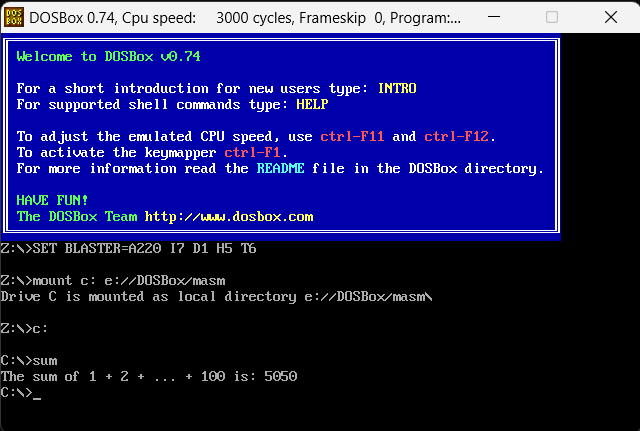
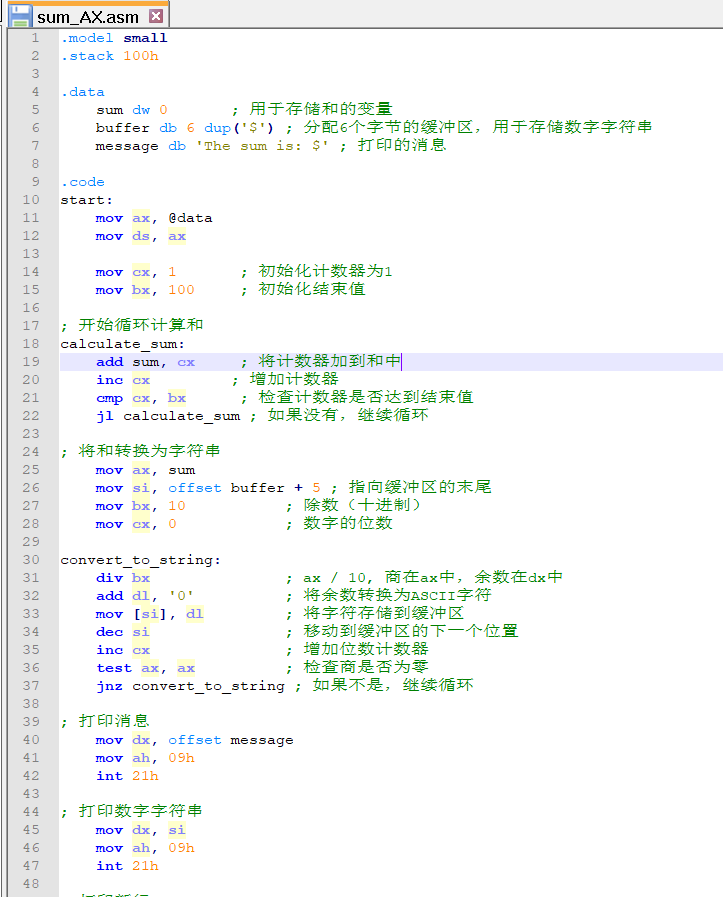
**1.运行结果：**



**将结果存放在寄存器中：**  
**将结果存放在栈中：**

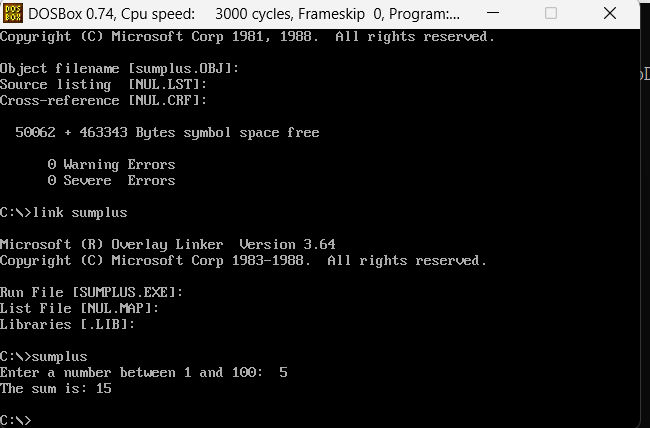


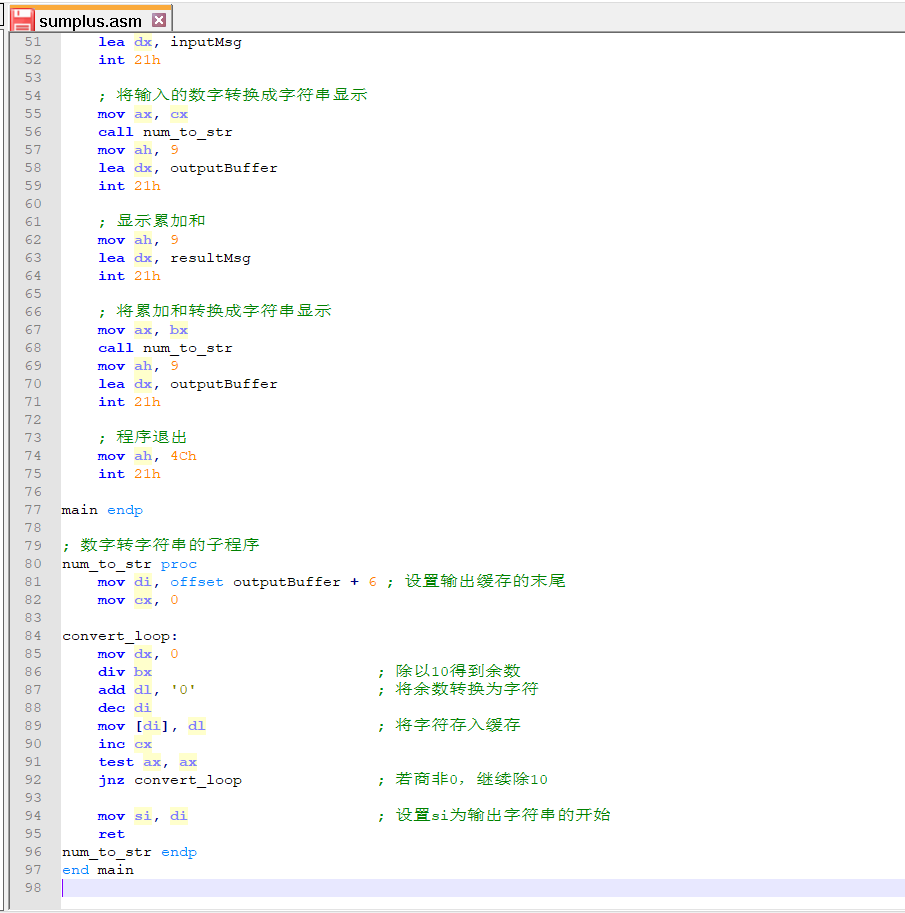
**将结果存放在数据段中：**



**2.**

**运行结果：**

****

**代码：  
**

**3.c语言的反汇编以及注释**

1: #include <stdio.h>

2:

3: int main() {

; 函数入口点

00F71870 55 push ebp

00F71871 8B EC mov ebp,esp

00F71873 81 EC D8 00 00 00 sub esp,0D8h ; 分配局部变量空间

00F71879 53 push ebx

00F7187A 56 push esi

00F7187B 57 push edi

00F7187C 8D 7D E8 lea edi,[ebp-18h] ; 初始化EBP帧指针

00F7187F B9 06 00 00 00 mov ecx,6

00F71884 B8 CC CC CC CC mov eax,0CCCCCCCCh

00F71889 F3 AB rep stos dword ptr es:[edi] ; 将栈初始化为0xCCCCCCCC

00F7188B B9 08 C0 F7 00 mov ecx,offset \_C128DA88\_assembler@c (0F7C008h)

00F71890 E8 95 FA FF FF call @\_\_CheckForDebuggerJustMyCode@4 (0F7132Ah)

00F71895 90 nop

4: int sum = 0;

00F71896 C7 45 F8 00 00 00 00 mov dword ptr [sum],0 ; 初始化sum为0

5: int i;

6:

7: // 计算1到100的和

8: for (i = 1; i <= 100; i++) {

00F7189D C7 45 EC 01 00 00 00 mov dword ptr [i],1 ; 初始化i为1

00F718A4 EB 09 jmp \_\_$EncStackInitStart+33h (0F718AFh) ; 跳转到循环体

00F718A6 8B 45 EC mov eax,dword ptr [i] ; 加载i的值

00F718A9 83 C0 01 add eax,1 ; i自增1

00F718AC 89 45 EC mov dword ptr [i],eax ; 存储i的新值

00F718AF 83 7D EC 64 cmp dword ptr [i],64h ; 检查i是否大于100

00F718B3 7F 0B jg \_\_$EncStackInitStart+44h (0F718C0h) ; 如果i大于100，跳转到循环外

9: sum += i;

00F718B5 8B 45 F8 mov eax,dword ptr [sum] ; 加载sum的值

00F718B8 03 45 EC add eax,dword ptr [i] ; sum加上i的值

00F718BB 89 45 F8 mov dword ptr [sum],eax ; 存储sum的新值

10: }

00F718BE EB E6 jmp \_\_$EncStackInitStart+2Ah (0F718A6h) ; 跳转回循环开始

11:

12: // 打印结果

13: printf("The sum of numbers from 1 to 100 is: %d\n", sum);

00F718C0 8B 45 F8 mov eax,dword ptr [sum] ; 加载sum的值

00F718C3 50 push eax ; 将sum的值压入栈

00F718C4 68 30 7D F7 00 push offset string "The sum of numbers from 1 to 10@"... (0F77D30h) ; 打印字符串

00F718C9 E8 04 F8 FF FF call \_printf (0F710D2h) ; 调用printf函数

00F718CE 83 C4 08 add esp,8 ; 清理栈空间

14:

15: return 0;

00F718D1 33 C0 xor eax,eax ; 设置返回值为0

16:

17: }

00F718D3 5F pop edi

00F718D4 5E pop esi

00F718D5 5B pop ebx

00F718D6 81 C4 D8 00 00 00 add esp,0D8h ; 清理栈空间

00F718DC 3B EC cmp ebp,esp

00F718DE E8 6B F9 FF FF call \_\_RTC\_CheckEsp (0F7124Eh) ; 检查栈溢出

00F718E3 8B E5 mov esp,ebp

00F718E5 5D pop ebp

00F718E6 C3 ret ; 返回到调用者