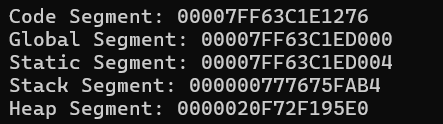
**你的系统程序内存空间分布情况，如代码部分、全局变量部分、堆区、栈区等排列顺序**

使用以下程序验证：



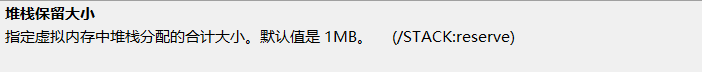
可以看到运行结果：



**你的系统栈空间默认多大？如何修改？**

在 Visual Studio 中，栈的默认大小为 1 MB

在项目属性->配置属性->链接器->系统->堆栈大小，可以看到默认栈大小设置。

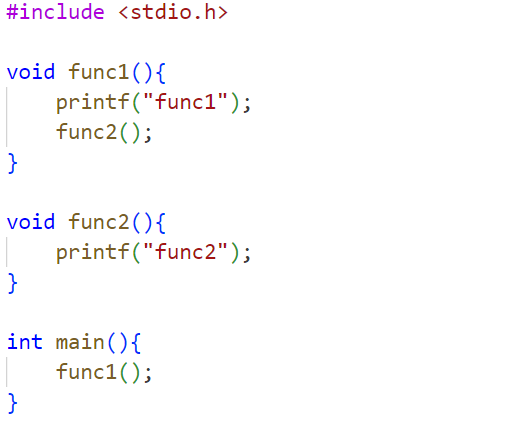


修改“保留大小”和“提交大小”；保留大小是总栈空间（默认 1 MB）；提交大小是初始分配的栈内存大小（默认 4 KB）。

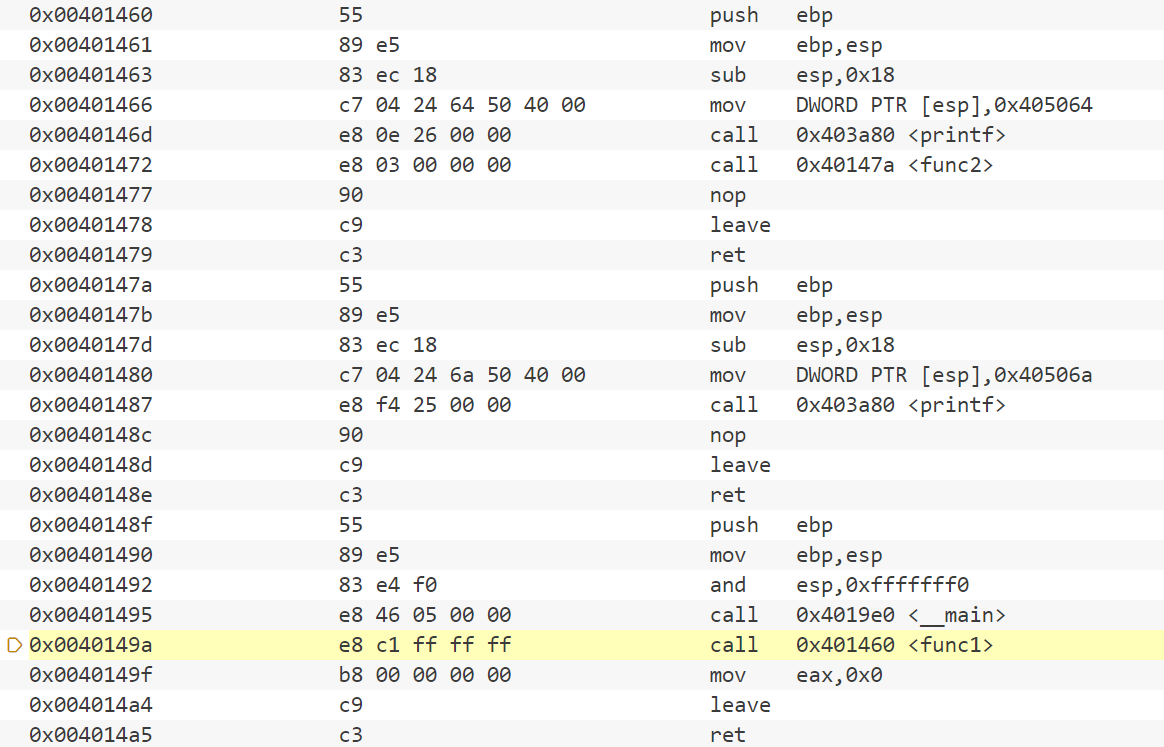
**课堂作业内容**

使用 vs code，gcc编译调试

源程序



反汇编结果



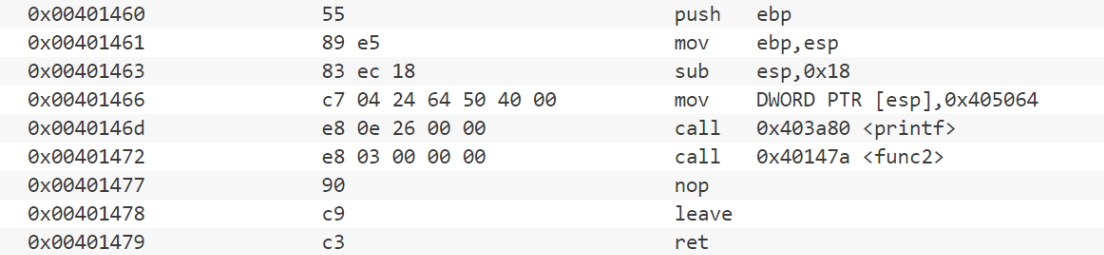
main 函数

这段代码调用了func1

mov eax, 0x0 设置返回值 eax 为0

leave 恢复上一个栈帧

func1 函数



push ebp 保存上一栈帧的基址，为func1 创建新的栈帧

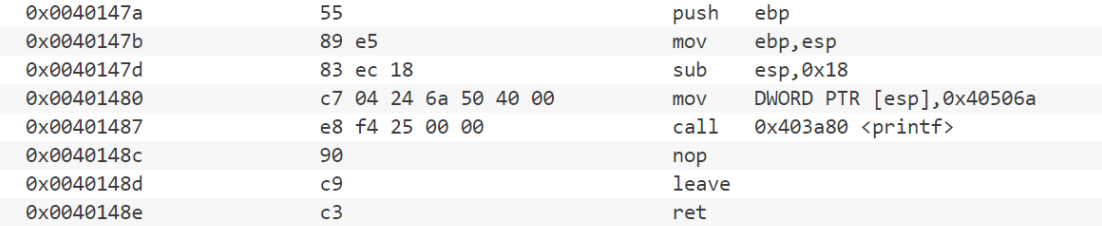
mov ebp, esp 设置当前栈指针 ebp

sub ebp, 0x18 为 func1 函数分配 24 字节的栈空间

mov DWORD PTR [esp], 0x405064: 将 printf 使用的格式化字符串地址存入栈顶（即作为 printf 的参数）

call func2: 将返回地址压入栈顶，并跳转到 func2 函数执行。

func2 函数



push ebp 保存上一栈帧的基址，为func1 创建新的栈帧

mov ebp, esp 设置当前栈指针 ebp

sub ebp, 0x18 为 func2 函数分配 24 字节的栈空间

函数调用时，返回地址和参数压入栈，保证函数间调用的正确性和参数传递。

局部变量分配在栈中，通过 sub esp, size 操作完成。

函数返回时，通过 leave 和 ret 恢复栈帧和控制权。