目录

回顾

1. OD 支持哪些调试方式？
   ① 支持拖拽
   ② 支持菜单-文件-打开，打开的时候可以添加命令行参数
   ③ 支持右键菜单打开
   ④ 支持附加调试
   ⑤ 支持 DLL 调试
   ⑥ 支持多线程调试

2. OD 中的断点有哪些？分别原理是什么？
   ① 软件断点，F2，修改代码处为 int3，异常触发再还原
   ② 硬件断点，寄存器设置地址到 dr0~dr3，dr6，触发单步异常
   ③ 内存断点，修改内存属性(VirtualProtectEx)
   ④ 条件断点，Shift+F2，与软件断点一样
   ⑤ 消息断点，在条件断点基础上增加了宏定义
   ⑥ 记录断点，利用跟踪记录，然后设置条件，可以断下
3. OD 快捷键有哪些？

F7 单步步入
F2 断点
F8 单步步过
Ctrl+P 补丁
F9 执行
Ctrl+F2 重新加载
F3 打开文件
Ctrl+G 打开跳转窗口
Alt+B 查看所有断点
Ctrl+F9 执行到返回
Ctrl+T 设置条件
Alt+X 退出
Ctrl+A 分析模块
-/+ 上一步.下一步
Ctrl+S 命令搜索
Ctrl+E 打开编辑窗口

ESC 返回上一步
F4 运行到光标处
Alt+E 打开模块列表
Alt+K 查看调用栈
Ctrl+F7 自动步进
Ctrl+F8 自动步过
Ctrl+ +/- 跳转到上/下一个函数
Alt+C 打开 CPU 窗口
; 注释
Alt+F4 退出 OD
Alt+M 查看内存
Alt+L 查看记录
Ctrl+B 查询二进制数据
空格 打开汇编窗口
Ctrl+L 查询下一条数据
Alt+O 调试设置
Ctrl+F11 跟踪记录

汇编指令

push eax

等价于：
①
sub esp, 4
mov [esp],eax

②
dec esp
dec esp
dec esp
dec esp
mov [esp],eax

VS 函数

1. CheckESP

调用之前 是 cmp xxx,esp 指令，call 内部是 jnz xxx

函数一般是在函数结束时、call 调用完之后出现

2. CheckStackVars

调用之前 是 lea edx,xxxxx 指令，call 内部是对 0Xcccccccc 进行判断

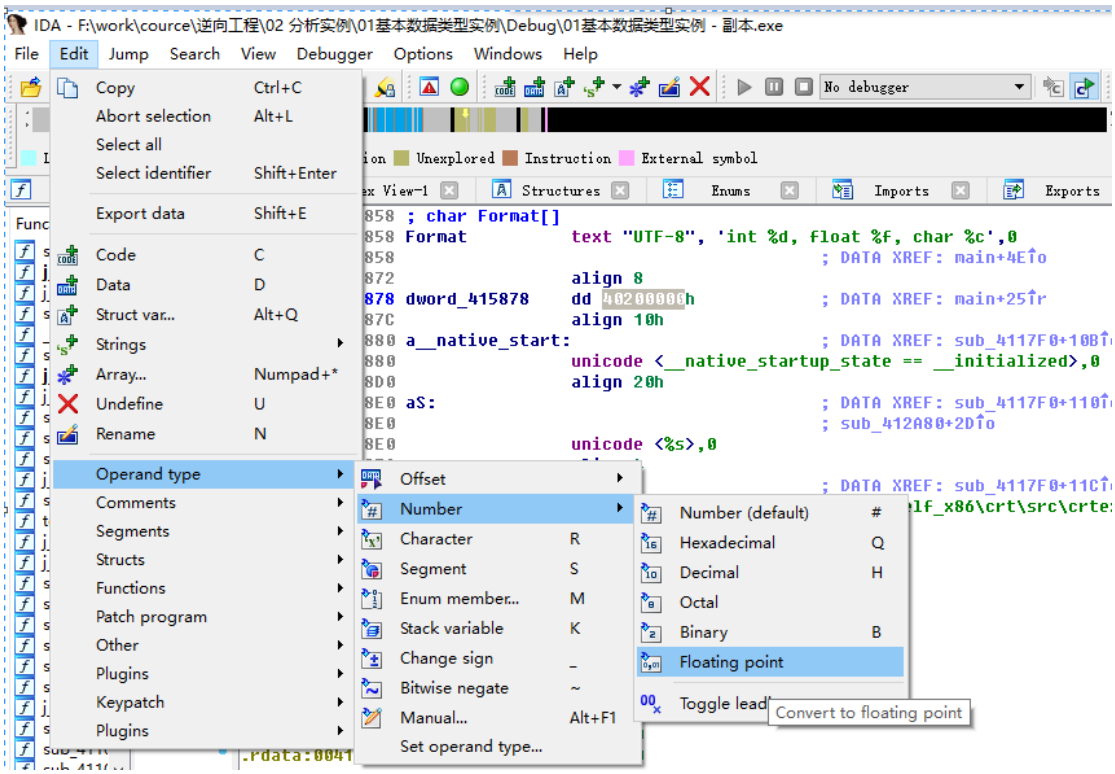
函数一般在函数末尾。

```
t:00411403        xor     eax, eax          ; -----------------
t:00411405        push    edx               ; CheckStackVars前面有两个push,保存寄存器环境
t:00411406        mov     ecx, ebp
t:00411408        push    eax
t:00411409        lea     edx, dword_411420
t:0041140F        call    j_CheckStackVars
t:00411414        pop     eax
t:00411415        pop     edx               <-- 平衡堆栈
t:00411416        pop     edi
```

3. security_check_cookie

调用之前是 xor exx, ebp 指令，call 内部是对参数与安全 cookie 进行判断。

IDA 中转换数据为浮点数



课堂练习-找 2015main 函数

源码到 main 函数流程

```
mainCRTStartup()->
        __scrt_common_main()->
                    __scrt_common_main_seh->
                                        invoke_main()->
                                                main(参数个数,参数数组,环境数组);
```

堆栈信息：