逆向新手踩坑指南之爬爬山能锻炼身体

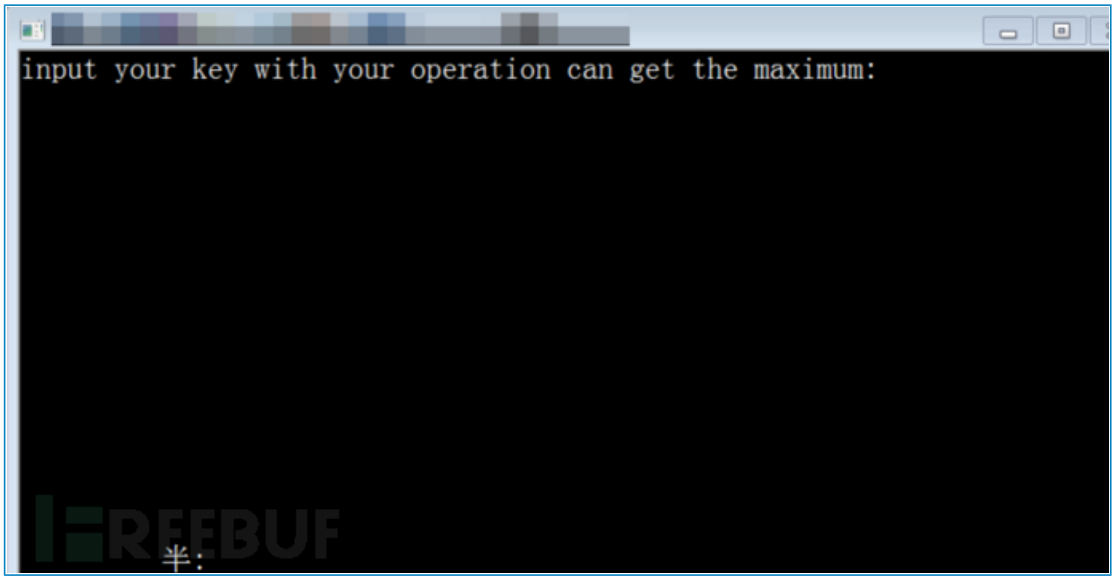
转载

weixin_30527551 于 2017-12-19 10:03:00 发布 60 收藏
文章标签： python c/c++ 操作系统
原文链接：http://www.cnblogs.com/h2zZhou/p/8064099.html

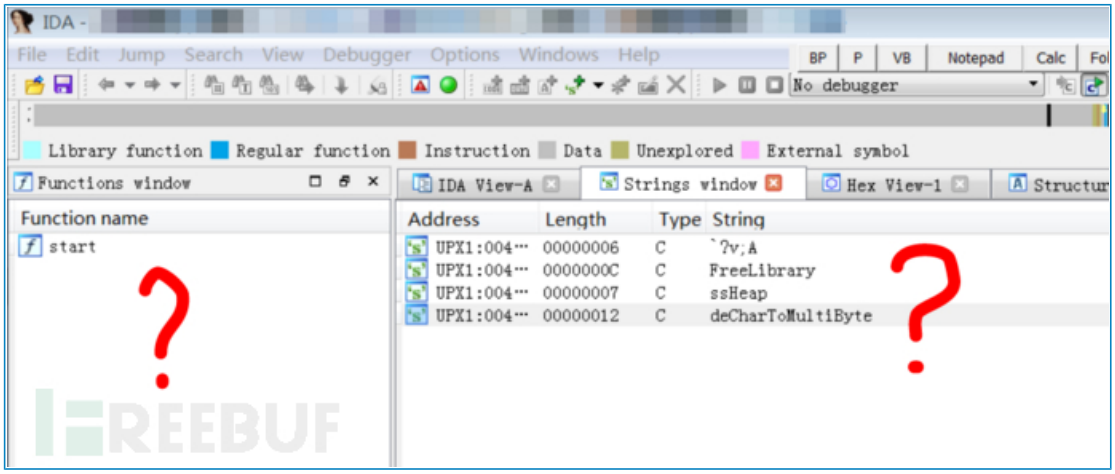
首先坐下，打开电脑，平复一下心情，开始逆向分析。

对了，开始之前，首先推荐一下本站大神的逆向工程系统教程：【传送门】

好了，正文开始。软件运行一下是这样的：

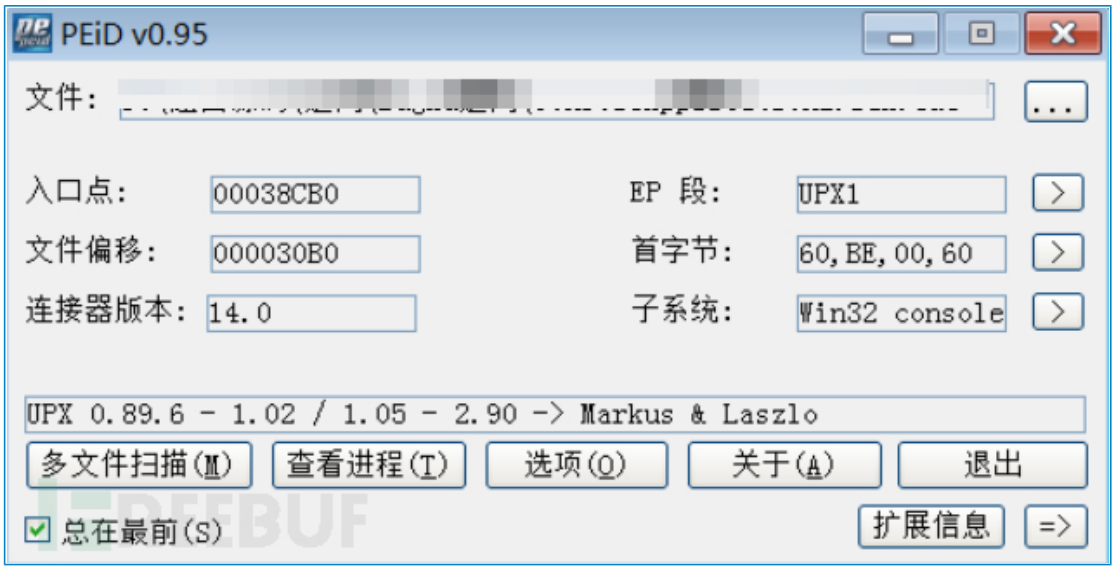


直接IDA打开，字符串窗口（shift+F12）：

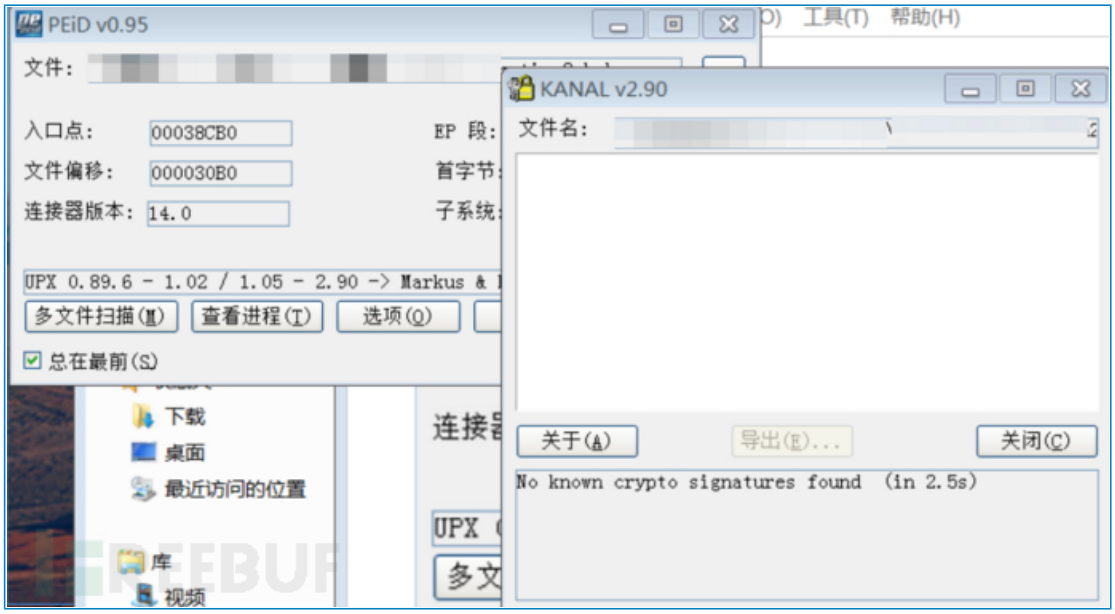


what a f********ck？？？！！！这是啥？说好的字符串呢？说好的各种函数呢？

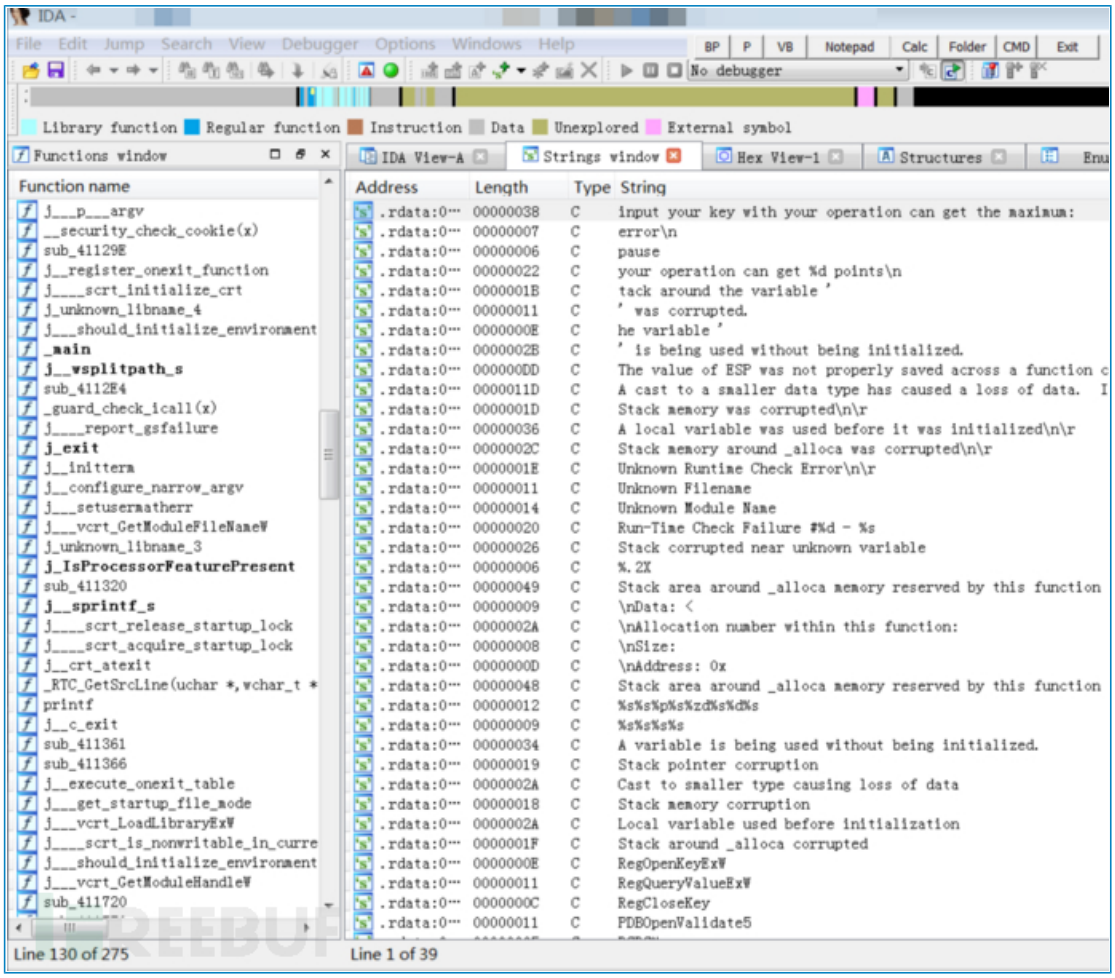
想了想忘了一件事情，先查查有没有壳：



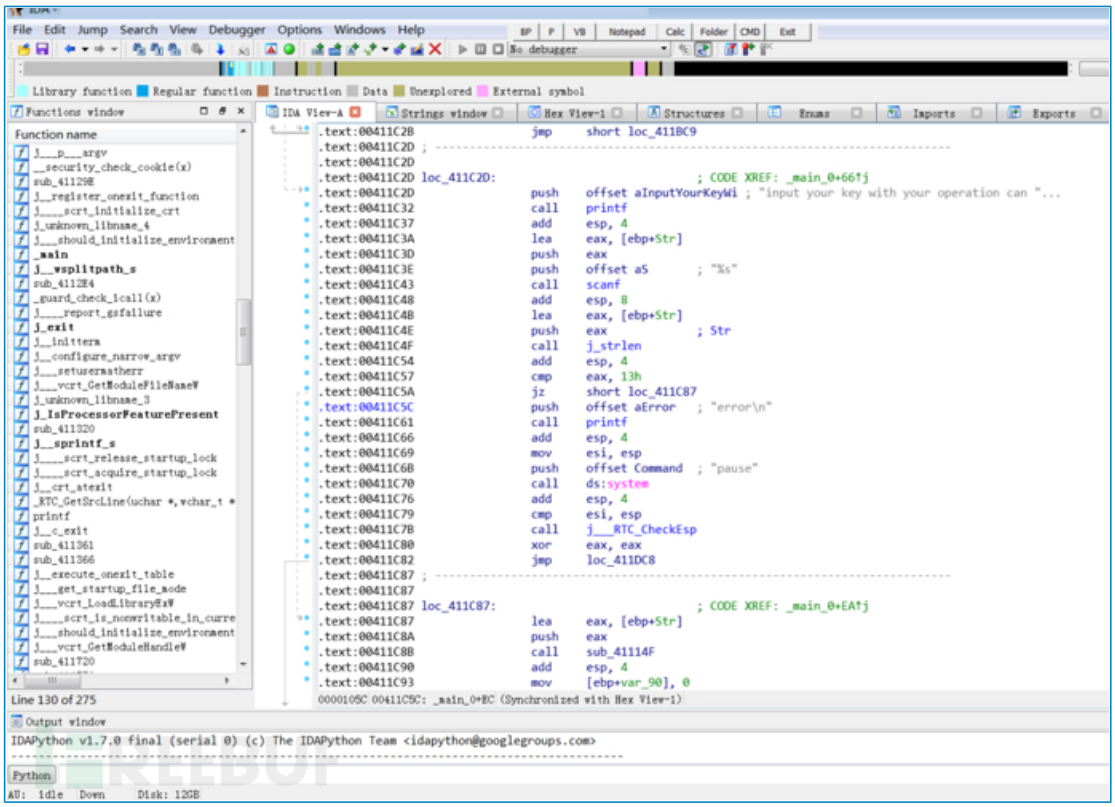
看到了刚才的小坑，顺便在看看有没有什么加密方式：



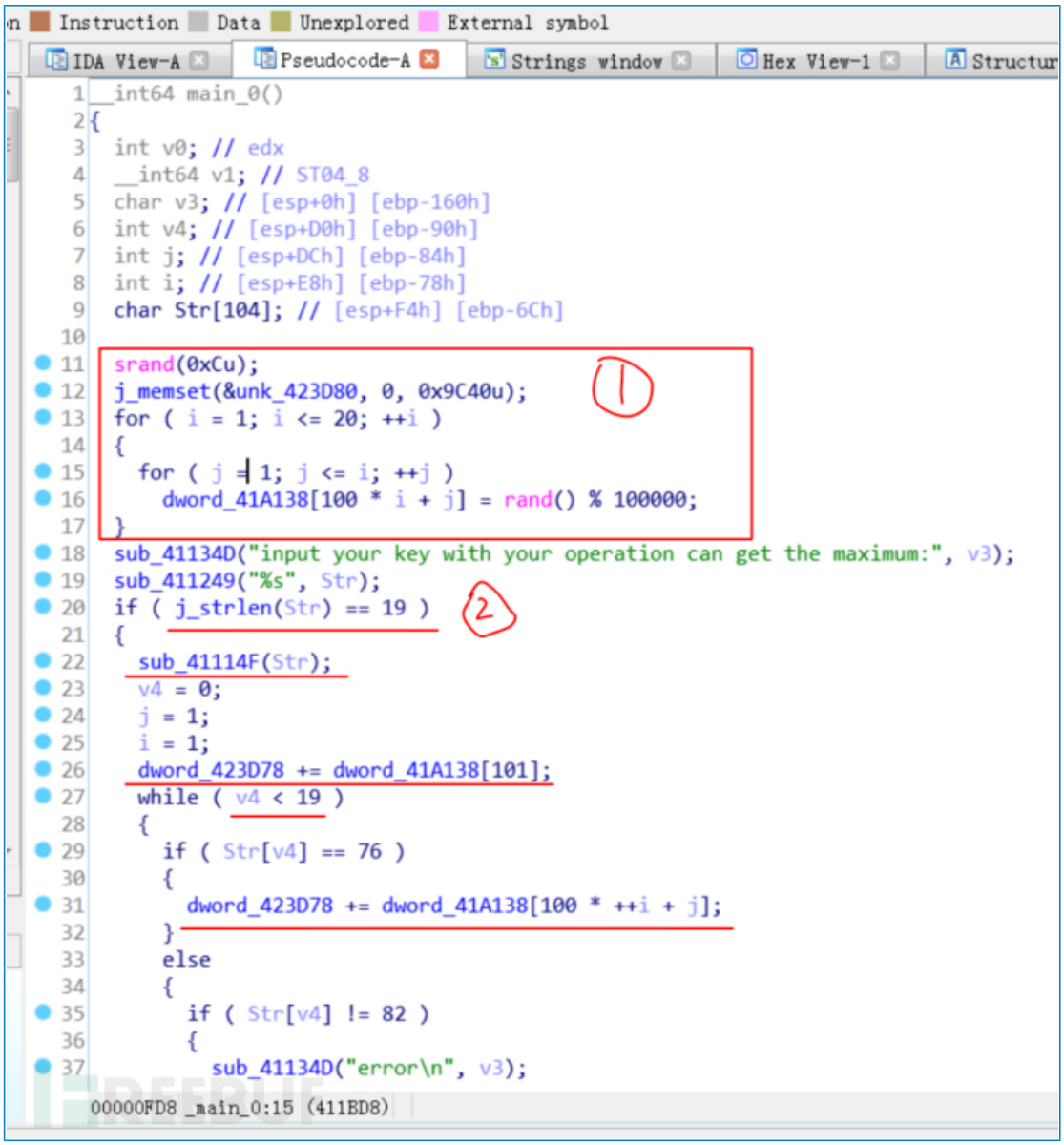
没有，很好，直接upx脱壳，然后再次进入IDA【其实根据刚才IDA的提示也能看出来是upx的壳】：



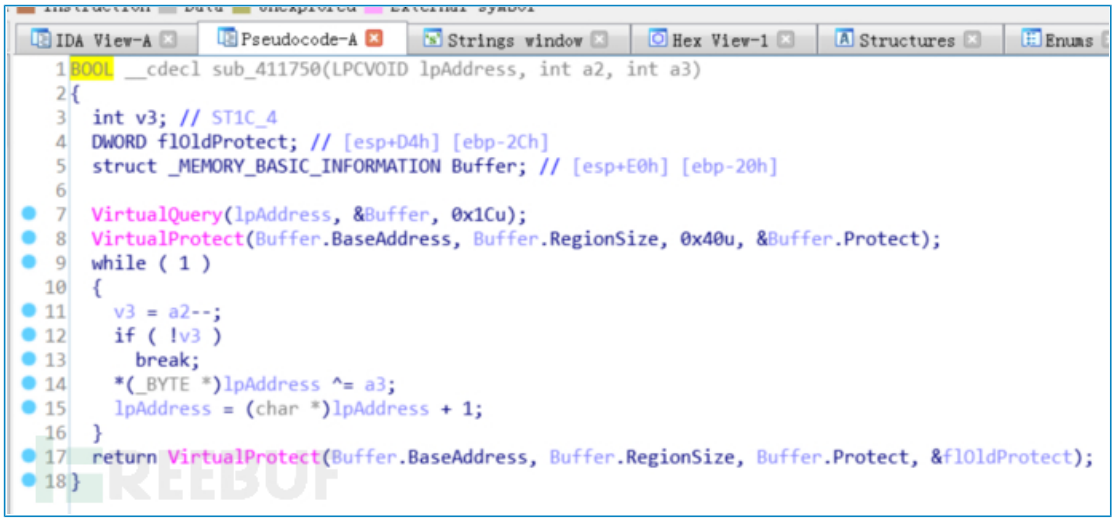
这次看到了字符串，然后根据关键字符串交叉引用后来到函数：



直接F5：



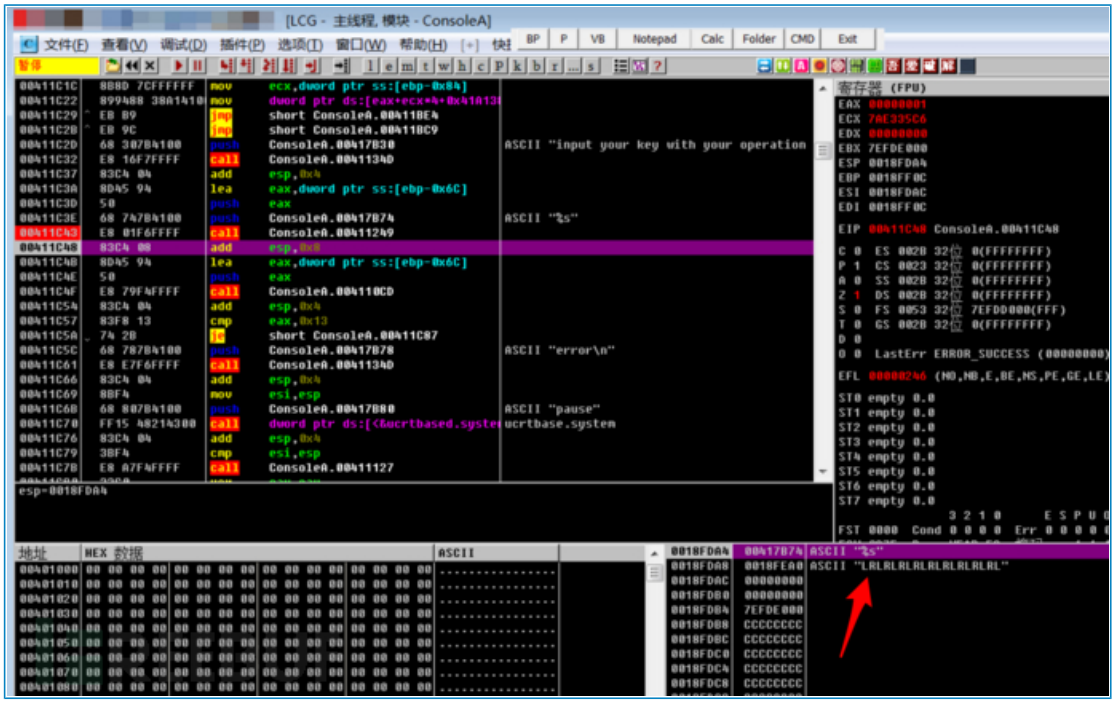
圈一里面是什么暂时不想管，直接从18行开始基本比较明显了，一个是屏幕输出函数printf，一个是从屏幕获得输入，圈二IDA本身分析得也很明显，就是判断长度的，字符串限制为19。22行跟进去以后是这样的：



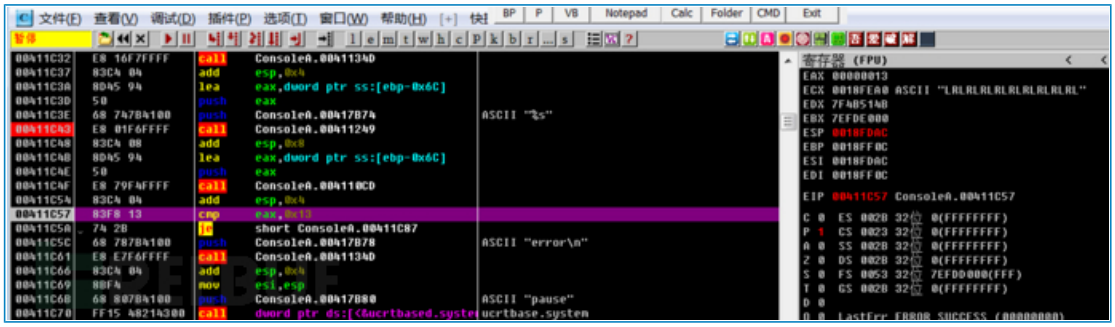
这TM是啥？！？！？！不过结合26行，31行，37行能看到类似于a=a+b一样的东西，是从dword_41A138[]里面取值，然后直接加到dword_423D78这个变量里面。

从29行和35行能判断这个输入的字符串一定是L或者R【根据ASCII码】，既然22行不懂，就直接先动态走一走：

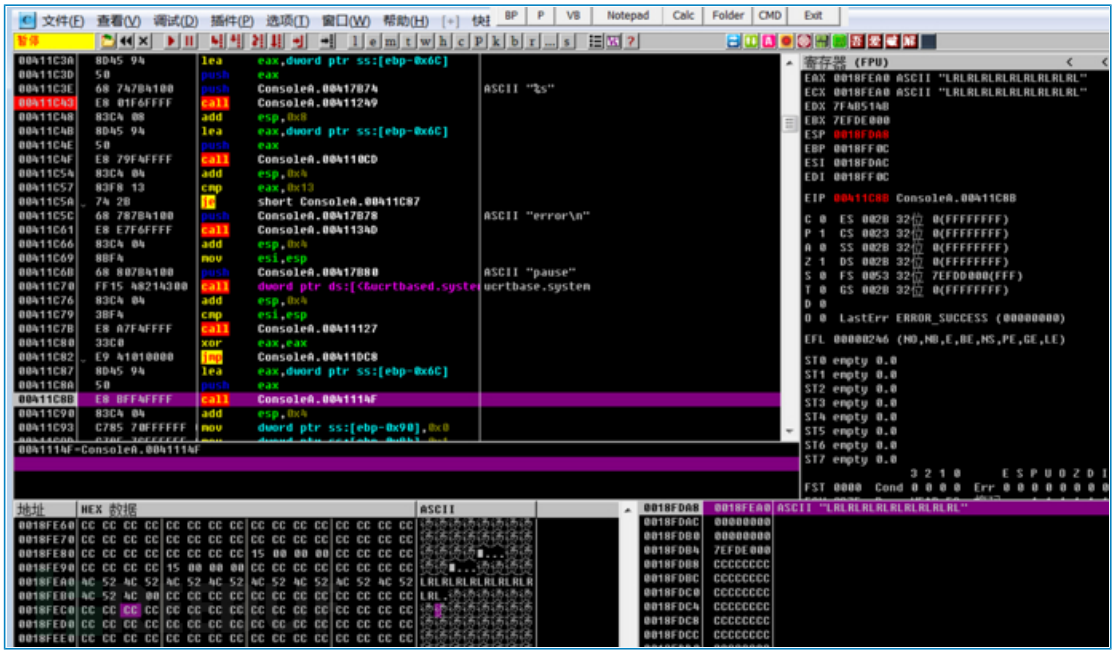
输入19个字符串后：



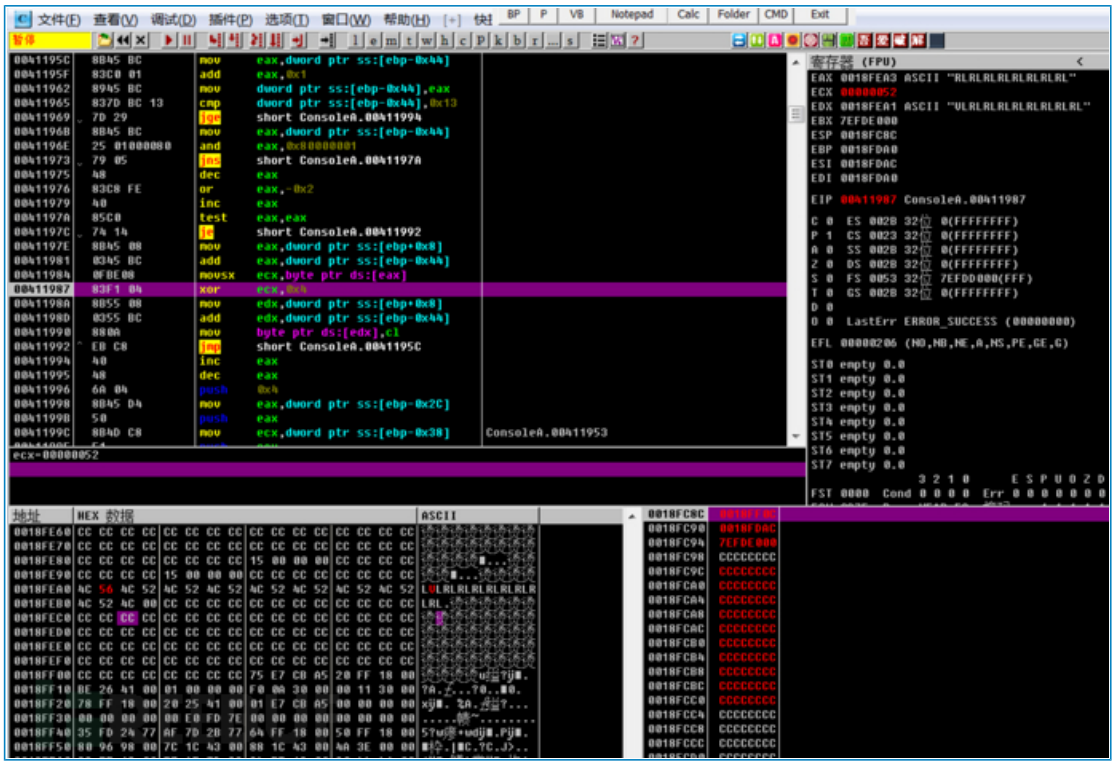
在堆栈段看到了自己的输入，然后跟进411C4F这个函数：



跟进去看得有点头晕....但是看到函数的返回值eax里面是13，换成十进制就是19，也就是这个函数是用来判断输入长度的【从IDA的整体流程也能推测出来】，继续跟：



进入411C88这个函数：

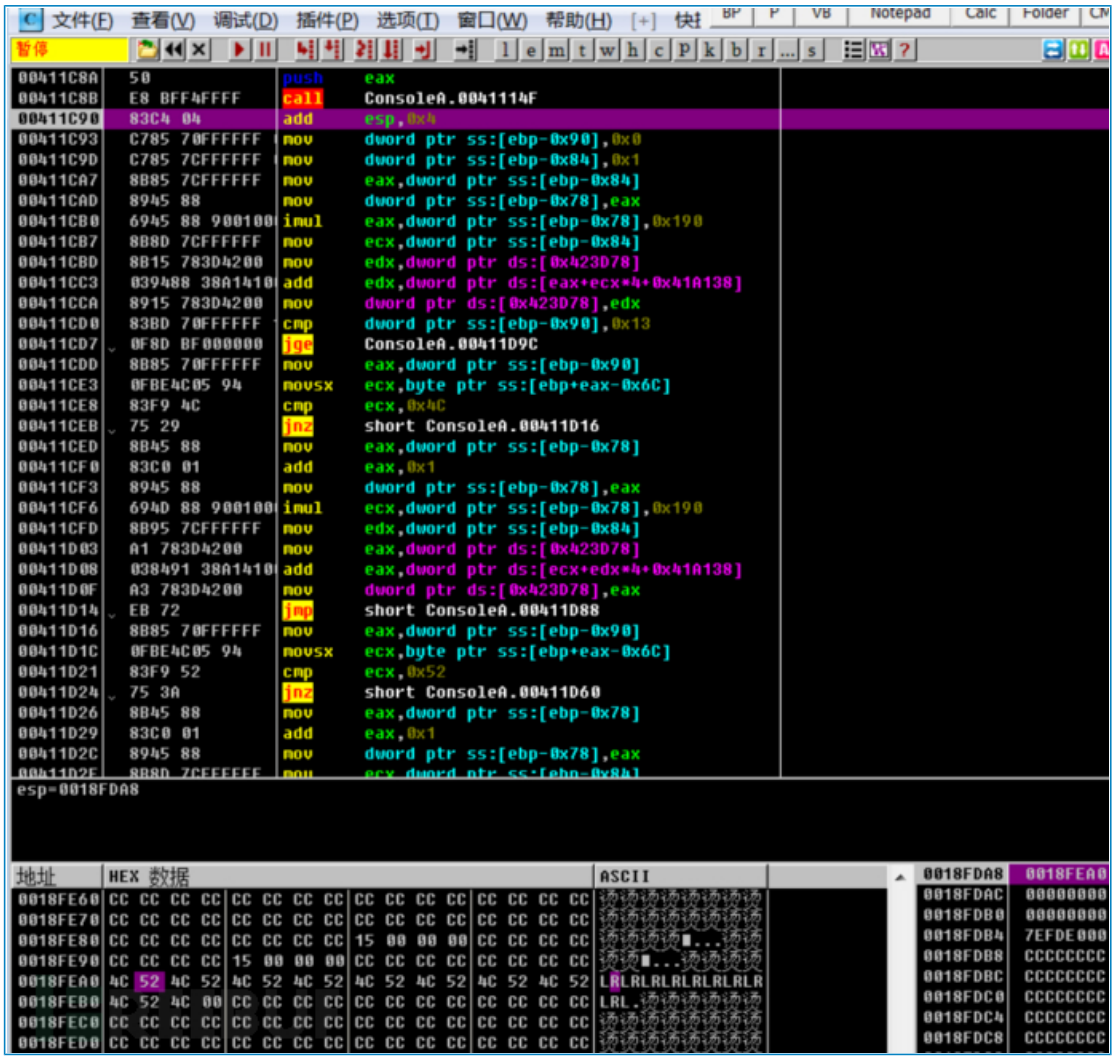


这里是一个重点，函数在这里一直在41195C和411992之间循环，并且，在411987这一步直接和4做了异或，直接导致将我们输入的字符串所有偶数位做了异或。从这一点判断，这个函数应该是IDA伪代码的第22行，对字符串进行了处理：

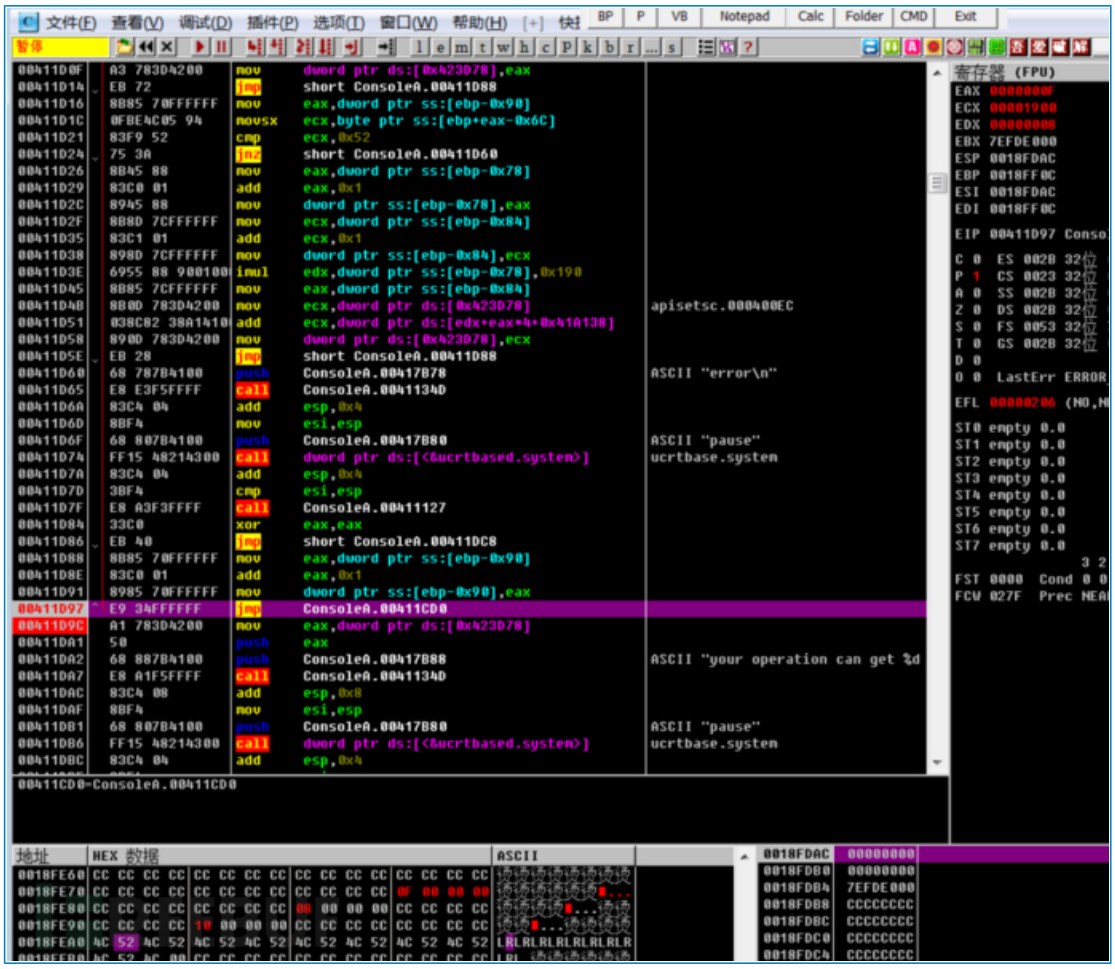
```
18  sub_41134D("input your key with your operation can get the maximum:", v3);
19  sub_411249("%s", Str);
20  if ( j_strlen(Str) == 19 )
21  {
22    sub_41114F();
23    v4 = 0;
24    j = 1;
25    i = 1;
26    dword_423D78 += dword_41A138[101];
27    while ( v4 < 19 )
28    {
29      if ( Str[v4] == 76 )
30      {
31        dword_423D78 += dword_41A138[100 * ++i + j];
```

处理方法就是偶数位和4做异或，根据IDA的判断，异或后的值只能是L或者R。于是再次输入的时候就知道该输入些什么了。经过反推，奇数位只能是L或者R，偶数位只能是V（代表R）或者H（代表L）。字符串输入什么变得清晰了。

直接让程序执行完，然后手动改栈里面的值，再返回：



从这里开始的一大长串我没有仔细分析，太多了，看得心烦，跟了几次以后发现总是在这里[411D97]跳转，方便起见直接这里下了断点，也在[411D9C]这里下了断点，之所以这样，是想随时关注着各个寄存器的变化和堆栈的变化 ：



然后结合IDA的整个流程：

```
Instruction  Data  Unexplored  External symbol
IDA View-A | Pseudocode-A | Strings window | Hex View-1 | Struc
16      dword_41A138[100 * i + j] = rand() % 100000;
17    }
18    sub_41134D("input your key with your operation can get the maximum:", v3);
19    sub_411249("%s", Str);
20    if ( j_strlen(Str) == 19 )
21    {
22      sub_41114F();
23      v4 = 0;
24      j = 1;
25      i = 1;
26      dword_423D78 += dword_41A138[101];
27      while ( v4 < 19 )
28      {
29        if ( Str[v4] == 76 )                    (1)
30        {
31          dword_423D78 += dword_41A138[100 * ++i + j];
32        }
33        else
34        {
35          if ( Str[v4] != 82 )                  (2)
36          {
37            sub_41134D("error\n", v3);
38            system("pause");
39            goto LABEL_18;
40          }
41          dword_423D78 += dword_41A138[100 * ++i + ++j];
42        }
43        ++v4;
44      }
45      sub_41134D("your operation can get %d points\n", dword_423D78);
46      system("pause");
47    }
```

能大概推测出这里应该就是根据输入的字符串判断是在圈一还是圈二。

处理完毕以后就能直接出结果了：

```
input your key with your operation can get the maximum:LRLRLRLRLRLRLRL
RLRL
your operation can get 312869 points
请按任意键继续. . .
```

还是我太天真，将输入提交以后发现并不正确。然后思路基本就卡到这里了，连续一两天没有进展，心里憋屈得不得了。后来再次看题目的时候发现题目是mountain climbing。英文翻译是爬山，爬山，爬山，山呢？！！！于是我的思路就有变成了找山...

继续回到IDA来看吧，因为已经没什么可以看的了。这次就变成了分析前面所有没分析过的函数，首先就是dword_41A138[]里面是什么？

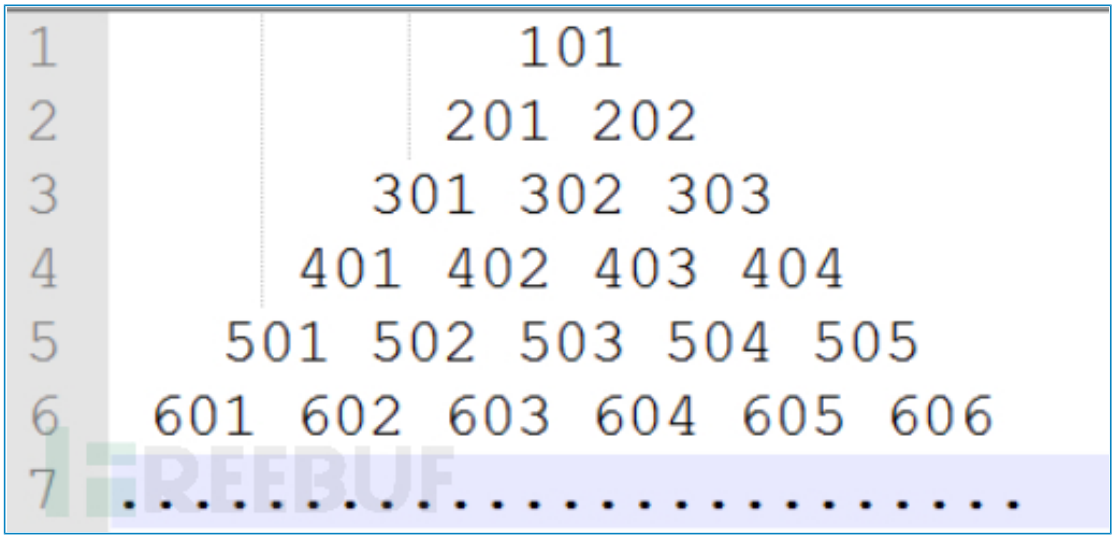
```
11    srand(0xCu);
12    j_memset(&unk_423D80, 0, 0x9C40u);
13    for ( i = 1; i <= 20; ++i )
14    {
15      for ( j = 1; j <= i; ++j )
16        dword_41A138[100 * i + j] = rand() % 100000;
17    }
18    sub_41134D("input your key with your operation can get the maximum:", v3);
19    sub_411249("%s", Str);
```

从字面上理解rand()就是随机数函数，每次都产生不一样的随机数，然后根据这个去进行加数字的运算，这TM能算出来啥？！

然而事实告诉我还是我太天真了(原谅我大学不是计算机专业，没好好学过C....)。

c语言里面有一个产生随机数的函数，rand()。但是在学习过程中发现这是伪机的，虽然叫随机，但是每一次结果都一样(下面图片来自随便一个搜索引擎，随便一搜一大把的讲解)：

于是思路一下子清晰了，srand()是一个固定的值(IDA里面对没有定义的数字后面会有一个u，undefined)，那么rand()产生的数字也是固定的！！！根据代码描述，基本可以断定那个dword_41A138[]就是山了，数组下标是这样的101,201,202,301,302,303...写成这样也许更直观：

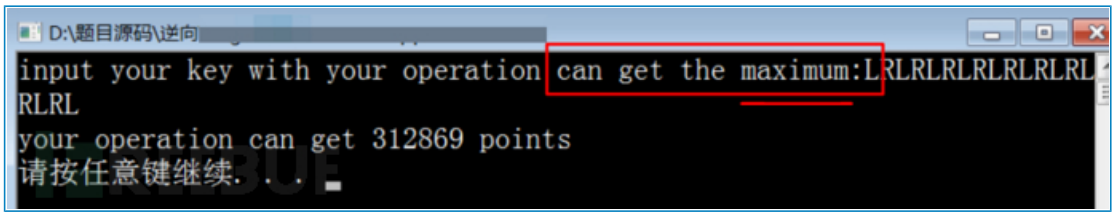


从IDA里的27行到41行看：

```
27    while ( v4 < 19 )
28    {
29      if ( Str[v4] == 76 )
30      {
31        dword_423D78 += dword_41A138[100 * ++i + j];
32      }
33      else
34      {
35        if ( Str[v4] != 82 )
36        {
37          sub_41134D("error\n", v3);
38          system("pause");
39          goto LABEL_18;
40        }
41        dword_423D78 += dword_41A138[100 * ++i + ++j];
42      }
43      ++v4;
44    }
```

76(十进制ASCII码)是L，82(十进制ASCII码)是R，对应上面那张图就是从101到201是L，从101到202是R。L和R就代表了如何下山。没错，你没有看错，就是下山，不是爬山。

问题来了，根据等差数列进行算数，一共由210个数字堆成山，一共20层，走法多了去了，这TM谁知道哪条路是对的...正在一筹莫展的时候想起了这一句话：



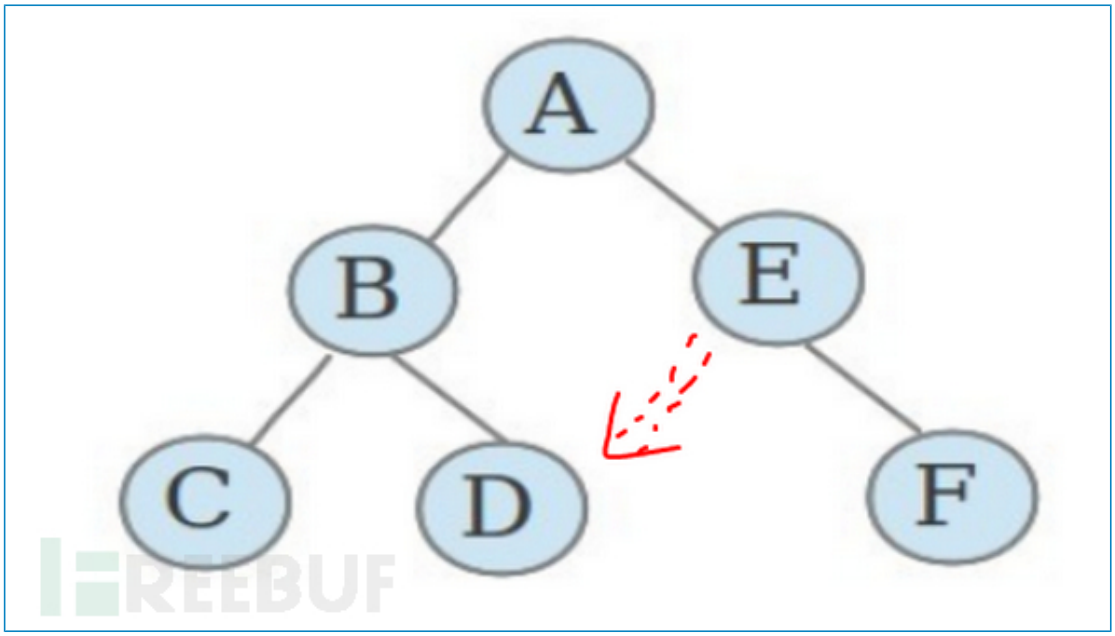
要找最大的数，那么也简单，从下山开始，只选择最大的那个数。后来发现也不行，比如201是比202大，但是303比所有101,201,202加起来都要大，如果选了201，那么永远走不到303。这怎么办？

只能遍历了。

问题又来了，怎么遍历？！！我想到了二叉树遍历，虽然C语言忘得差不多了，但是还依稀记得一点数据结构。当我自信满满得去操作一番搜索引擎的时候又进到了一个坑里，二叉树遍历的代码是这样的(来自用搜索引擎搜索二叉树遍历后返回的一大把博客中随便一个网络博客)：

```c
#include<stdio.h>
#include<stdlib.h>

#define STACKINITSIZE 100
#define STACKINCREASESIZE 20

typedef char ElemType;
//树结构
typedef struct tree
{
    ElemType data;
    struct tree * lchild;
    struct tree * rchild;
    unsigned int isOut;   //专为后序遍历设置的，0为不需要被输出，1为需要被输出
}TreeNode,*Tree;

//栈结构
typedef struct stack
{
    Tree * base;
    Tree * top;
    int stacksize;
}Sqstack;


/*****************栈的操作声明*******************/
```

一共316行，相信我，这不是我想要的，首先以我近乎于只穿着内裤的C语言功底读懂316行代码还需要很多时间，其次还要搞明白并熟练使用结构体变量。而且，经过耐心的学习以后发现二叉树遍历不能满足我的需求，因为二叉树遍历无法实现从E到D：



于是我的这道题目再次陷入深坑....

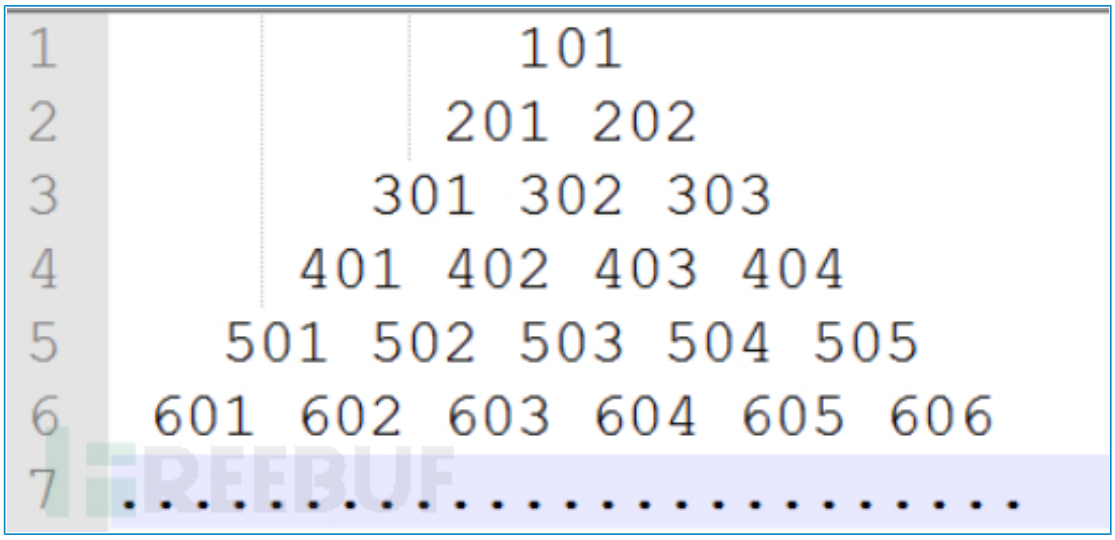
这里卡了一天半左右，没有任何思路，后来无聊，想了想一共有多少条路？我能不能手工遍历(我是一个天真善良吃苦耐劳的八道杠优秀代表)。从上往下数第二层有2条，走到第三层有4层，第四层有8条…不会算了，在纸上画到第五层的时候我已经有点崩溃了，又耐心思考了一个小时左右，耐着性子画到了六层，终于让我这个晕乎乎的脑袋看到了规律，2的几次方。从山顶到第二层有2条路，到第三层有4条路，是2的平方。第四层是2的3次方。因为每一个节点对应下一层的节点只有两个选择。我打开windows键，在运行处输入calc以后用科学计算器算了算2的19次方：

这是逼着我用程序来解这道题(此时心里有一万句脏话想说出口)。

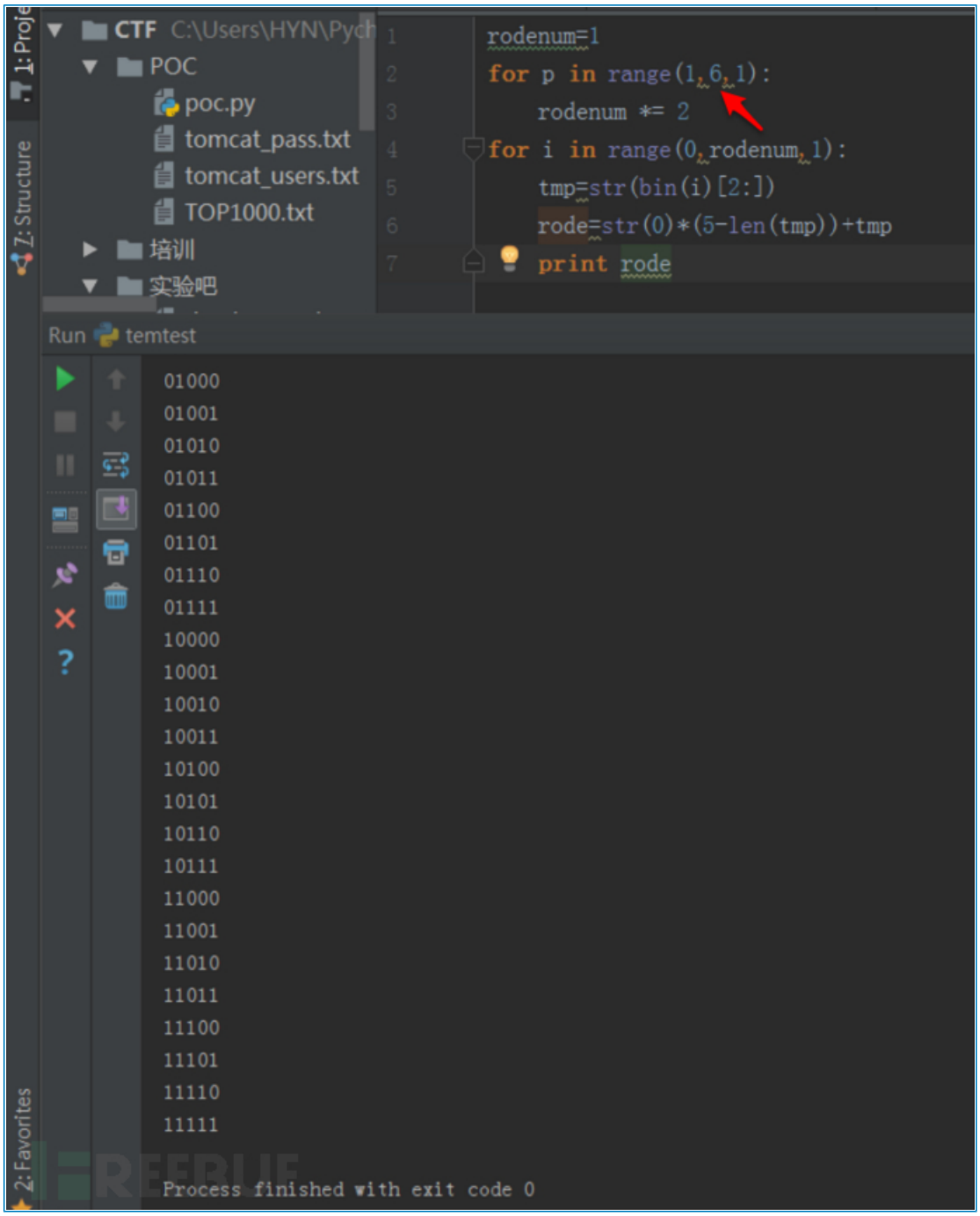
既然用程序，问题就又来了，我如何表现每一种走法呢？这里我又卡住了，盯着自己在纸上画的所谓的山看了好久也没有想出来用什么办法来表现每一条路径。看看作者的源程序：

```
12  j_memset(&unk_423D80, 0, 0x9C40u);
13  for ( i = 1; i <= 20; ++i )
14  {
15    for ( j = 1; j <= i; ++j )
16      dword_41A138[100 * i + j] = rand() % 100000;
17  }
```

作者用了两个数字来写，i就是第几层，j就是在i层的第几个数。可是我表现不了从302到403啊，等等，302,403两个数层数和尾标都是差1，在看看整个山的分布：



找到了一个规律，每条路都是只能差1，比如这条路：101,201,302,403,503,604…这条路里，层数差1是固定的，可以用循环实现，问题是如何实现1,1,2,3,3,4这个路数。要么加0，要么加1。要么加0，要么加1。要么加0，要么加1。默念无数遍以后得到了这样一个数列01101，我嘞个王母娘娘，臣妾终于做到了~！！！这TM不就是二进制么，0代表往左走一步，1代表往右走一步。（伟大的二进制~！伟大的十六进制~！）如果有6层，那么01101换算成十进制就是13，一共有32条路，00000也算一条路的话（十进制的0），11111就是最后一条路，十进制就是31，0到31一共32个数字，Nice，完美实现了所有路径的表示，此时，半天又过去了…于是代码实现就容易得多了，先定一个小目标，实现6层的山，用Python：

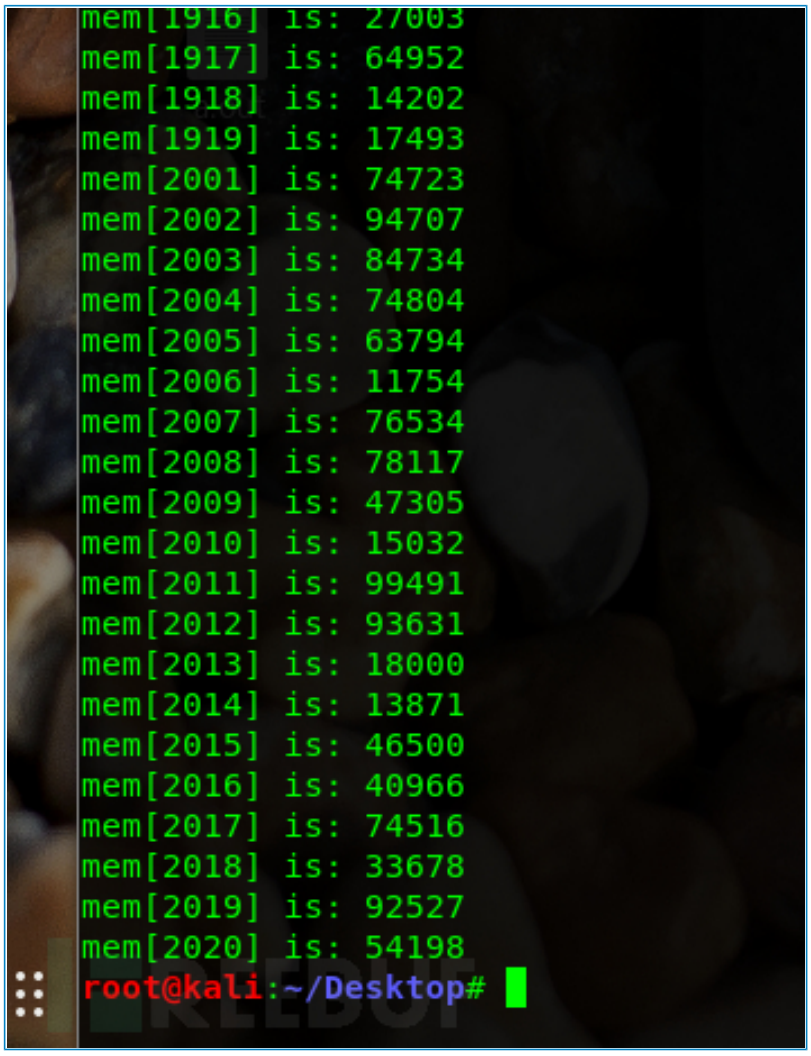


已然实现了所有路径的表示方法，Nice！！！

下面面临着怎么把山里的数字给扒出来，从作者的定义方式来看，定义的数组的长度在2000多，否则下标放不下，简单，直接对照伪代码写出来C代码，然后自己跑一边就行，于是有了这段C代码：

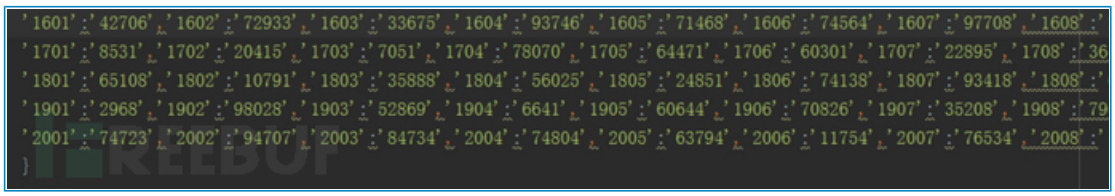
```c
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int i,j;
    long int mem[3000];
    srand(12);
    for (i=1;i<=20;++i)
    {
        for (j=1;j<=i;++j)
            mem[i*100+j]=rand() % 100000;
    }
    for (i=1;i<=20;++i)
    {
        for (j=1;j<=i;++j)
            printf("mem[%d] is: %d\n",i*100+j,mem[i*100+j]);
    }
}
```

由于强迫症的限制，我就不想在windows上跑，没环境，不想装，伟大的linux干啥都行，为啥非要在windows上吊死，于是在我的kali上跑完以后是这样的（自己又给自己挖了一个坑）：



好了，数字找到了，也能按照自己的意愿去进行遍历了，剩下的就是融合C和Python在一起，用Python实现脚本计算（别问我为啥不用C）。

问题TMD又来了，我要如何把210个数字移到Python里，这里，我选择了裸奔的方式：手工输入！（C语言如何实现文件读写不会！不想学！）210个数字写到Python的字典里，比如：



实现遍历的方法在前面已经写出来，还剩下如何根据路径把对应的数字进行相加，给一段关键代码：

```
for i in range(0,rodenum,1):
    tmp=str(bin(i)[2:])
    rode=str(0)*(19-len(tmp))+tmp
    j=0
    mountainlr=1
    ournum=0
    #print "current rode is:",rode
    while j < len(rode):
        mountainh=100*j+200
        mountainlr += int(rode[j])
        mountainpoint = mountainh+mountainlr
        j += 1
        ournum += int(mountain[str(mountainpoint)])
        if ournum > bigestnum:
            bigestnum = ournum
            maxrode = str(rode)
    print bigestnum,maxrode,i
```

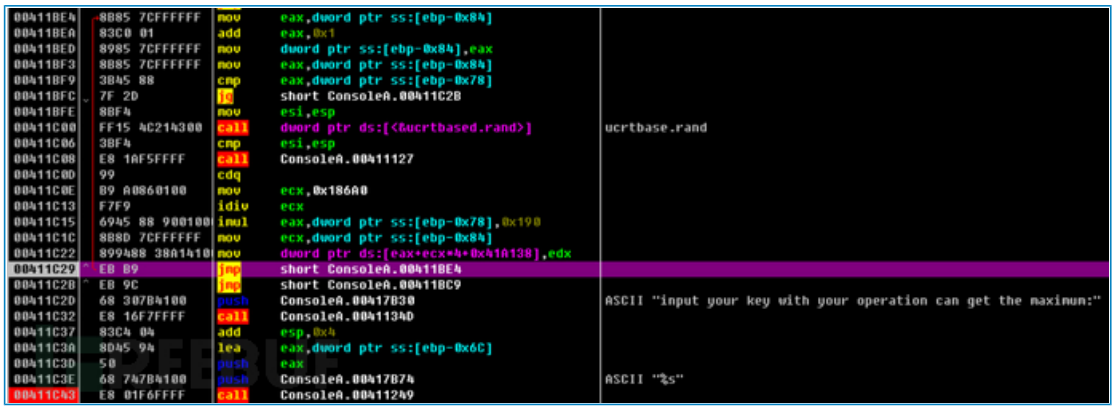
原理就是根据路径寻找字典里对应的点的值，然后相加，每条路径都产生一个值，不断的比较，只取最大的那个就行。

一阵开心地等待过后有了结果，然后根据最开始的对字符串进行处理的方式逆回去，提交答案以后又给了我当头一棒，我是有多天真！！！为什么不对？后来我把Python跑出来的结果用题目的exe跑了一遍，发现得到的数字不一样.....郁闷。为啥不一样？同样是rand函数，固定srand。痛定思痛过后反思到了也许是不同的平台就不一样，然而我还是不想在电脑上装VC。

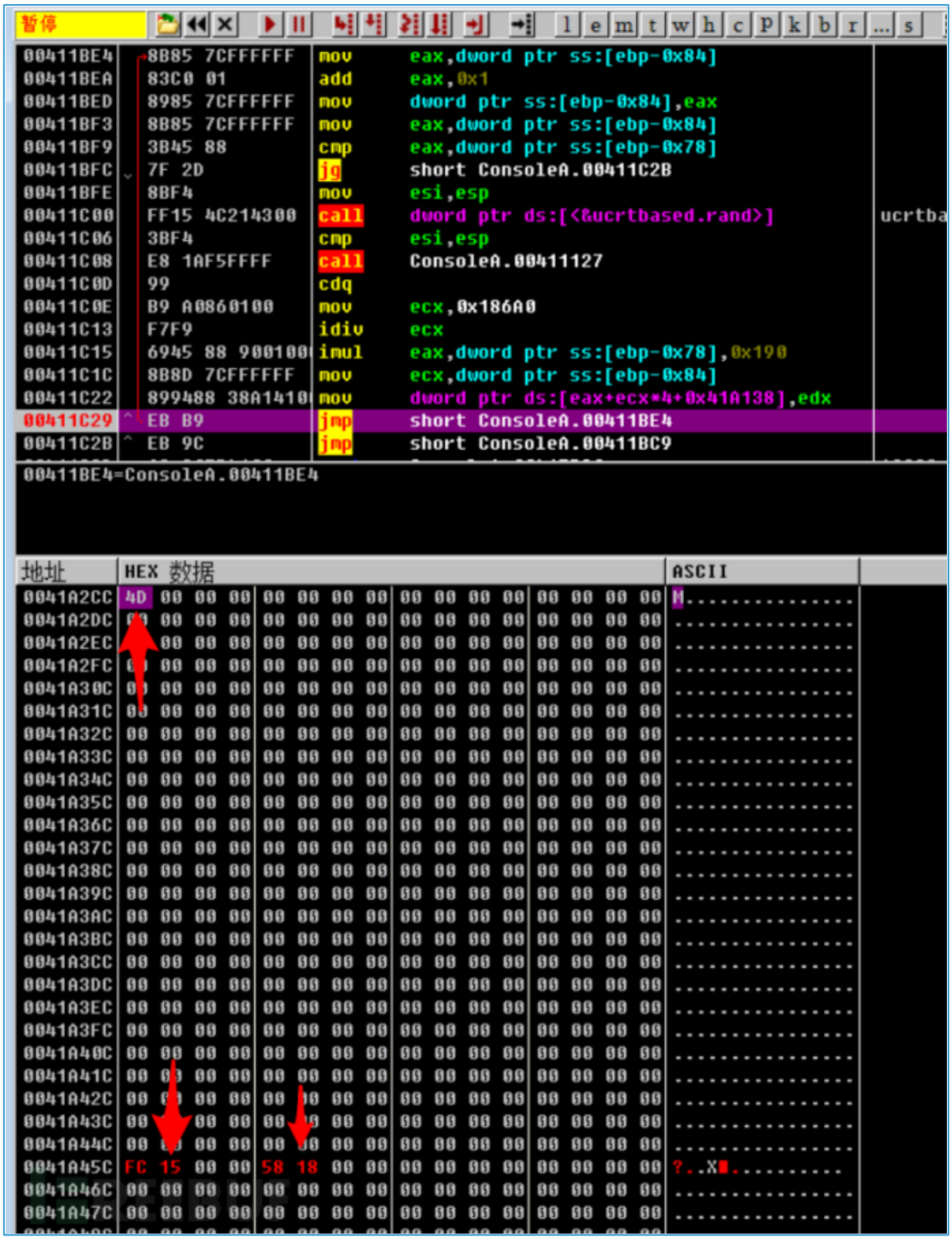
那怎么办？

让程序自己吐出来这个数组！

然后重新用OD运行这个exe，不过这次用F8走，为的是找到数组生成过程中产生的数字，于是我来到了这里：



图里面这个不断的跳转就是在生成我们需要的这个数组，我们在411C29这个地方下断点，并且每次查看对应地址的内容：



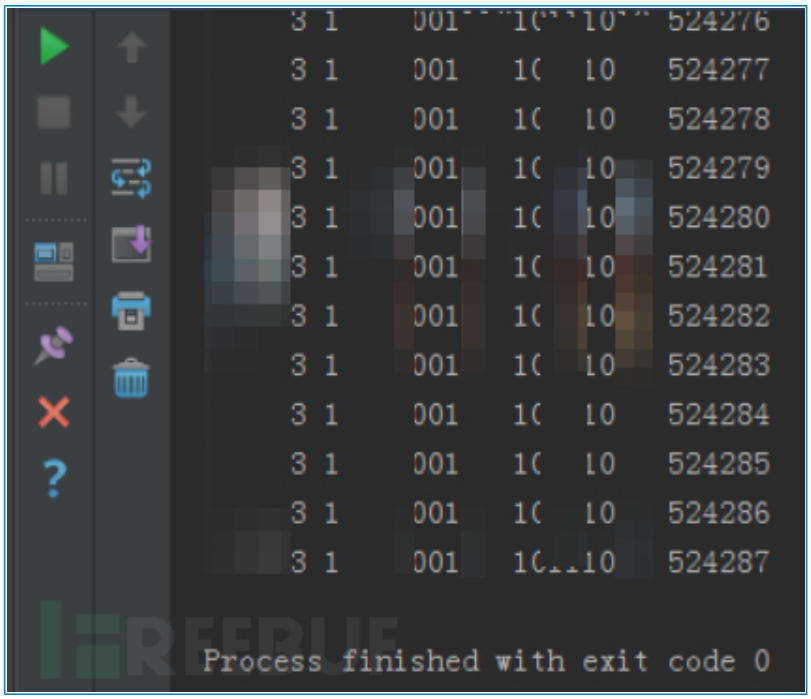
这就是我们需要的数组，总不能再跟210次，再写210个字典里的值吧？于是经过再次使用搜索引擎的技能，学会了条件断点记录到日志，这样，能把每次经过411C22这个点时的edx的值记录在案！！！！

顺便一说，在Python里的文件读写，我还是会一点的.....

记录的日志大概是这个样子的：

```
00411C22  COND: 0000004D
00411C22  COND: 000015FC
00411C22  COND: 00001858
00411C22  COND: 0000717C
00411C22  COND: 00000616
00411C22  COND: 00006626
00411C22  COND: 00003293
00411C22  COND: 000074E6
00411C22  COND: 00002ECD
00411C22  COND: 00005763
00411C22  COND: 00000FEE
00411C22  COND: 00006FD5
00411C22  COND: 00001239
00411C22  COND: 000008B5
00411C22  COND: 0000607B
00411C22  COND: 00006AEA
00411C22  COND: 00000C09
00411C22  COND: 0000465C
00411C22  COND: 00006185
00411C22  COND: 00000810
00411C22  COND: 0000690A
00411C22  COND: 0000523E
00411C22  COND: 00001469
00411C22  COND: 00002E01
00411C22  COND: 0000749D
00411C22  COND: 00000B8C
00411C22  COND: 000057A7
00411C22  COND: 00000D0D
```

经过这次的改变，我的Python脚本再次跑了起来，得到如下结果：



然后根据那一连串的1001去反推回L和R，再反推回V和H，于是这次用我的字符串跑exe以后，exe跑出来的结果和我的Python一样了。然后提交分数，FFFFF＊＊＊＊＊＊＊＊KKKKKKKK！！！！！！还不对！！！为啥？我都找到最大的路了，得到最大的分数了！

然后，我笑了，脑子高兴糊涂了，我提交了我的路径，正确！

七天过去了，做逆向真是打发时间的好帮手～

相信聪明的你已经看到了我所有犯过的错误，并且已经知道了我写的是哪里的writeup.....

转载于:https://www.cnblogs.com/h2zZhou/p/8064099.html