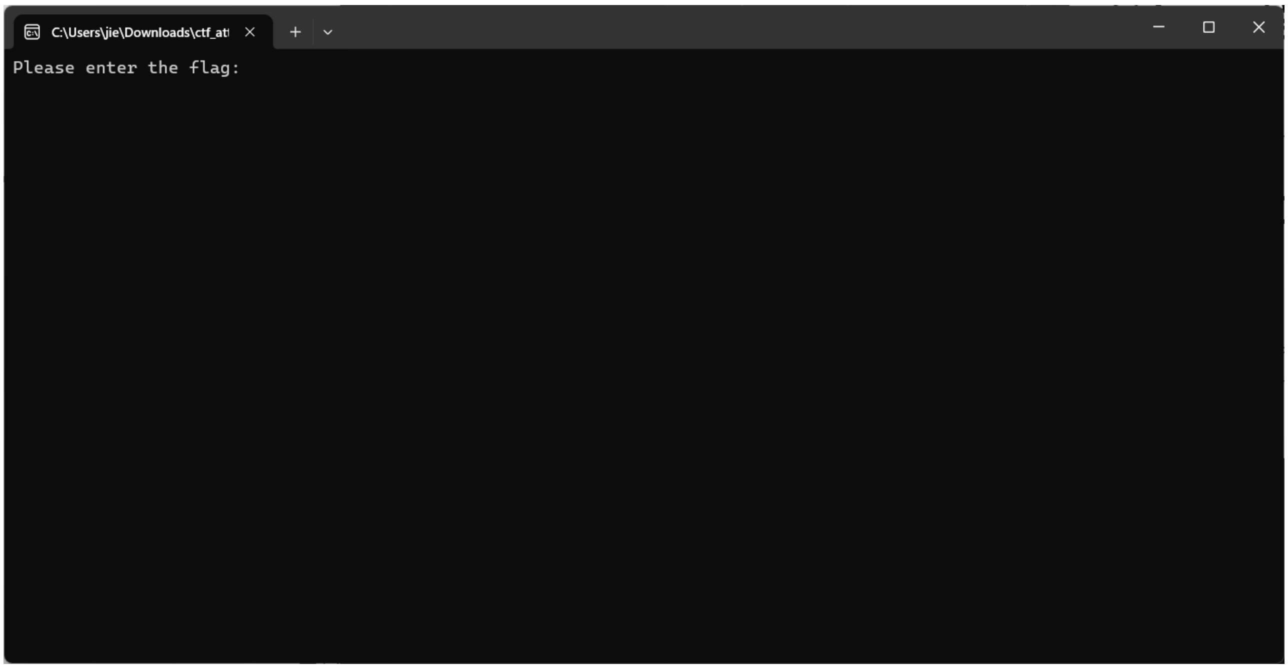
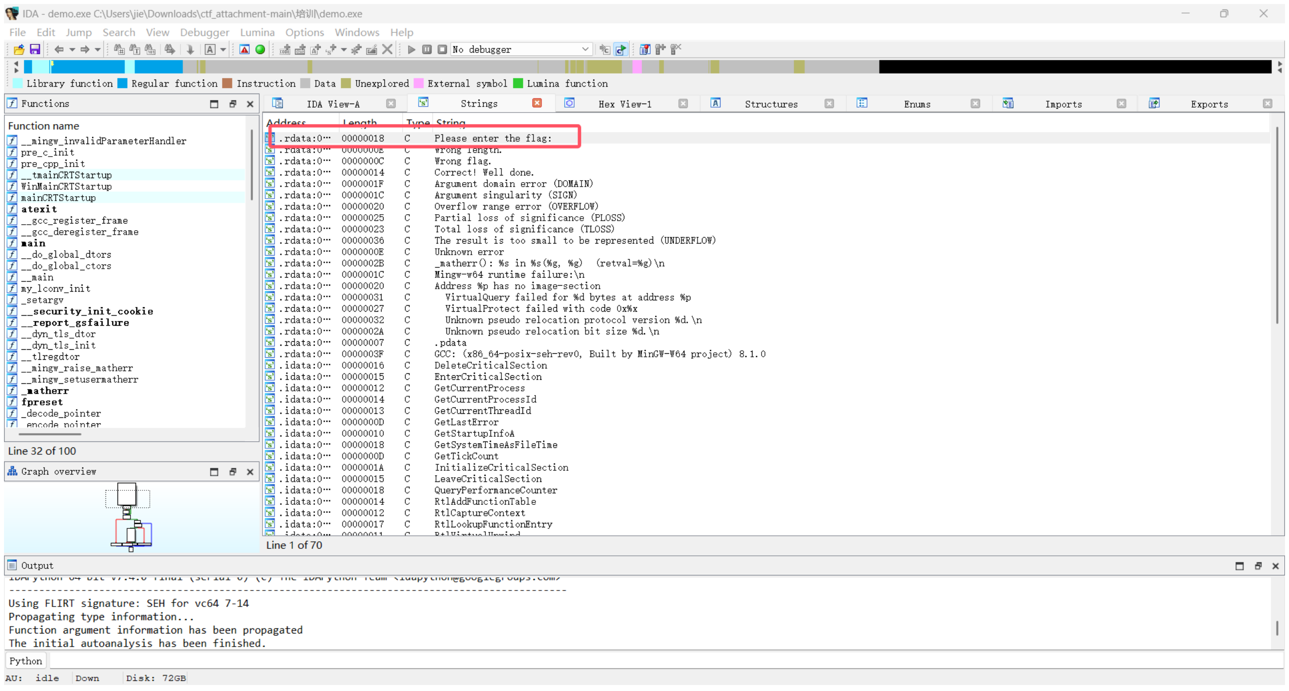
demo.exe

1. 首先打开该exe文件，发现要求输入flag


```
Please enter the flag:
```

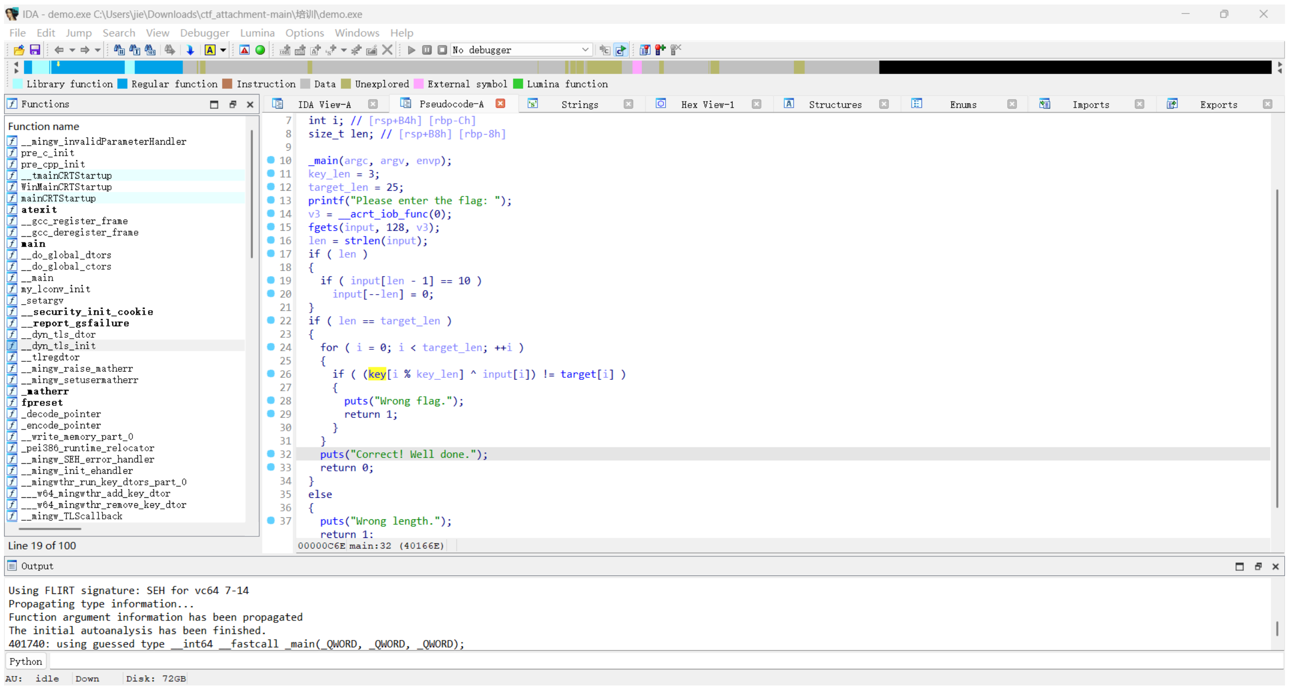
2. 然后使用IDA对其逆向，按下`Shift + F12`进入`string`界面，对string的值进行查找，从而找到`main()`函数，对`main()`函数进行分析得，该段代码进行了异或加密


IDA - demo.exe C:\Users\jie\Downloads\ctf_attachment-main\培训\demo.exe

```
Function name
__mingw_invalidParameterHandler
pre_c_init
pre_cpp_init
__tmainCRTStartup
WinMainCRTStartup
mainCRTStartup
atexit
__gcc_register_frame
__gcc_deregister_frame
main
__do_global_dtors
__do_global_ctors
__main
my_lconv_init
_setargv
__security_init_cookie
__report_gsfailure
__dyn_tls_dtor
__dyn_tls_init
__tlregdtor
__mingw_raise_matherr
__mingw_setusermatherr
_matherr
fpreset
_decode_pointer
_encode_pointer
Line 32 of 100
```

```
Address       Length    Type String
.rdata:0⋯    00000018  C    Please enter the flag:
.rdata:0⋯    0000000E  C    Wrong length.
.rdata:0⋯    0000000C  C    Wrong flag.
.rdata:0⋯    00000014  C    Correct! Well done.
.rdata:0⋯    0000001F  C    Argument domain error (DOMAIN)
.rdata:0⋯    0000001C  C    Argument singularity (SIGN)
.rdata:0⋯    00000020  C    Overflow range error (OVERFLOW)
.rdata:0⋯    00000025  C    Partial loss of significance (PLOSS)
.rdata:0⋯    00000023  C    Total loss of significance (TLOSS)
.rdata:0⋯    00000036  C    The result is too small to be represented (UNDERFLOW)
.rdata:0⋯    0000000E  C    Unknown error
.rdata:0⋯    0000002B  C    _matherr(): %s in %s(%g, %g)  (retval=%g)\n
.rdata:0⋯    0000001C  C    Mingw-w64 runtime failure:\n
.rdata:0⋯    00000020  C    Address %p has no image-section
.rdata:0⋯    00000031  C      VirtualQuery failed for %d bytes at address %p
.rdata:0⋯    00000027  C      VirtualProtect failed with code 0x%x
.rdata:0⋯    00000032  C      Unknown pseudo relocation protocol version %d.\n
.rdata:0⋯    0000002A  C      Unknown pseudo relocation bit size %d.\n
.rdata:0⋯    00000007  C    .pdata
.rdata:0⋯    0000003F  C    GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
.idata:0⋯    00000016  C    DeleteCriticalSection
.idata:0⋯    00000015  C    EnterCriticalSection
.idata:0⋯    00000012  C    GetCurrentProcess
.idata:0⋯    00000014  C    GetCurrentProcessId
.idata:0⋯    00000013  C    GetCurrentThreadId
.idata:0⋯    0000000D  C    GetLastError
.idata:0⋯    00000010  C    GetStartupInfoA
.idata:0⋯    00000018  C    GetSystemTimeAsFileTime
.idata:0⋯    0000000D  C    GetTickCount
.idata:0⋯    0000001A  C    InitializeCriticalSection
.idata:0⋯    00000015  C    LeaveCriticalSection
.idata:0⋯    00000018  C    QueryPerformanceCounter
.idata:0⋯    00000014  C    RtlAddFunctionTable
.idata:0⋯    00000012  C    RtlCaptureContext
.idata:0⋯    00000017  C    RtlLookupFunctionEntry
Line 1 of 70
```

```
Using FLIRT signature: SEH for vc64 7-14
Propagating type information...
Function argument information has been propagated
The initial autoanalysis has been finished.
```


IDA - demo.exe C:\Users\jie\Downloads\ctf_attachment-main\培训\demo.exe

```
Function name
__mingw_invalidParameterHandler
pre_c_init
pre_cpp_init
__tmainCRTStartup
WinMainCRTStartup
mainCRTStartup
atexit
__gcc_register_frame
__gcc_deregister_frame
main
__do_global_dtors
__do_global_ctors
__main
my_lconv_init
_setargv
__security_init_cookie
__report_gsfailure
__dyn_tls_dtor
__dyn_tls_init
__tlregdtor
__mingw_raise_matherr
__mingw_setusermatherr
_matherr
fpreset
_decode_pointer
_encode_pointer
__write_memory_part_0
_pei386_runtime_relocator
__mingw_SEH_error_handler
__mingw_init_ehandler
__mingwthr_run_key_dtors_part_0
___w64_mingwthr_add_key_dtor
___w64_mingwthr_remove_key_dtor
__mingw_TLScallback
Line 19 of 100
```

```c
  int i; // [rsp+B4h] [rbp-Ch]
  size_t len; // [rsp+B8h] [rbp-8h]

  _main(argc, argv, envp);
  key_len = 3;
  target_len = 25;
  printf("Please enter the flag: ");
  v3 = __acrt_iob_func(0);
  fgets(input, 128, v3);
  len = strlen(input);
  if ( len )
  {
    if ( input[len - 1] == 10 )
      input[--len] = 0;
  }
  if ( len == target_len )
  {
    for ( i = 0; i < target_len; ++i )
    {
      if ( (key[i % key_len] ^ input[i]) != target[i] )
      {
        puts("Wrong flag.");
        return 1;
      }
    }
    puts("Correct! Well done.");
    return 0;
  }
  else
  {
    puts("Wrong length.");
    return 1:
00000C6E main:32 (40166E)
```

```
Using FLIRT signature: SEH for vc64 7-14
Propagating type information...
Function argument information has been propagated
The initial autoanalysis has been finished.
401740: using guessed type __int64 __fastcall _main(_QWORD, _QWORD, _QWORD);
```

3. 于是，对密文与密钥进行了寻找，发现了代表密文的`target`与代表密钥的`key`

```
.data:0000000000403020 target          db 1Eh, 5Ch, 13h, 1Fh, 4Bh, 25h, 4Bh, 5Ch, 2Dh, 1Bh, 0
.data:0000000000403020                                         ; DATA XREF: main+ED↑o
.data:000000000040302B                 db 5, 1Dh, 6Fh, 6, 17h, 6Fh, 0, 1Dh, 46h, 41h, 0Ah, 3
.data:0000000000403037                 db 17h, 5
```

4. 随后运用python写出脚本

```python
def xor_encrypt(lst1, lst2):
cipher = [int(c, 16) for c in lst1]
key = [int(k, 16) for k in lst2]

result = []
key_len = len(lst2)
for i in range(len(lst1)):
    encrypted = cipher[i] ^ key[i % key_len]
    result.append(encrypted)
    return result

    def main():


    lst1 =
['1E','5C','13','1F','4B','25','4B','5C','2D','1B','00','05','1D','6F','06','17','
6F','00','1D','46','41','0A','03','17','05']

    lst2 = ['78','30','72']

    encrypted = xor_encrypt(lst1, lst2)


    for i, val in enumerate(encrypted):
        hex_str = hex(val)[2:]
    ascii_str = ''.join([chr(val) if 32 <= val <= 126 else '�' for val in
encrypted])
    print("\n完整ASCII字符串: ")
    print(ascii_str)

    if __name__ == "__main__":
    main()
```

```
运行得flag为`flag{W3l_c0we_to_rev3r3e}`,代入exe文件发现运行正确
```