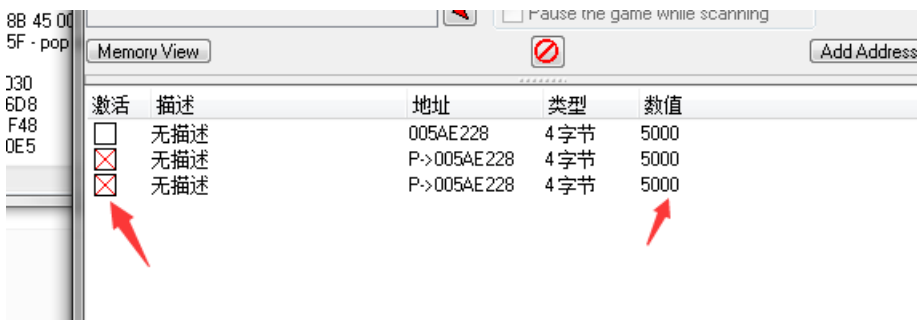
扫描后左边出现两个值，依次试一下。选中第一个值，点击右下角“Add Address Manually”，先勾选POINT，然后在下方填入查找到的地址和偏移量



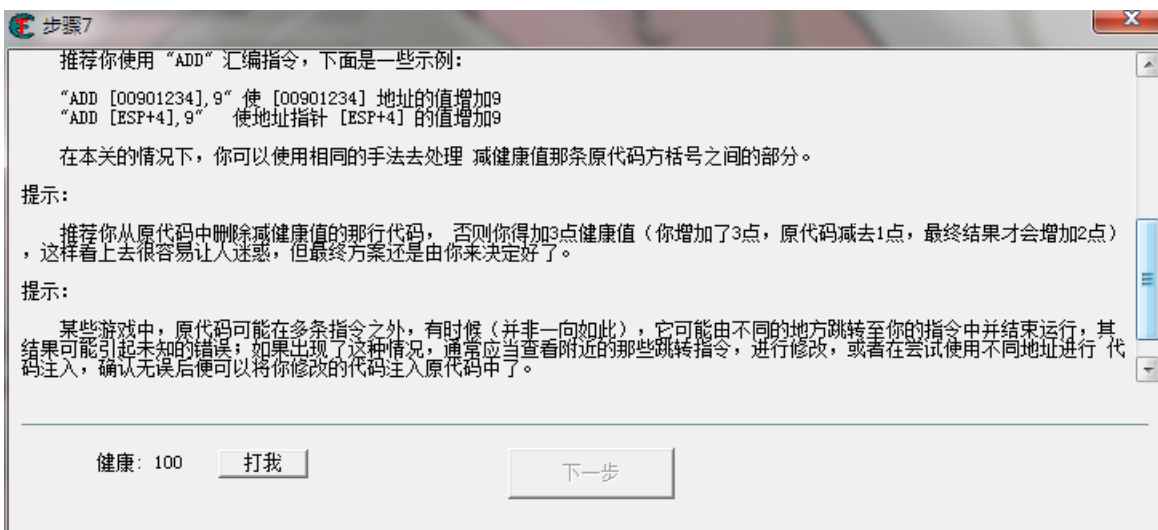
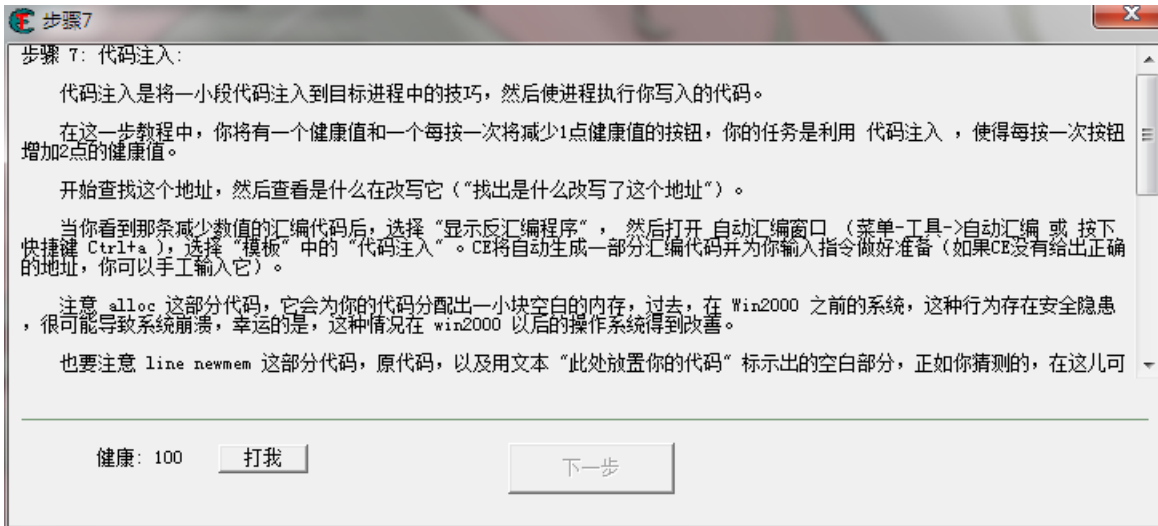
这时我们观察到指向的数值与下方的数值是一致的



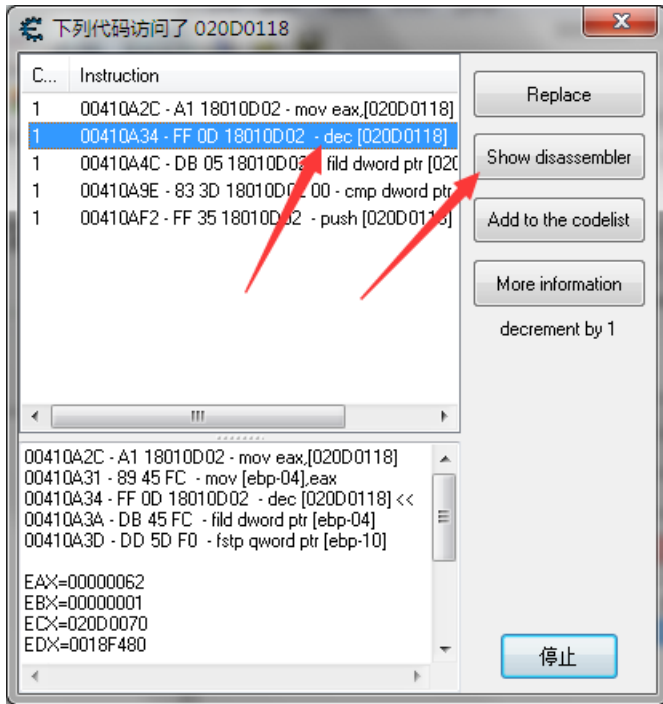
将第二条同样添加到下方。并且选中前方的锁定，然后把数值改为5000.在游戏中选择改变指针，过关。



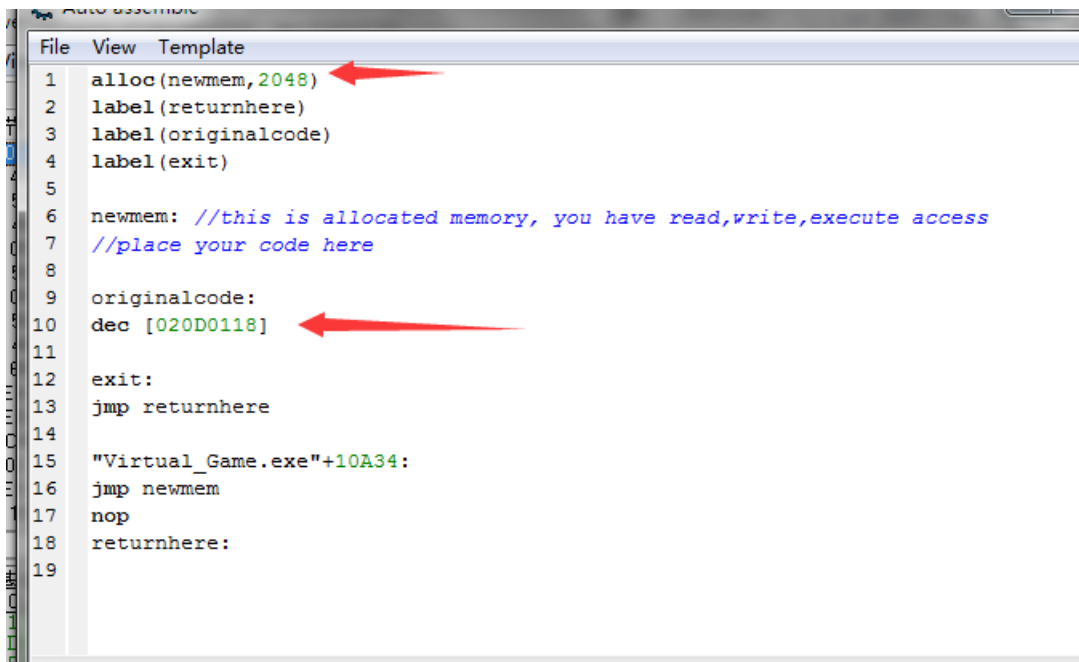
STEP7: 代码注入



首先寻找到数值的地址。右键打开调试。根据题目提示, 找到减少数值的汇编代码后, 选择右侧的“Show disassembler”即“显示反汇编程序”



打开后选择上方tools栏中的最后一项，auto assmble即自动汇编。同没办法截图。打开后选择模板（template）中的代码注入（code injection）。



其中第一个箭头指向的代码，会为代码分配出一块内存。在win2000后的操作系统改善的，否则容易存在安全隐患使系统崩溃。下方箭头指向的代码是减少血量的代码。

```

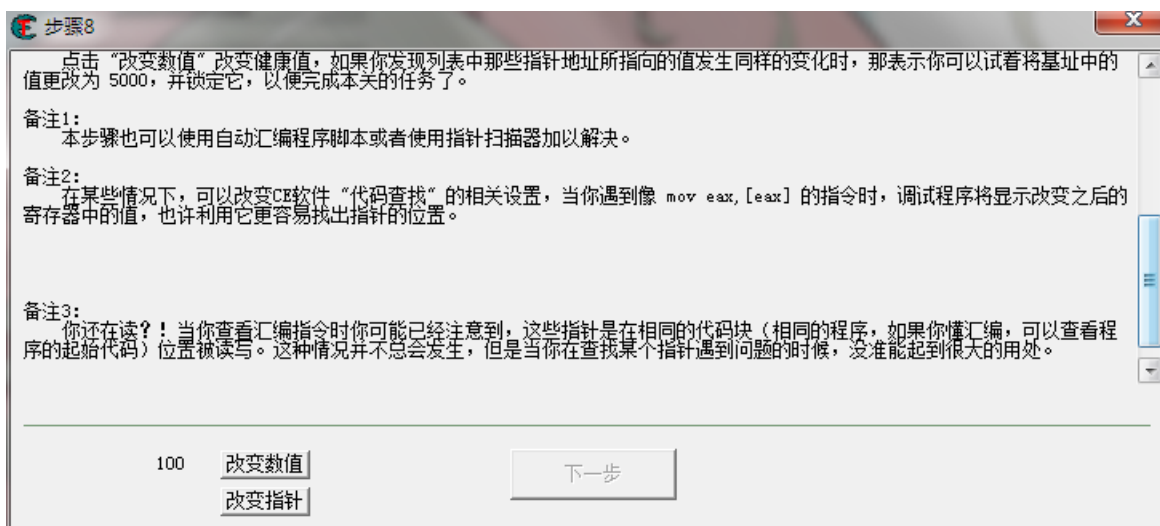
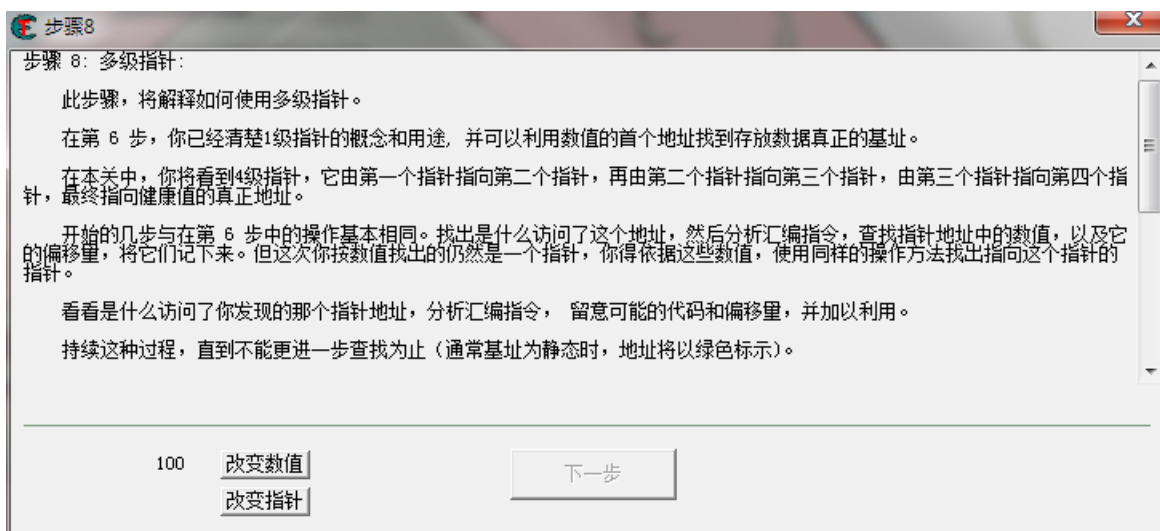
5
6 newmem: //this is allocated memory, you have 1
7 //place your code here
8 add [020D0118],3
9
10 originalcode:
11 dec [020D0118]
12
13 exit:

```

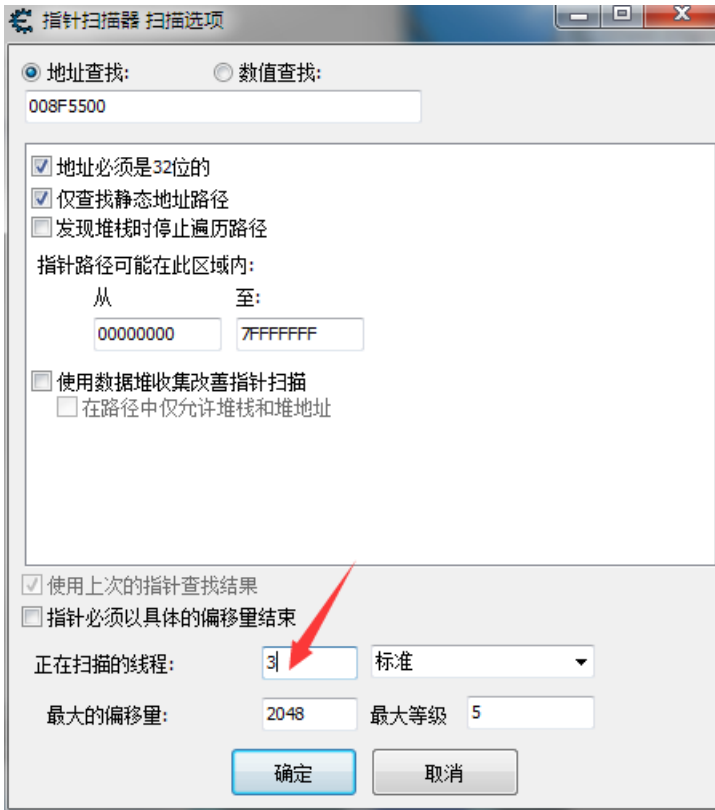
我们添加一句代码，使每点击一次“打我”按钮，健康值加三，这样健康值加三减一，最后得到题目要求的摠一次增加两点的要求。当然你可以把dec那句减少血量的代码删除。

保存后关闭界面，在游戏中点击“打我”按钮，可以看到每次点击加两点血，进入下一关。

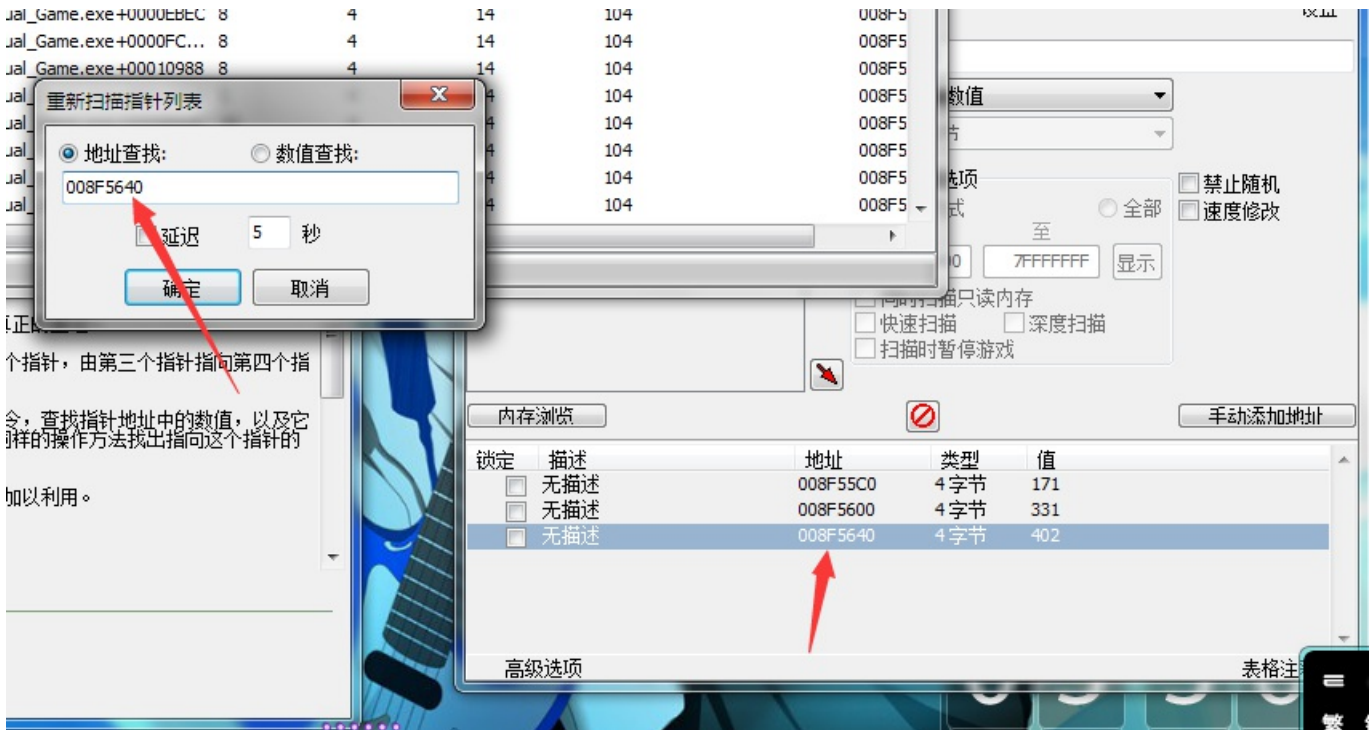
STEP8: 多级指针



采用指针扫描的方式，首先找到数值的地址。然后右键在菜单里选择“Generate Pointermap”。在弹出的新对话框如图选择，保存文件，然后多次改变指针，筛选得最后的地址。



扫描后得到约11w个结果。最小化窗口。在游戏中点击改变指针，然后重新找到数值对应的地址，在指针扫描器中选择重新扫描，输入地址为第二次找到的数值地址。



多次重复操作。然后选中第一个，锁定改5000即可过关。

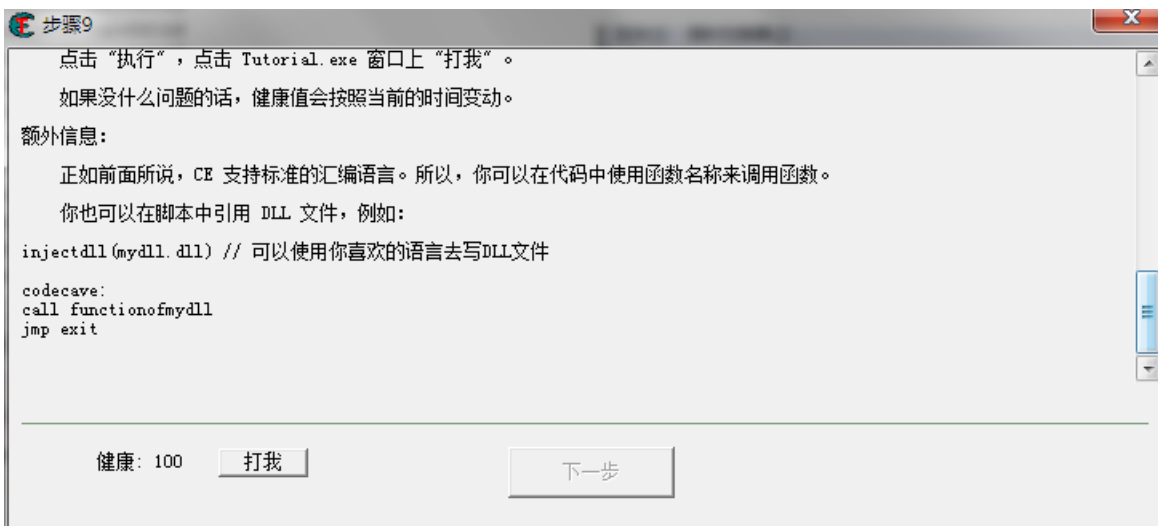
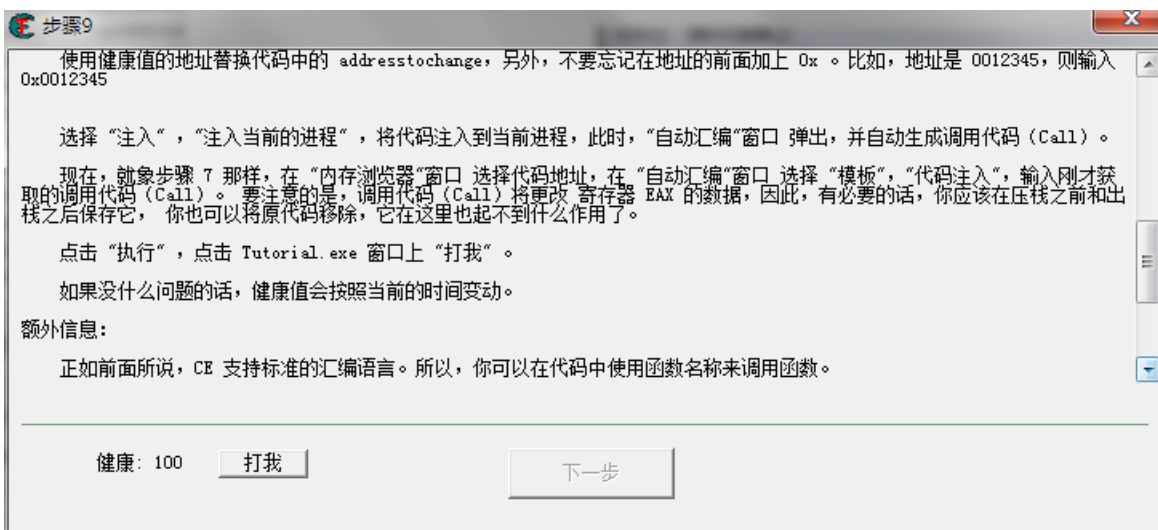
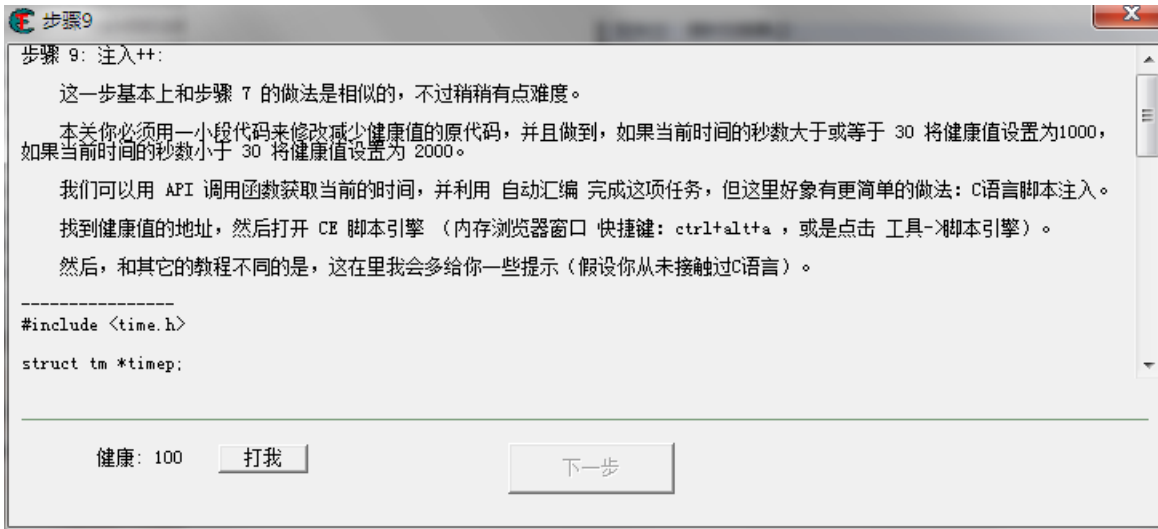
文件(F) 指针扫描器(Z)

4 字节 pointercount:40

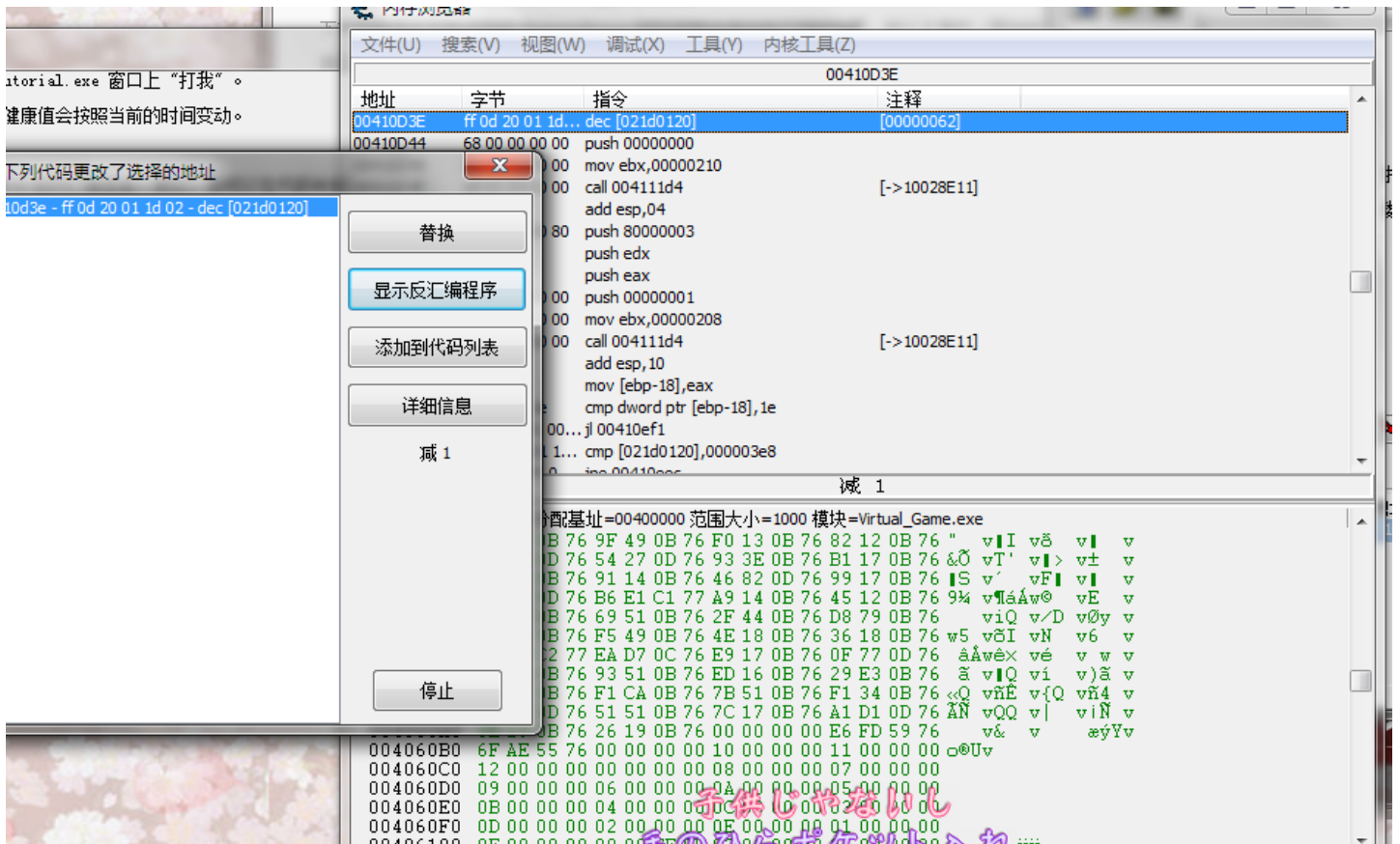
Base Address	Offset 0	Offset 1	Offset 2	Offset 3	Offset 4	Points to:
Virtual_Game.exe+0000F1A8	0	44	64	4D8	160	08489558 = 5000
Virtual_Game.exe+0000F308	0	44	64	4D8	160	08489558 = 5000
Virtual_Game.exe+0000FD...	0	44	64	4D8	160	08489558 = 5000
Virtual_Game.exe+00010B54	0	44	64	4D8	160	08489558 = 5000
Virtual_Game.exe+00010C...	0	44	64	4D8	160	08489558 = 5000

ps: 其实这步是pcat大神教的。。在我问为啥是第一个时候，大神说因为爱。。啊啊啊好可爱！！！！

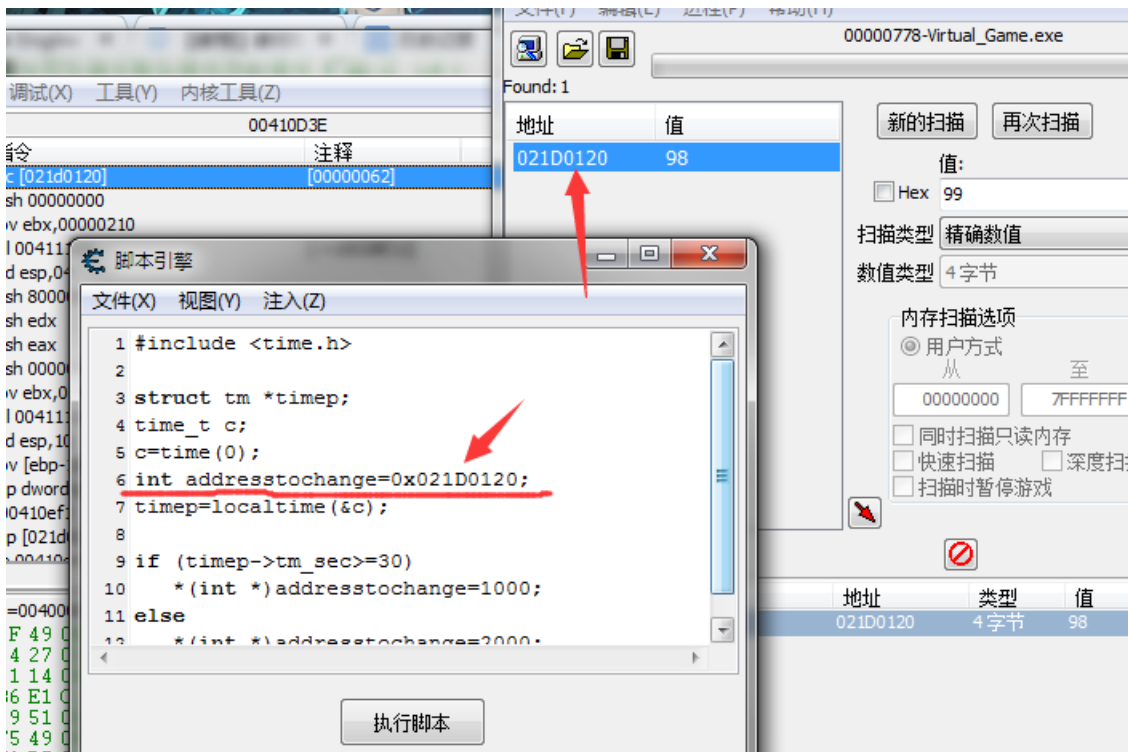
STEP9: 注入++



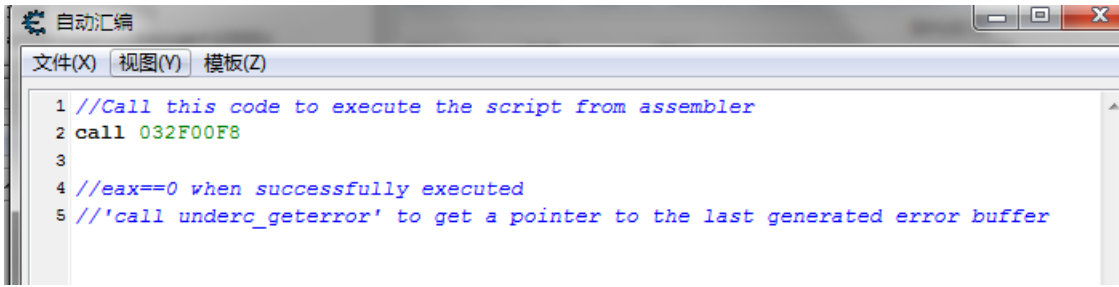
然后找到数值对应的地址, 查看反汇编程序。



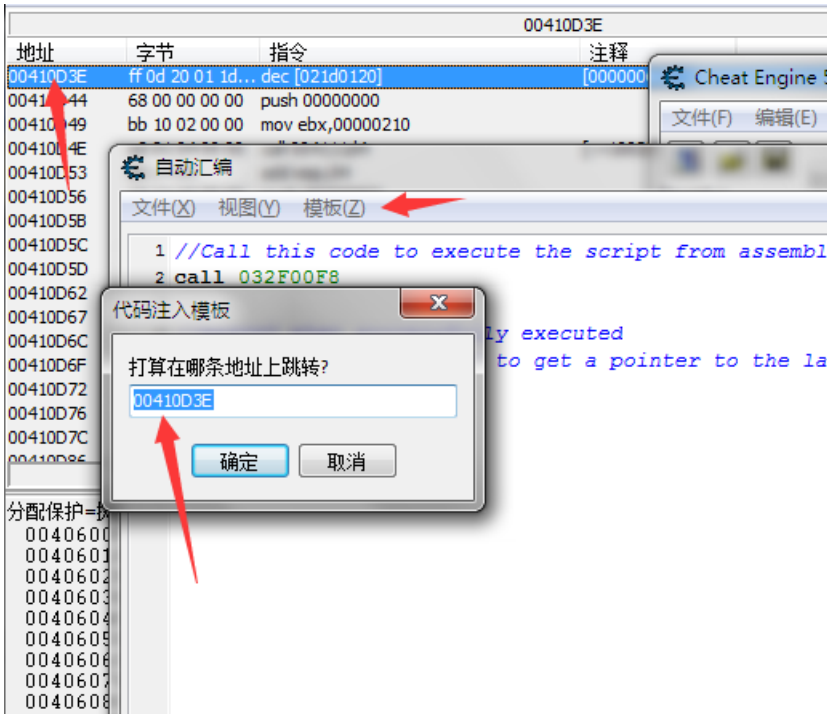
然后在工具中选择脚本引擎，打开界面、并在窗口中填入提示中的代码。并替换变量为查找到的地址。注意地址前要写上0x，如图。然后从注入栏中选择“注入到当前进程”



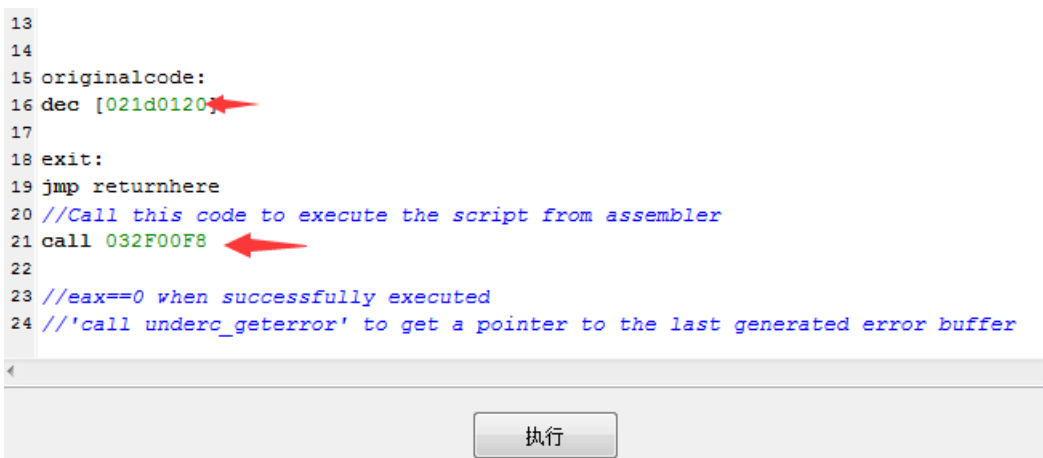
随后弹出了一个“自动汇编”的窗口



点击模板，选择代码注入。然后在弹出对话框中填入第一项的地址。如图



然后得到如下的代码。其中第一个箭头指向的是源代码。第二个箭头指向我们输入的代码。



然后我们把call这一条代码放在newmem区域，并且把源代码注释，如图。

```
自动汇编
文件(X) 视图(V) 模板(Z)
7 jmp newmem
8 nop
9 returnhere:
10
11 newmem: //this is allocated memory, you have read,write,execute access
12 //place your code here
13 call 032F00F8
14
15 originalcode:
16 //dec [021d0120]
17
18 exit:
19 jmp returnhere
20 //Call this code to execute the script from assembler
21
22
23 //eax==0 when successfully executed
24 //'call underc_geterror' to get a pointer to the last generated error buffer
```

执行

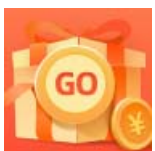
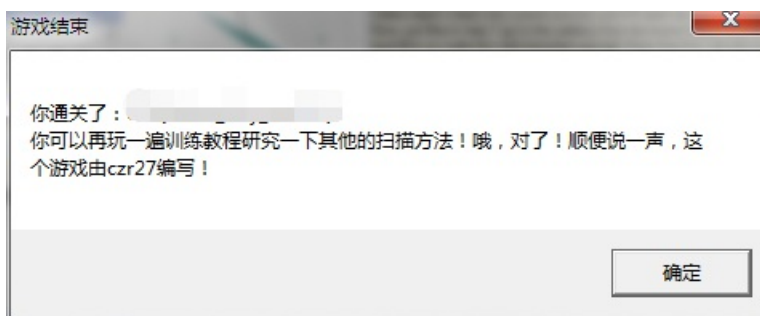
点击执行，原来代码已被改写，如图。

内存浏览器

文件(U) 搜索(V) 视图(W) 调试(X) 工具(V) 内核工具(Z)

地址	字节	指令	注释
00410D3E	e9 bd f2 21 03	jmp 03630000	
00410D43	90	nop	
00410D44	68 00 00 00 00	push 00000000	
00410D49	bb 10 02 00 00	mov ebx,00000210	
00410D4E	e8 81 04 00 00	call 004111d4	[->10028E11]
00410D53	83 c4 04	add esp,04	
00410D56	68 03 00 00 80	push 80000003	
00410D5B	52	push edx	
00410D5C	50	push eax	
00410D5D	68 01 00 00 00	push 00000001	
00410D62	bb 08 02 00 00	mov ebx,00000208	
00410D67	e8 68 04 00 00	call 004111d4	[->10028E11]

回到游戏窗口点击“打我”按钮，得到通关的对话框。耶！



[创作打卡挑战赛](#) >

[赢取流量/现金/CSDN周边激励大奖](#)