**汇编语言与逆向技术实验报告**

**Lab5 - Reverse Engineering Challenge**

**学号：2112492 姓名：刘修铭 专业：信息安全**

**一、实验目的**

1.熟悉静态反汇编工具IDA Freeware；

2.熟悉反汇编代码的逆向分析过程；

3.掌握反汇编语言中的数学计算、数据结构、条件判断、分支结构的识别和逆向分析

**二、实验原理**

1.通过IDA可以得到二进制代码的反汇编代码，如图1和图2所示。

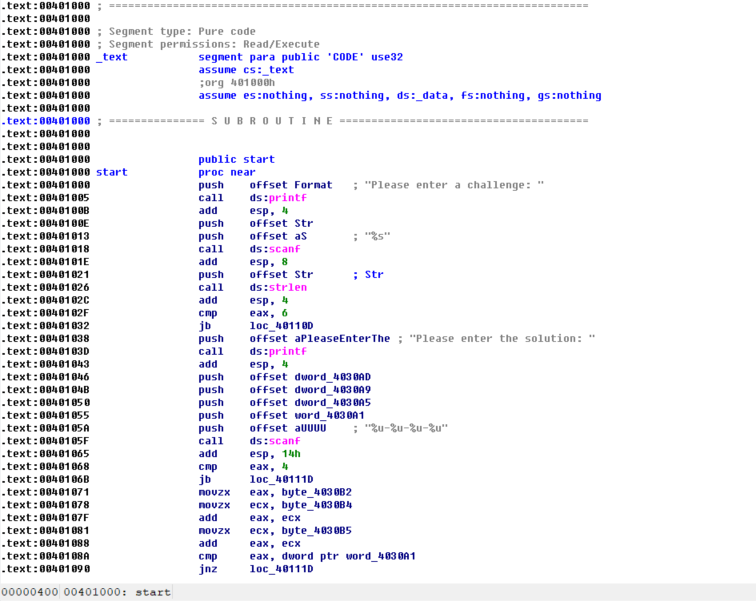


图 1 challenge.exe的反汇编代码

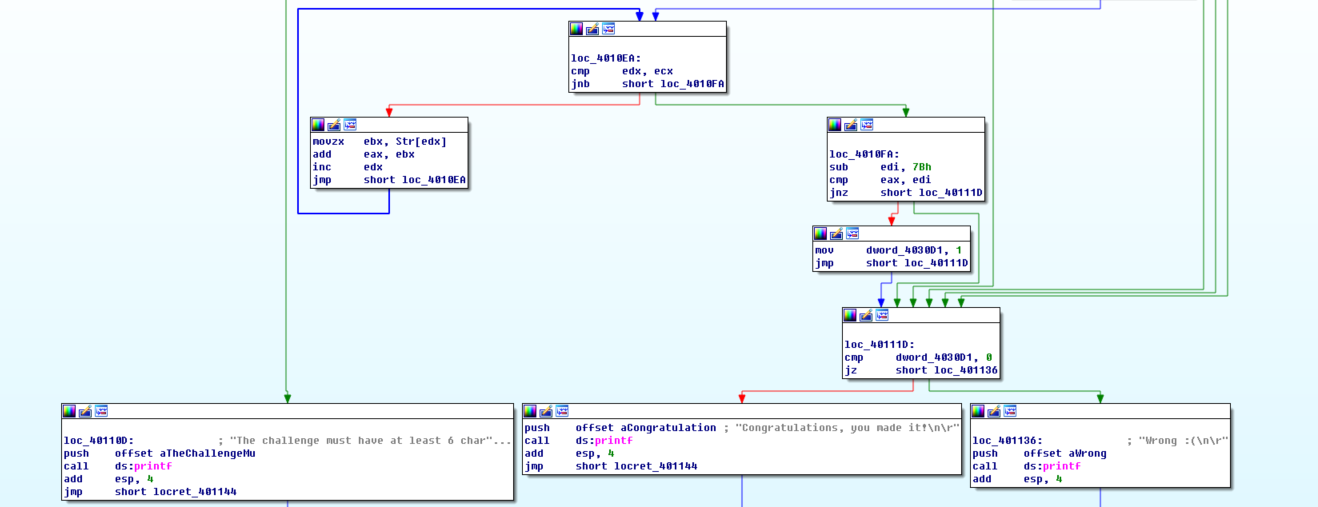


图 2 challenge.exe的反汇编代码的图形化显示

2.不修改二进制代码，分析汇编代码的计算过程、条件判断、分支结构等信息，逆向推理出程序的正确输入数据，完成逆向分析挑战。

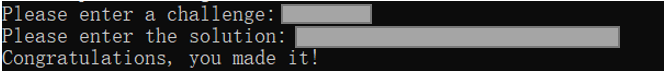
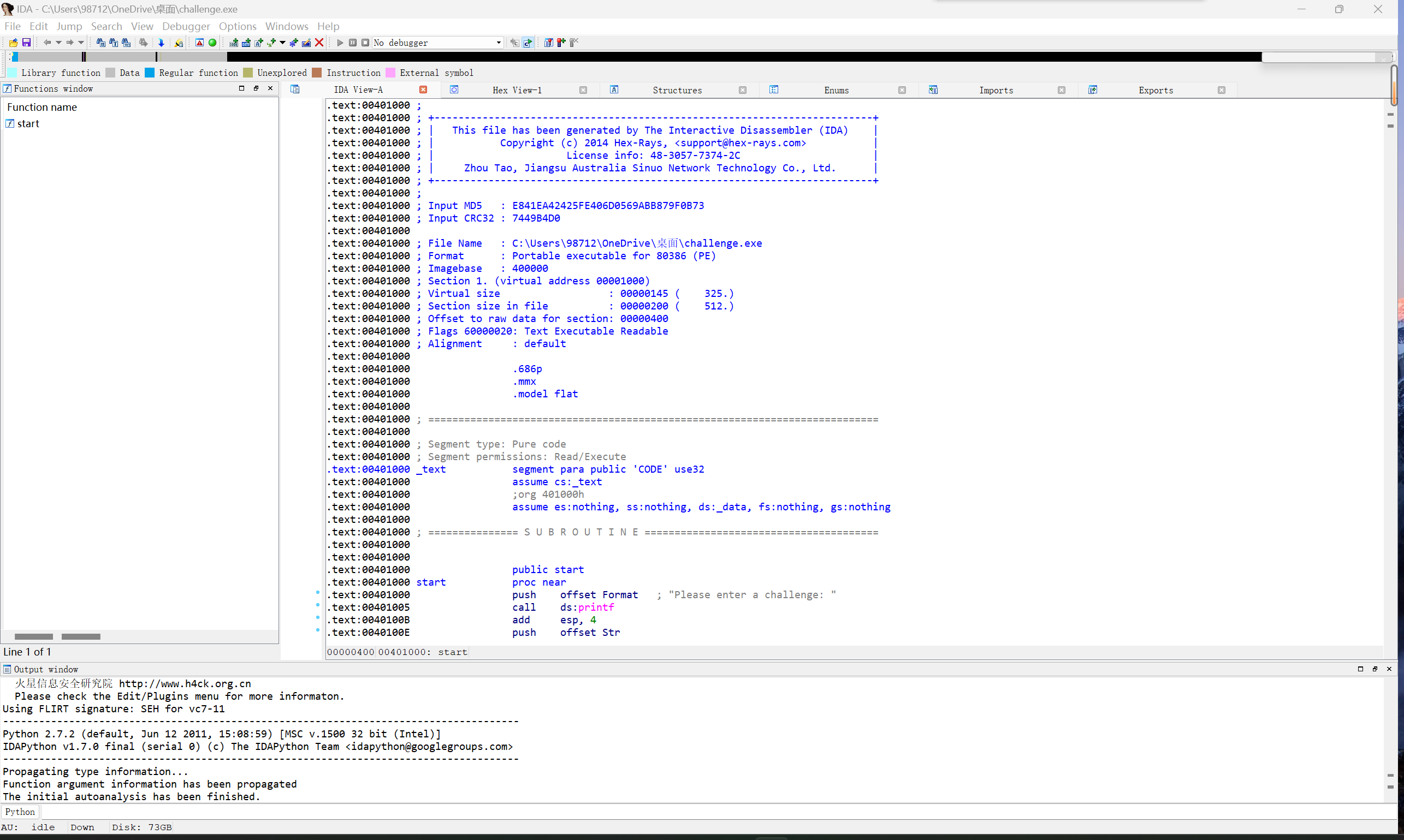
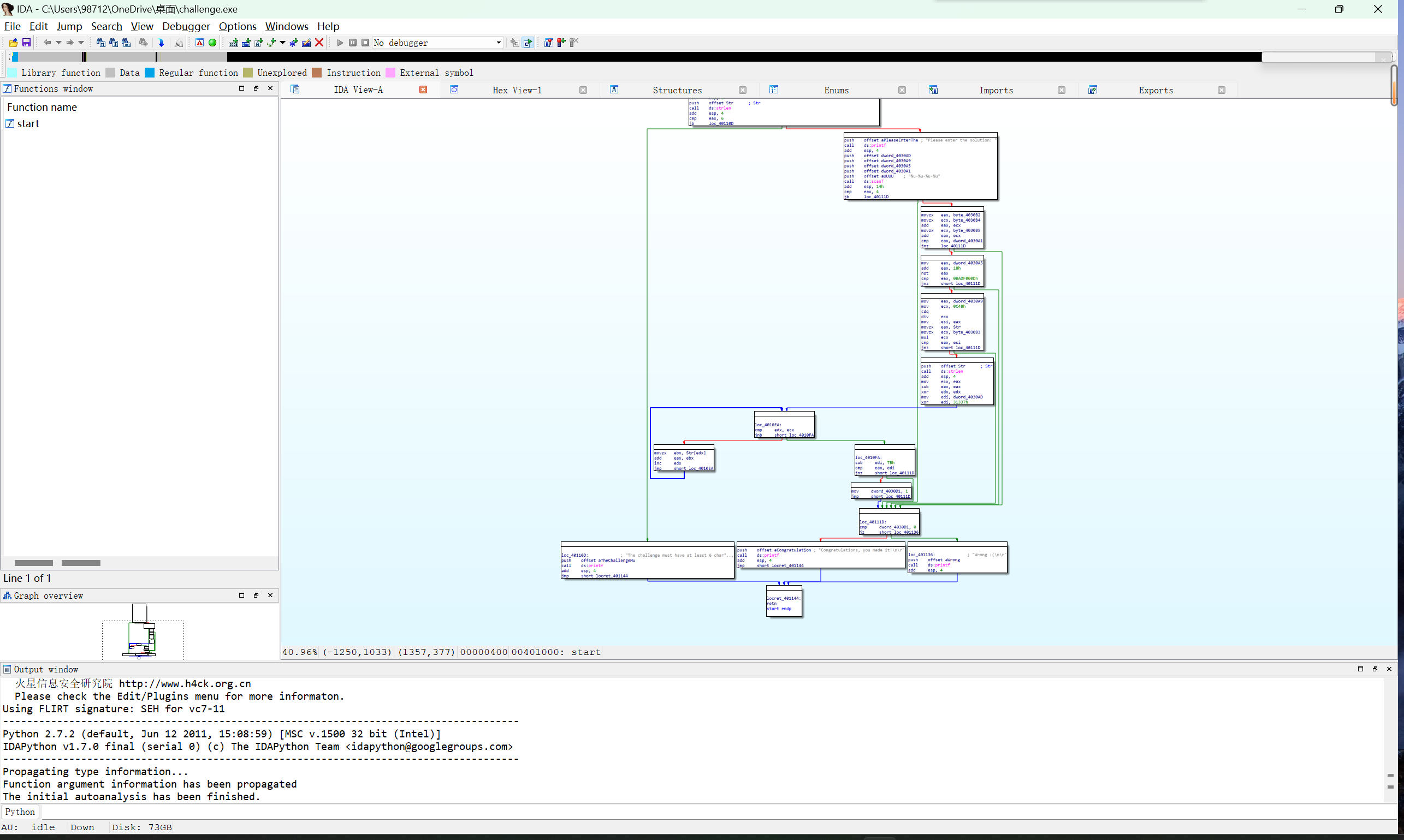


图 3 逆向分析，完成挑战

**三、获取反汇编代码**

通过IDA Freeware可以得到二进制代码的反汇编代码（源代码见最后）：

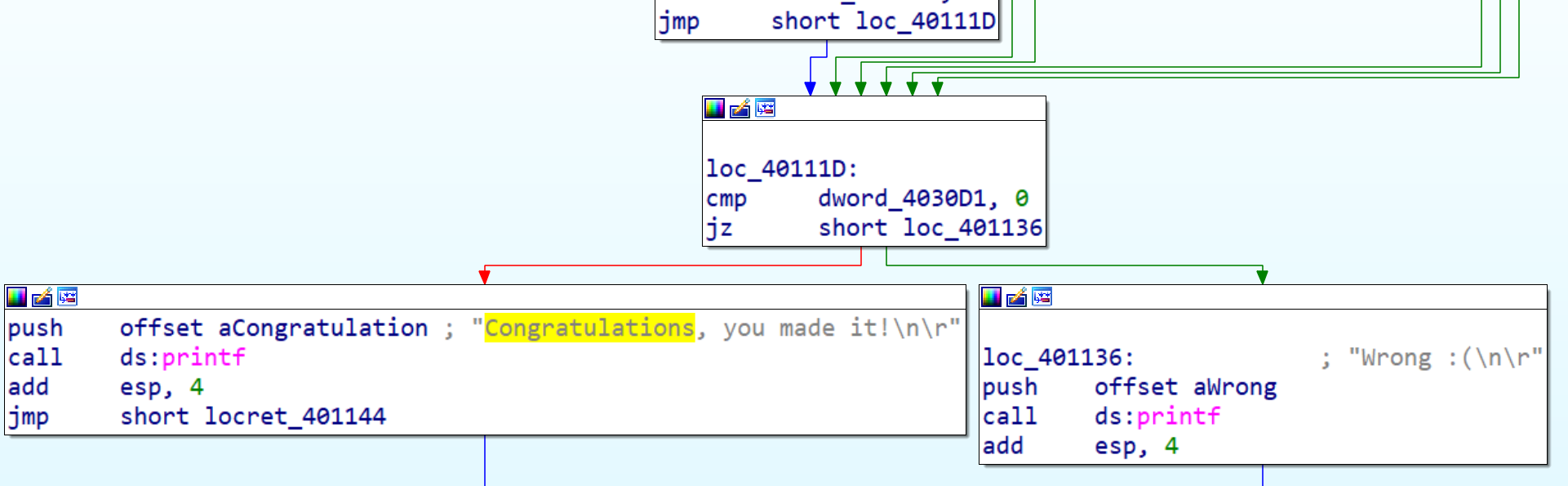
****



**四、逆向过程分析**

**从成功输出的结果“Congratulations，you made it！”逆推其所需条件**

由分支结构可知，要实现目标字符串的输出，需要满足指向目标字符串所在代码块的几个代码块的条件。



下面来依次实现该若干代码块的条件：

**1.“Please enter a challenge” 输入字符串**



此代码块的功能是：

* 调用printf函数，输出“Please enter a challenge” ，提示用户输入字符串
* 调用scanf函数，输入字符串，并存入Str
* 调用strlen函数，获取上一步输入字符串的长度，并将其和6比较
* 输入的字符串长度为6，进入下一步；否则报错"The challenge must have at least 6 char"

**因此推出：首先需要输入一个6位的字符串：**

为简单起见，本程序选择Str为111111

**2.“Please enter the solution:” 输入"%u-%u-%u-%u"型数字**



此代码块的功能是：

* 调用printf函数，输出“Please enter the solution:” ，提示用户输入solution
* 调用scanf函数，输入"%u-%u-%u-%u"型数字，并依次存入dword\_4030A1、 dword\_4030A5、 dword\_4030A9、 dword\_4030AD为起始的数据段

**因此推出：其次答案需要输入一个满足条件的"%u-%u-%u-%u"，即答案的格式为"%u-%u-%u-%u"**

**3.第一个%u**



此代码块的功能是：

* 先将用户第一次输入的字符串Str的第2位的ASCII码存入寄存器eax
* 再将Str的第4位的ASCII码存入寄存器ecx
* 将eax与ecx相加（2、4位ASCII相加）
* 将ecx与Str的第5位的ASCII码相加
* 将ecx相加之后的结果加到eax上（此时eax就是2、4、5位ASCII之和）
* 比较第一个%u与eax的大小

**因此推出：第一个数值是Str的2、4、5位ASCII之和**

计算得到 49+49+49=147

第一个数值为 147

**4.第二个%u.**



此代码块的功能是：

* 首先，让第二个%u加上18h
* 然后对其取反
* 把取反后的结果与0BADF000Dh比较

**因此推出：第二个数值是0BADF000Dh取反之后减去18h**

计算得到 (0BADF000Dh) 3135176717——>(取反)1159790578——>(减18h)1159790554

第二个数值为 1159790554

**5.第三个%u**



此代码块的功能是：

* 首先，让第三个%u除以0C48h，存入esi里
* 然后，令Str的第1位和第3位的ASCII相乘存入eax里
* 比较 esi (第三个%u除以0C48h) 和 eax（Str的第1位和第3位的ASCII相乘）

**因此推出：第三个数值是 Str第1、3位ASCII相乘 再乘上0C48h**

计算得到49\*49\*0C48h=7548744

第三个数值为7548744

**6.第四个%u**



此代码块的功能是：

* 将第四个%u与31337h异或，存入edi
* 按照Str的位数进行循环（共6位），并将edx初始置0，作为循环变量
* 将每次循环中的Str[edx] 加到eax（eax存放Str六位的ASCII之和）
* 将edi（第四个%u与31337h异或）的值减去7Bh 与 eax（Str六位的ASCII之和）比较

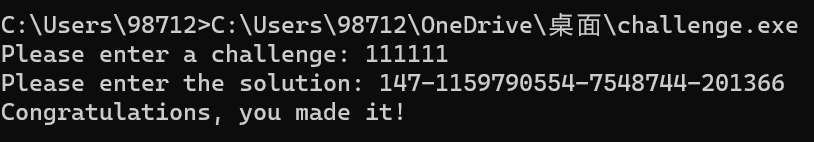
**因此推出：第四个数值与31337h异或之后 = Str六位ASCII之和 + 7Bh**

计算得到49\*6+7Bh=417（1 1010 0001）

201366（11 0001 0010 1001 0110）与31337h（11 0001 0011 0011 0111）异或之后 = 417（1 1010 0001）

第四个数值为201366

**综上，以输入Str为 111111 为例时，再输入 147-1159790554-7548744-201366 则输出“Congratulations，you made it！”表示成功。如图**

****

**五、逆向源代码**

.text:00401000 ;

.text:00401000 ; +-------------------------------------------------------------------------+

.text:00401000 ; |   This file has been generated by The Interactive Disassembler (IDA)    |

.text:00401000 ; |           Copyright (c) 2014 Hex-Rays, <support@hex-rays.com>           |

.text:00401000 ; |                      License info: 48-3057-7374-2C                      |

.text:00401000 ; |     Zhou Tao, Jiangsu Australia Sinuo Network Technology Co., Ltd.      |

.text:00401000 ; +-------------------------------------------------------------------------+

.text:00401000 ;

.text:00401000 ; Input MD5   : E841EA42425FE406D0569ABB879F0B73

.text:00401000 ; Input CRC32 : 7449B4D0

.text:00401000

.text:00401000 ; File Name   : C:\Users\98712\OneDrive\桌面\challenge.exe

.text:00401000 ; Format      : Portable executable for 80386 (PE)

.text:00401000 ; Imagebase   : 400000

.text:00401000 ; Section 1. (virtual address 00001000)

.text:00401000 ; Virtual size                  : 00000145 (    325.)

.text:00401000 ; Section size in file          : 00000200 (    512.)

.text:00401000 ; Offset to raw data for section: 00000400

.text:00401000 ; Flags 60000020: Text Executable Readable

.text:00401000 ; Alignment     : default

.text:00401000

.text:00401000                 .686p

.text:00401000                 .mmx

.text:00401000                 .model flat

.text:00401000

.text:00401000 ; ===========================================================================

.text:00401000

.text:00401000 ; Segment type: Pure code

.text:00401000 ; Segment permissions: Read/Execute

.text:00401000 \_text           segment para public 'CODE' use32

.text:00401000                 assume cs:\_text

.text:00401000                 ;org 401000h

.text:00401000                 assume es:nothing, ss:nothing, ds:\_data, fs:nothing, gs:nothing

.text:00401000

.text:00401000 ; =============== S U B R O U T I N E =======================================

.text:00401000

.text:00401000

.text:00401000                 public start

.text:00401000 start           proc near

.text:00401000                 push    offset Format   ; "Please enter a challenge: "

.text:00401005                 call    ds:printf

.text:0040100B                 add     esp, 4

.text:0040100E                 push    offset Str

.text:00401013                 push    offset aS       ; "%s"

.text:00401018                 call    ds:scanf

.text:0040101E                 add     esp, 8

.text:00401021                 push    offset Str      ; Str

.text:00401026                 call    ds:strlen

.text:0040102C                 add     esp, 4

.text:0040102F                 cmp     eax, 6

.text:00401032                 jb      loc\_40110D

.text:00401038                 push    offset aPleaseEnterThe ; "Please enter the solution: "

.text:0040103D                 call    ds:printf

.text:00401043                 add     esp, 4

.text:00401046                 push    offset dword\_4030AD

.text:0040104B                 push    offset dword\_4030A9

.text:00401050                 push    offset dword\_4030A5

.text:00401055                 push    offset dword\_4030A1

.text:0040105A                 push    offset aUUUU    ; "%u-%u-%u-%u"

.text:0040105F                 call    ds:scanf

.text:00401065                 add     esp, 14h

.text:00401068                 cmp     eax, 4

.text:0040106B                 jb      loc\_40111D

.text:00401071                 movzx   eax, byte\_4030B2

.text:00401078                 movzx   ecx, byte\_4030B4

.text:0040107F                 add     eax, ecx

.text:00401081                 movzx   ecx, byte\_4030B5

.text:00401088                 add     eax, ecx

.text:0040108A                 cmp     eax, dword\_4030A1

.text:00401090                 jnz     loc\_40111D

.text:00401096                 mov     eax, dword\_4030A5

.text:0040109B                 add     eax, 18h

.text:0040109E                 not     eax

.text:004010A0                 cmp     eax, 0BADF000Dh

.text:004010A5                 jnz     short loc\_40111D

.text:004010A7                 mov     eax, dword\_4030A9

.text:004010AC                 mov     ecx, 0C48h

.text:004010B1                 cdq

.text:004010B2                 div     ecx

.text:004010B4                 mov     esi, eax

.text:004010B6                 movzx   eax, Str

.text:004010BD                 movzx   ecx, byte\_4030B3

.text:004010C4                 mul     ecx

.text:004010C6                 cmp     eax, esi

.text:004010C8                 jnz     short loc\_40111D

.text:004010CA                 push    offset Str      ; Str

.text:004010CF                 call    ds:strlen

.text:004010D5                 add     esp, 4

.text:004010D8                 mov     ecx, eax

.text:004010DA                 sub     eax, eax

.text:004010DC                 xor     edx, edx

.text:004010DE                 mov     edi, dword\_4030AD

.text:004010E4                 xor     edi, 31337h

.text:004010EA

.text:004010EA loc\_4010EA:                             ; CODE XREF: start+F8 j

.text:004010EA                 cmp     edx, ecx

.text:004010EC                 jnb     short loc\_4010FA

.text:004010EE                 movzx   ebx, Str[edx]

.text:004010F5                 add     eax, ebx

.text:004010F7                 inc     edx

.text:004010F8                 jmp     short loc\_4010EA

.text:004010FA ; ---------------------------------------------------------------------------

.text:004010FA

.text:004010FA loc\_4010FA:                             ; CODE XREF: start+EC j

.text:004010FA                 sub     edi, 7Bh

.text:004010FD                 cmp     eax, edi

.text:004010FF                 jnz     short loc\_40111D

.text:00401101                 mov     dword\_4030D1, 1

.text:0040110B                 jmp     short loc\_40111D

.text:0040110D ; ---------------------------------------------------------------------------

.text:0040110D

.text:0040110D loc\_40110D:                             ; CODE XREF: start+32 j

.text:0040110D                 push    offset aTheChallengeMu ; "The challenge must have at least 6 char"...

.text:00401112                 call    ds:printf

.text:00401118                 add     esp, 4

.text:0040111B                 jmp     short locret\_401144

.text:0040111D ; ---------------------------------------------------------------------------

.text:0040111D

.text:0040111D loc\_40111D:                             ; CODE XREF: start+6B j

.text:0040111D                                         ; start+90 j ...

.text:0040111D                 cmp     dword\_4030D1, 0

.text:00401124                 jz      short loc\_401136

.text:00401126                 push    offset aCongratulation ; "Congratulations, you made it!\n\r"

.text:0040112B                 call    ds:printf

.text:00401131                 add     esp, 4

.text:00401134                 jmp     short locret\_401144

.text:00401136 ; ---------------------------------------------------------------------------

.text:00401136

.text:00401136 loc\_401136:                             ; CODE XREF: start+124 j

.text:00401136                 push    offset aWrong   ; "Wrong :(\n\r"

.text:0040113B                 call    ds:printf

.text:00401141                 add     esp, 4

.text:00401144

.text:00401144 locret\_401144:                          ; CODE XREF: start+11B j

.text:00401144                                         ; start+134 j

.text:00401144                 retn

.text:00401144 start           endp

.text:00401144

.text:00401144 ; ---------------------------------------------------------------------------

.text:00401145                 align 100h

.text:00401200                 dd 380h dup(?)

.text:00401200 \_text           ends

.text:00401200

.idata:00402000 ; Section 2. (virtual address 00002000)

.idata:00402000 ; Virtual size                  : 00000070 (    112.)

.idata:00402000 ; Section size in file          : 00000200 (    512.)

.idata:00402000 ; Offset to raw data for section: 00000600

.idata:00402000 ; Flags 40000040: Data Readable

.idata:00402000 ; Alignment     : default

.idata:00402000 ;

.idata:00402000 ; Imports from msvcrt.dll

.idata:00402000 ;

.idata:00402000 ; ===========================================================================

.idata:00402000

.idata:00402000 ; Segment type: Externs

.idata:00402000 ; \_idata

.idata:00402000 ; int scanf(const char \*Format, ...)

.idata:00402000                 extrn scanf:dword       ; CODE XREF: start+18 p

.idata:00402000                                         ; start+5F p

.idata:00402000                                         ; DATA XREF: ...

.idata:00402004 ; size\_t \_\_cdecl strlen(const char \*Str)

.idata:00402004                 extrn strlen:dword      ; CODE XREF: start+26 p

.idata:00402004                                         ; start+CF p

.idata:00402004                                         ; DATA XREF: ...

.idata:00402008 ; int printf(const char \*Format, ...)

.idata:00402008                 extrn printf:dword      ; CODE XREF: start+5 p

.idata:00402008                                         ; start+3D p ...

.idata:0040200C

.idata:0040200C

.rdata:00402010 ; ===========================================================================

.rdata:00402010

.rdata:00402010 ; Segment type: Pure data

.rdata:00402010 ; Segment permissions: Read

.rdata:00402010 \_rdata          segment para public 'DATA' use32

.rdata:00402010                 assume cs:\_rdata

.rdata:00402010                 ;org 402010h

.rdata:00402010 \_\_IMPORT\_DESCRIPTOR\_msvcrt dd rva off\_402038 ; Import Name Table

.rdata:00402014                 dd 0                    ; Time stamp

.rdata:00402018                 dd 0                    ; Forwarder Chain

.rdata:0040201C                 dd rva aMsvcrt\_dll      ; DLL Name

.rdata:00402020                 dd rva scanf            ; Import Address Table

.rdata:00402024                 db    0

.rdata:00402025                 db    0

.rdata:00402026                 db    0

.rdata:00402027                 db    0

.rdata:00402028                 db    0

.rdata:00402029                 db    0

.rdata:0040202A                 db    0

.rdata:0040202B                 db    0

.rdata:0040202C                 db    0

.rdata:0040202D                 db    0

.rdata:0040202E                 db    0

.rdata:0040202F                 db    0

.rdata:00402030                 db    0

.rdata:00402031                 db    0

.rdata:00402032                 db    0

.rdata:00402033                 db    0

.rdata:00402034                 db    0

.rdata:00402035                 db    0

.rdata:00402036                 db    0

.rdata:00402037                 db    0

.rdata:00402038 ;

.rdata:00402038 ; Import names for msvcrt.dll

.rdata:00402038 ;

.rdata:00402038 off\_402038      dd rva word\_402052      ; DATA XREF: .rdata:\_\_IMPORT\_DESCRIPTOR\_msvcrt o

.rdata:0040203C                 dd rva word\_40205A

.rdata:00402040                 dd rva word\_402048

.rdata:00402044                 dd 0

.rdata:00402048 word\_402048     dw 281h                 ; DATA XREF: .rdata:00402040 o

.rdata:0040204A                 db 'printf',0

.rdata:00402051                 align 2

.rdata:00402052 word\_402052     dw 28Eh                 ; DATA XREF: .rdata:off\_402038 o

.rdata:00402054                 db 'scanf',0

.rdata:0040205A word\_40205A     dw 2A1h                 ; DATA XREF: .rdata:0040203C o

.rdata:0040205C                 db 'strlen',0

.rdata:00402063                 align 4

.rdata:00402064 aMsvcrt\_dll     db 'msvcrt.dll',0       ; DATA XREF: .rdata:0040201C o

.rdata:0040206F                 align 1000h

.rdata:0040206F \_rdata          ends

.rdata:0040206F

.data:00403000 ; Section 3. (virtual address 00003000)

.data:00403000 ; Virtual size                  : 000000D5 (    213.)

.data:00403000 ; Section size in file          : 00000200 (    512.)

.data:00403000 ; Offset to raw data for section: 00000800

.data:00403000 ; Flags C0000040: Data Readable Writable

.data:00403000 ; Alignment     : default

.data:00403000 ; ===========================================================================

.data:00403000

.data:00403000 ; Segment type: Pure data

.data:00403000 ; Segment permissions: Read/Write

.data:00403000 \_data           segment para public 'DATA' use32

.data:00403000                 assume cs:\_data

.data:00403000                 ;org 403000h

.data:00403000 ; char Format[]

.data:00403000 Format          db 'Please enter a challenge: ',0 ; DATA XREF: start o

.data:0040301B ; char aS[]

.data:0040301B aS              db '%s',0               ; DATA XREF: start+13 o

.data:0040301E ; char aTheChallengeMu[]

.data:0040301E aTheChallengeMu db 'The challenge must have at least 6 characters',0Ah

.data:0040301E                                         ; DATA XREF: start:loc\_40110D o

.data:0040301E                 db 0Dh,0

.data:0040304E ; char aPleaseEnterThe[]

.data:0040304E aPleaseEnterThe db 'Please enter the solution: ',0 ; DATA XREF: start+38 o

.data:0040306A ; char aUUUU[]

.data:0040306A aUUUU           db '%u-%u-%u-%u',0      ; DATA XREF: start+5A o

.data:00403076 ; char aWrong[]

.data:00403076 aWrong          db 'Wrong :(',0Ah       ; DATA XREF: start:loc\_401136 o

.data:00403076                 db 0Dh,0

.data:00403081 ; char aCongratulation[]

.data:00403081 aCongratulation db 'Congratulations, you made it!',0Ah

.data:00403081                                         ; DATA XREF: start+126 o

.data:00403081                 db 0Dh,0

.data:004030A1 dword\_4030A1    dd 0                    ; DATA XREF: start+55 o

.data:004030A1                                         ; start+8A r

.data:004030A5 dword\_4030A5    dd 0                    ; DATA XREF: start+50 o

.data:004030A5                                         ; start+96 r

.data:004030A9 dword\_4030A9    dd 0                    ; DATA XREF: start+4B o

.data:004030A9                                         ; start+A7 r

.data:004030AD dword\_4030AD    dd 0                    ; DATA XREF: start+46 o

.data:004030AD                                         ; start+DE r

.data:004030B1 ; char Str

.data:004030B1 Str             db 0                    ; DATA XREF: start+E o

.data:004030B1                                         ; start+21 o ...

.data:004030B2 byte\_4030B2     db 0                    ; DATA XREF: start+71 r

.data:004030B3 byte\_4030B3     db 0                    ; DATA XREF: start+BD r

.data:004030B4 byte\_4030B4     db 0                    ; DATA XREF: start+78 r

.data:004030B5 byte\_4030B5     db 0                    ; DATA XREF: start+81 r

.data:004030B6                 db    0

.data:004030B7                 db    0

.data:004030B8                 db    0

.data:004030B9                 db    0

.data:004030BA                 db    0

.data:004030BB                 db    0

.data:004030BC                 db    0

.data:004030BD                 db    0

.data:004030BE                 db    0

.data:004030BF                 db    0

.data:004030C0                 db    0

.data:004030C1                 db    0

.data:004030C2                 db    0

.data:004030C3                 db    0

.data:004030C4                 db    0

.data:004030C5                 db    0

.data:004030C6                 db    0

.data:004030C7                 db    0

.data:004030C8                 db    0

.data:004030C9                 db    0

.data:004030CA                 db    0

.data:004030CB                 db    0

.data:004030CC                 db    0

.data:004030CD                 db    0

.data:004030CE                 db    0

.data:004030CF                 db    0

.data:004030D0                 db    0

.data:004030D1 dword\_4030D1    dd 0                    ; DATA XREF: start+101 w

.data:004030D1                                         ; start:loc\_40111D r

.data:004030D5                 align 1000h

.data:004030D5 \_data           ends

.data:004030D5

.data:004030D5

.data:004030D5                 end start