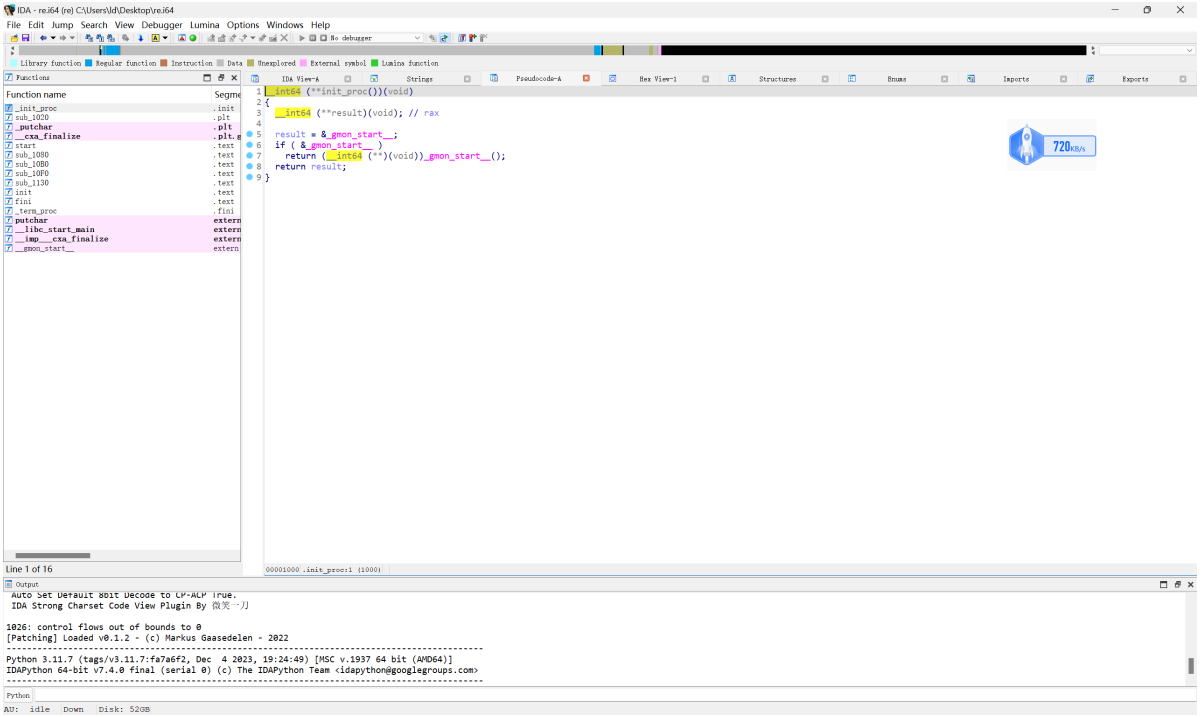
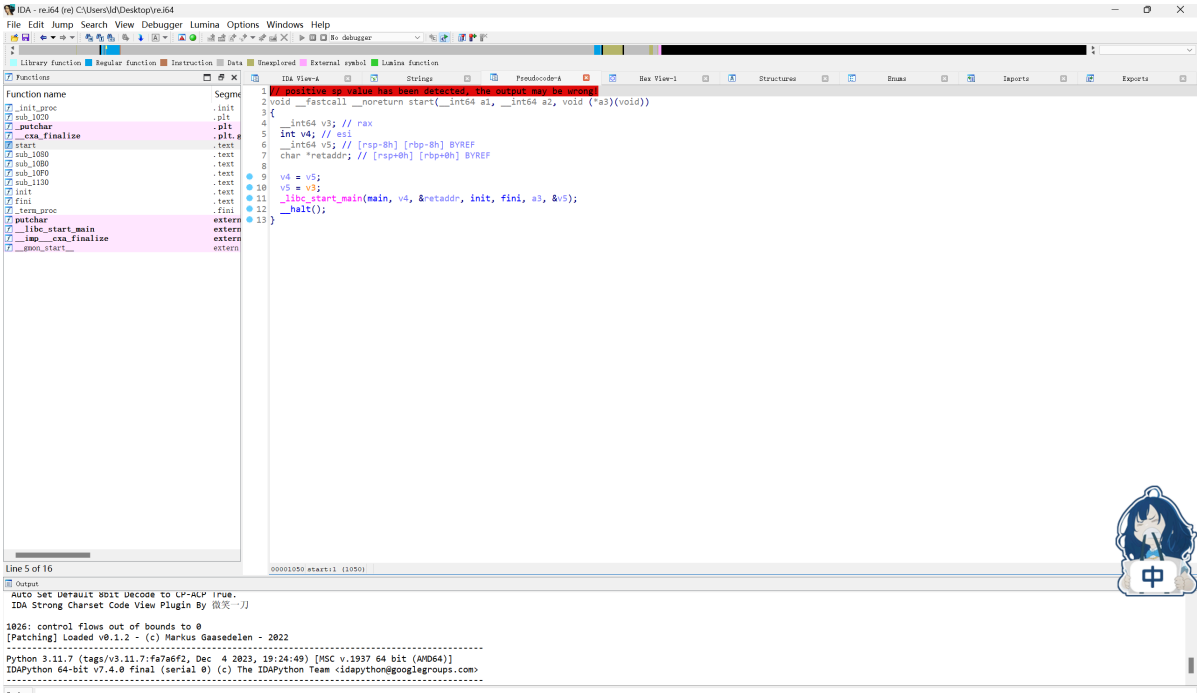
关于花指令的一道题

诗雨学姐听说我上个星期专门搞了花指令，故发了一道花指令题给我练手

下载下来以后丢die里面查看，没壳并且是64位（因为下完以后不小心误删了所以就没放图了，有意思的是这是一个elf文件）



进入主函数发现主函数（start）编译失败，猜测大概是花指令搞的鬼



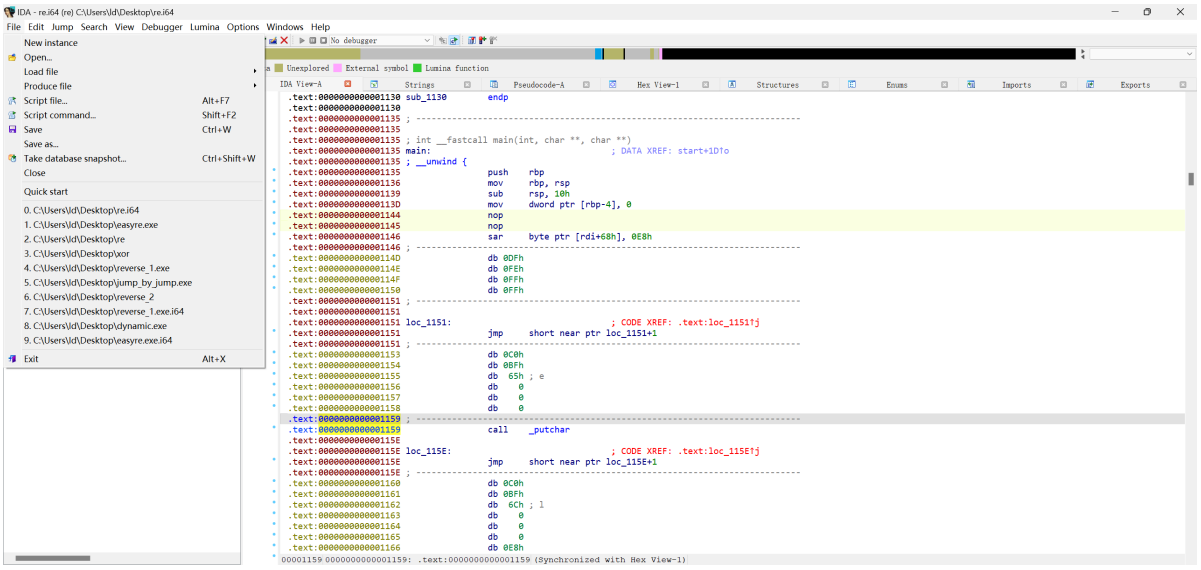
双击进入汇编界面窗口，发现了一个很明显的花指令

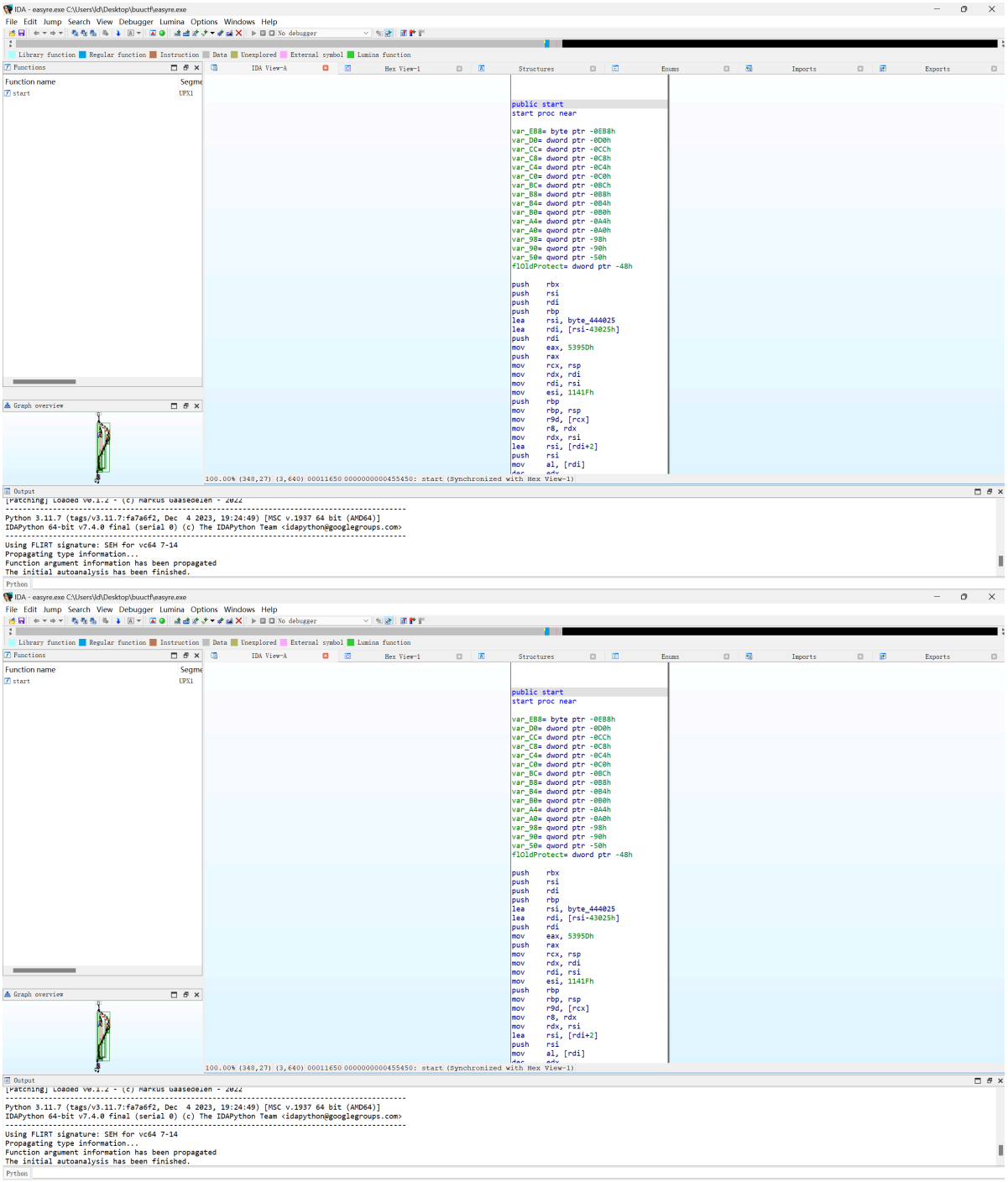
```
                ; int __fastcall main(int, char **, char **)
                main:                                   ; DATA XREF: start+1D↑o
                ; __unwind {
                push    rbp
                mov     rbp, rsp
                sub     rsp, 10h
) 00 00         mov     dword ptr [rbp-4], 0

                loc_1144:                               ; CODE XREF: .text:loc_1144↑j
                jmp     short near ptr loc_1144+1

                ; ---------------------------------------------------------------------------
                dw 0BFC0h
} DF FE FF FF EB+dq 0FFFEDFE800000068h, 65BFC0FFEBFFh, 0FFEBFFFFFED2E800h, 0C5E80000006CBFC0h, 6CBFC0
) 00 00 E8 D2 FE+dq 6FBFC0FFEBh, 0C0FFEBFFFFFEABE8h, 0FE9EE800000020BFh, 77BFC0FFEBFFFFh, 0EBFFFFFE91
) BF 6C 00 00 00+dq 0BFC0FFEBFFFFFE84h, 0FFFE77E800000072h, 6CBFC0FFEBFFh, 0FFEBFFFFFE6AE800h, 5DE800
: EB FF C0 BF 6C+dq 0FFFFFE50E8000000h, 0ABFC0FFEBh, 0C0FFEBFFFFFE43E8h, 0FE36E800000054BFh, 68BFC0FF
} FE FF FF EB FF+dq 0E800000065BFC0FFh, 0BFC0FFEBFFFFFE1Ch, 0FFFE0FE800000072h, 65BFC0FFEBFFh, 0FFEBF
) 00 E8 AB FE FF+dq 61BFC0FFEBFFFFFDh, 0FFFFFDE8E8000000h, 72BFC0FFEBh, 0C0FFEBFFFFFDDBE8h, 0FDCEE800
: 20 00 00 00 E8+dq 0EBFFFFFDC1E80000h, 0E80000006DBFC0FFh, 0BFC0FFEBFFFFFDB4h, 0FFFDA7E80000006Fh, 6
} FF C0 BF 77 00+dq 8DE800000065BFC0h, 6EBFC0FFEBFFFFFDh, 0FFFFFD80E8000000h, 74BFC0FFEBh, 0C0FFEBFFF
: FF FF EB FF C0+dq 20BFC0FFEBFFFFh, 0EBFFFFFD59E80000h, 0E800000069BFC0FFh, 0BFC0FFEBFFFFFD4Ch, 0FFF
) E8 84 FE FF FF+dq 0FFEBFFFFFD32E800h, 25E80000006CBFC0h, 69BFC0FFEBFFFFFDh, 0FFFFFD18E8000000h, 66B
00000001146: .text:0000000000001146 (Synchronized with Hex View-1)
```

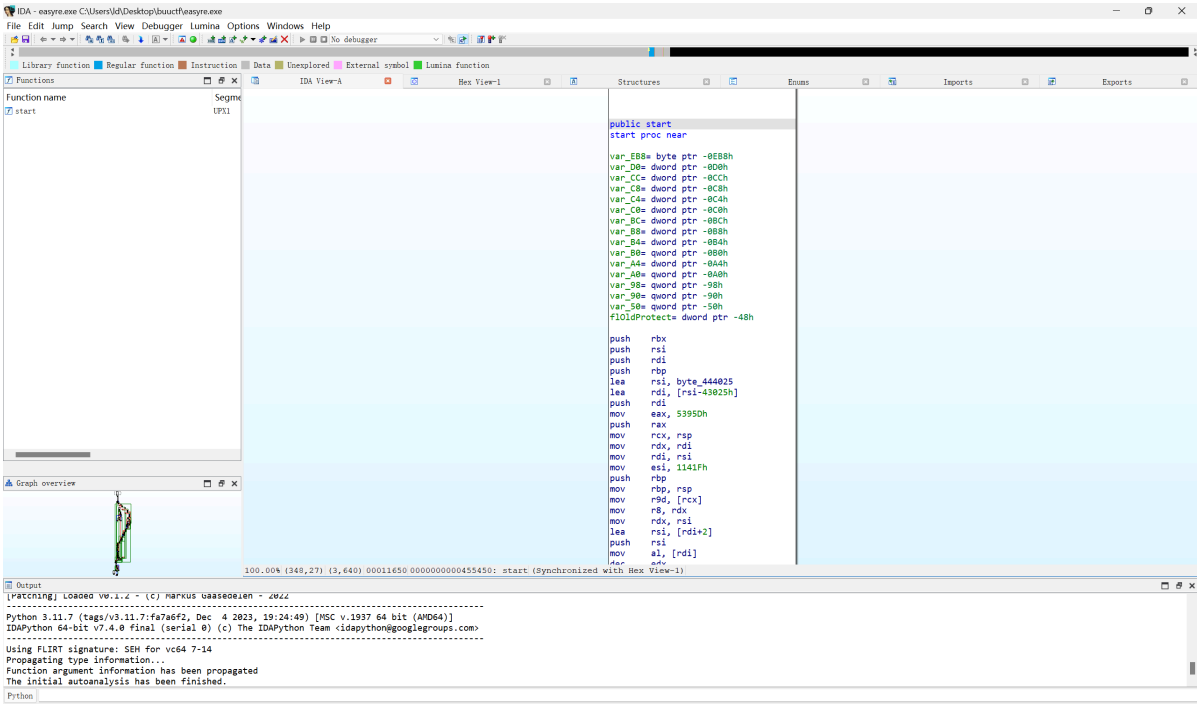

我以为只有这一个花指令，将其nop掉以后生成函数发现依然没有什么跟flag有关的线索，我按c键将下面的数据转化为代码发现有大量的与上面相同的花指令，一个个手动nop不现实，于是我就决定在ida中编写脚本来实现自动nop



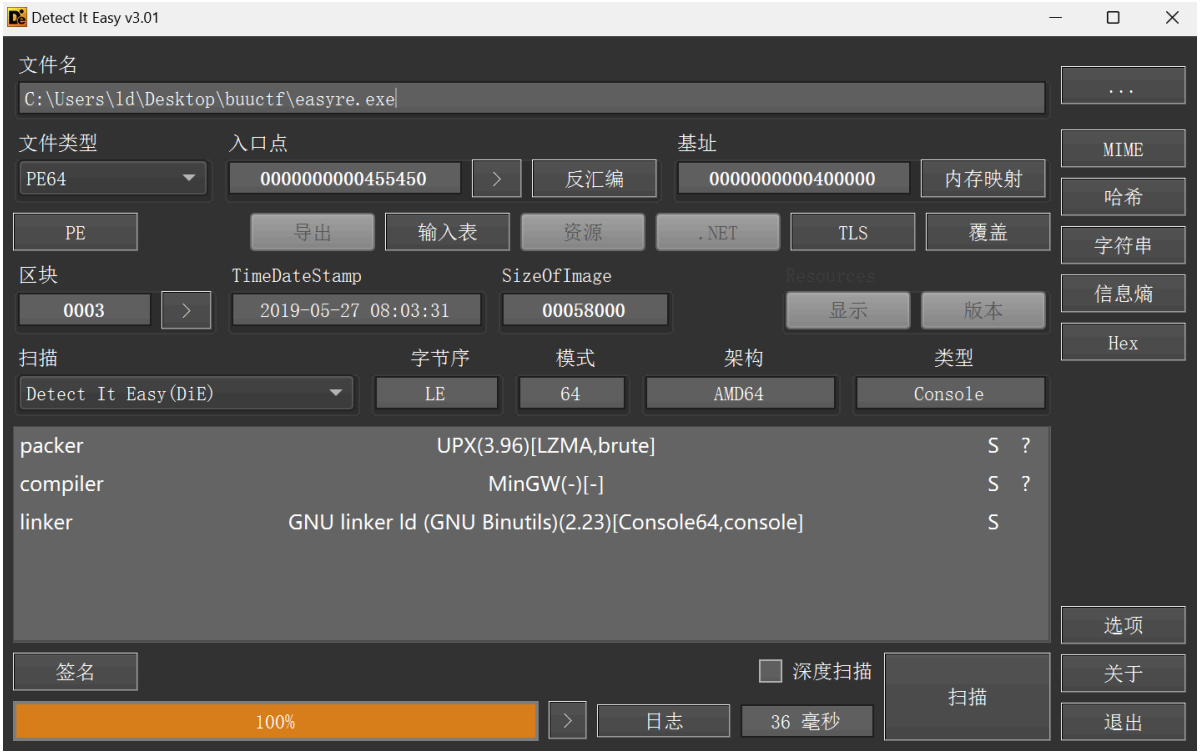


目前貌似ida只能用python敲脚本

花指令的机器码如下



编写完脚本运行后



flag就在这些灰色字母里面