《软件安全》实验报告

姓名：张丛 学号：2113662 班级：信安一班

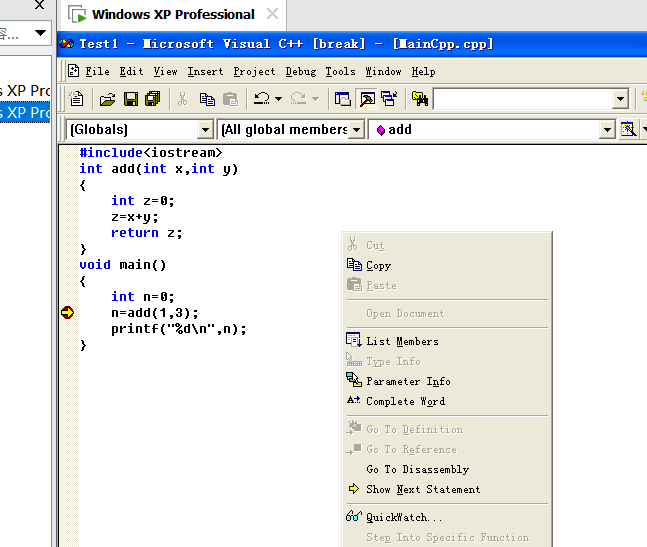
**实验名称：**

IDE反汇编实验

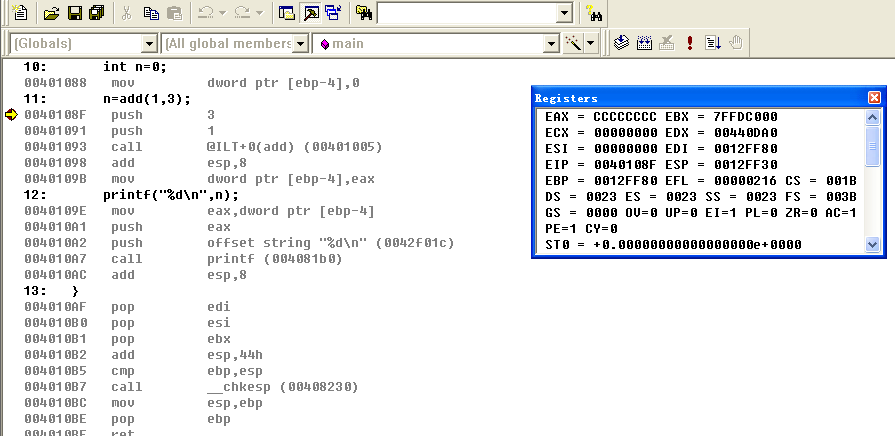
**实验要求：**

根据第二章示例2-1，在XP环境下进行VC6反汇编调试，熟悉函数调用、栈帧切换、CALL和RET指令等汇编语言实现，将call语句执行过程中的EIP变化、ESP、EBP变化等状态进行记录，解释变化的主要原因。

**实验过程：**

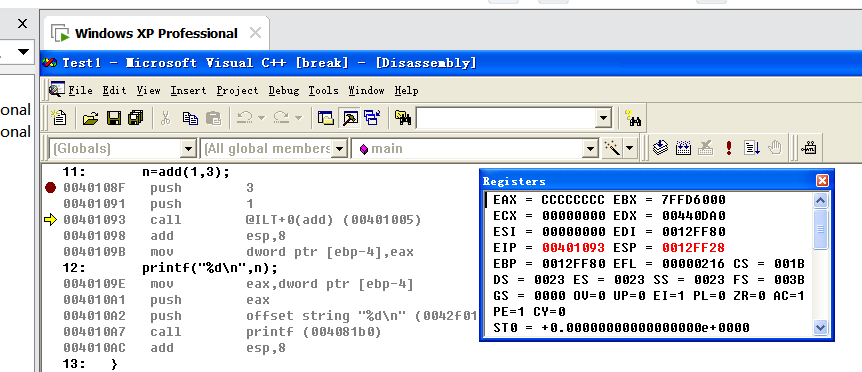


**1. 设置断点，进入调试，右键点击“Go to Disassembly”进入VC反汇编**



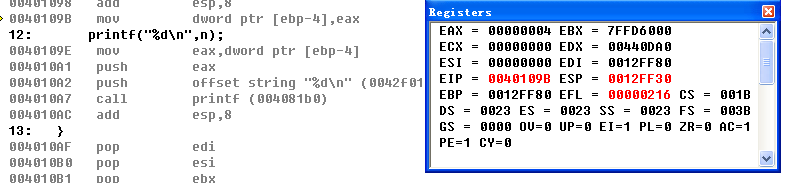
1. **观察add函数调用前后语句**

首先初始化局部变量n,地址为[ebp-4];



进行两次push，是参数从右向左入栈，ESP地址由0012FF30变为0012FF28;

CALL指令调用add函数并返回后：

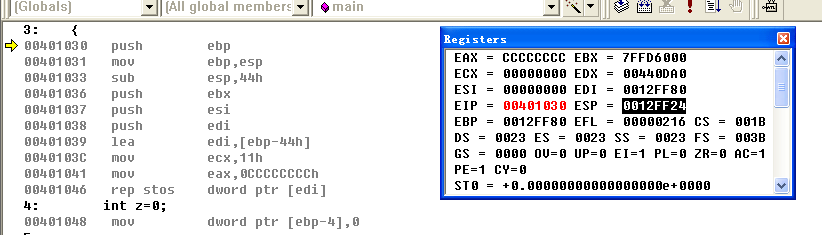


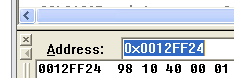
可见，ESP仍为0012FF30;

而后将EAX中的返回值赋给变量n;

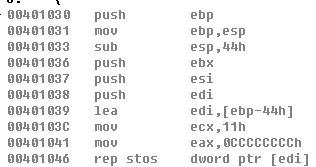
最后调用输出函数输出结果。

**3. add函数内部栈帧切换等关键汇编代码**





调用add函数，ESP变为0012FF24，而此地址存储的正是主函数中call指令下一条指令的地址00401098，即返回地址；



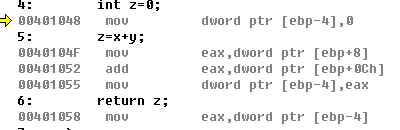
在add函数中，首先将主函数栈帧EBP入栈，此时ESP为0012FF20,对应存储的值为0012FF80;

MOV指令使得，ESP=EBP=0012FF20;

SUB指令使得ESP扩展为0012FEDC，此时便为add函数开辟了栈帧空间；

三次push将主函数可能用到的值入栈，ESP变为0012FED0;

而后11次循环初始化栈帧空间；

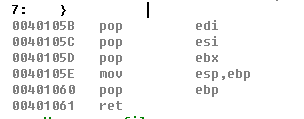


执行add操作：

初始化局部变量z,地址为[ebp-4],值为0；

先将参数累加到EAX中，再将EAX赋给局部变量z;

需要返回时，将z值赋给EAX;

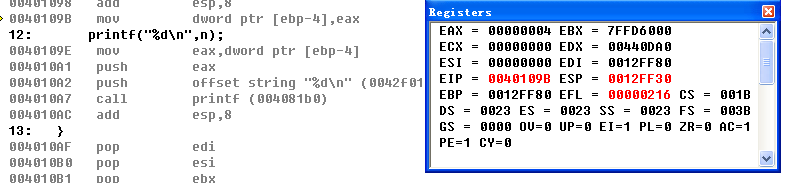


执行完add操作后,将之前的三次push出栈，ESP=0012FEDC;

将EBP赋给ESP后EBP出栈，此时ESP=0012FF24;

RET指令，返回主函数；

返回后，ESP=0012FF30;



**心得体会：**

通过此次实验，掌握虚拟机的安装，在虚拟机上熟悉了函数调用、栈帧切换、CALL和RET指令等汇编语言实现；

参数默认从右向左传递；RET指令实际就是执行了Pop EIP；函数先将EBP入栈，后开辟函数的栈帧空间；返回值一般存储在EAX中；等等。

也明白了汇编语言具有规律性但代码实现却十分繁杂，不同于高级语言，这与直接对话机器有关。

掌握好如参数传递、栈帧切换这些基础的知识，才能有进一步的学习。