--- E:\Desktop\大三上课程文件\汇编语言\Homework01\hw3_C语言代码\hw3_C语言代码\hw3_C语言代码.c ---------
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main() {
; 函数序言 - 建立栈帧
00007FF7F8131960 push   rbp   ；保存旧的基址指针
00007FF7F8131962 push   rdi   ；保存rdi寄存器
00007FF7F8131963 sub   rsp,168h ；在栈上分配168h(360)字节空间
00007FF7F813196A lea   rbp,[rsp+20h] ；设置新的基址指针(rbp = rsp + 20h)
00007FF7F813196F lea   rdi,[rsp+20h] ；rdi指向栈帧开始
00007FF7F8131974 mov   ecx,22h   ；设置循环计数(34次 = 168h/4)
00007FF7F8131979 mov   eax,0CCCCCCCCh ；填充值(未初始化内存标记)
00007FF7F813197E rep stos dword ptr [rdi] ；用0xCC填充栈空间(调试版本的安全检查)
00007FF7F8131980 mov   rax,qword ptr [**security_cookie (07FF7F813D000h)] ；加载安全cookie**
00007FF7F8131987 xor   rax,rbp   ；异或操作(栈保护)
00007FF7F813198A mov   qword ptr [rbp+138h],rax ；保存保护值到栈中
00007FF7F8131991 lea   rcx,[E8BEC052_hw3_C语言代码@c (07FF7F8142008h)] ；加载调试信息
00007FF7F8131998 call   __CheckForDebuggerJustMyCode (07FF7F813137Ah) ；调试检查调用

```
// 1：直接计算
int sum = 0;
```

00007FF7F813199D mov   dword ptr [sum],0 ；初始化sum = 0

```
for (int i = 1; i <= 100; i++) {
```

00007FF7F81319A4 mov   dword ptr [rbp+24h],1 ；i = 1 (i存储在[rbp+24h])
00007FF7F81319AB jmp   __$EncStackInitStart+46h (07FF7F81319B5h) ；跳转到循环条件检查

; 循环体开始
00007FF7F81319AD mov   eax,dword ptr [rbp+24h] ；加载i到eax
00007FF7F81319B0 inc   eax   ; i++ (eax = i + 1)
00007FF7F81319B2 mov   dword ptr [rbp+24h],eax ；保存i回内存

; 循环条件检查
00007FF7F81319B5 cmp   dword ptr [rbp+24h],64h ；比较i和100(0x64)
00007FF7F81319B9 jg   __$EncStackInitStart+5Bh (07FF7F81319CAh) ；如果i>100，跳出循环

```
    sum += i;
```

00007FF7F81319BB mov   eax,dword ptr [rbp+24h] ；加载i到eax
00007FF7F81319BE mov   ecx,dword ptr [sum]   ；加载sum到ecx
00007FF7F81319C1 add   ecx,eax   ; ecx = sum + i
00007FF7F81319C3 mov   eax,ecx   ；结果移到eax
00007FF7F81319C5 mov   dword ptr [sum],eax   ；保存结果回sum

```
}
```

00007FF7F81319C8 jmp   __$EncStackInitStart+3Eh (07FF7F81319ADh) ；跳回循环开始

; 循环结束
    printf("1+2+...+100 = %d\n", sum);
00007FF7F81319CA mov   edx,dword ptr [sum]   ；第二个参数: sum值
00007FF7F81319CD lea   rcx,[string "1+2+...+100 = %d\n" (07FF7F813AD28h)] ；第一个参数: 格式字符串地址
00007FF7F81319D4 call   printf (07FF7F813119Fh) ；调用printf函数

```
// 2：用户输入版本
int n;
printf("Please enter a number (1-100): ");
```

00007FF7F81319D9 lea   rcx,[string "Please enter a number (1-100): " (07FF7F813AD40h)] ；提示字符串
00007FF7F81319E0 call   printf (07FF7F813119Fh) ；调用printf

```
scanf("%d", &n);
```

00007FF7F81319E5 lea   rdx,[n]   ；第二个参数: n的地址(&n)
00007FF7F81319E9 lea   rcx,[string "%d" (07FF7F813AD68h)] ；第一个参数: 格式字符串
00007FF7F81319F0 call   scanf (07FF7F813109Bh) ；调用scanf函数

```
sum = 0;
```

00007FF7F81319F5 mov   dword ptr [sum],0   ；重新初始化sum = 0

```
for (int i = 1; i <= n; i++) {
```

00007FF7F81319FC mov   dword ptr [rbp+64h],1   ; i = 1 (存储在[rbp+64h])
00007FF7F8131A03 jmp   __$EncStackInitStart+9Eh (07FF7F8131A0Dh) ；跳转到条件检查

; 循环体开始
00007FF7F8131A05 mov   eax,dword ptr [rbp+64h] ；加载i到eax
00007FF7F8131A08 inc   eax   ; i++
00007FF7F8131A0A mov   dword ptr [rbp+64h],eax ；保存i回内存

; 循环条件检查
00007FF7F8131A0D mov   eax,dword ptr [n]   ；加载n到eax
00007FF7F8131A10 cmp   dword ptr [rbp+64h],eax ；比较i和n
00007FF7F8131A13 jg   __$EncStackInitStart+0B5h (07FF7F8131A24h) ；如果i>n，跳出循环

```
    sum += i;
```

```
00007FF7F8131A15  mov         eax,dword ptr [rbp+64h] ；加载i到eax
00007FF7F8131A18  mov         ecx,dword ptr [sum]      ；加载sum到ecx
00007FF7F8131A1B  add         ecx,eax       ; ecx = sum + i
00007FF7F8131A1D  mov         eax,ecx       ；结果移到eax
00007FF7F8131A1F  mov         dword ptr [sum],eax      ；保存结果回sum
```

```
}
```

```
00007FF7F8131A22  jmp         __$EncStackInitStart+96h (07FF7F8131A05h) ；跳回循环开始

; 循环结束
    printf("1+2+...+%d = %d\n", n, sum);
00007FF7F8131A24  mov         r8d,dword ptr [sum]      ；第三个参数: sum值
00007FF7F8131A28  mov         edx,dword ptr [n]        ；第二个参数: n值
00007FF7F8131A2B  lea         rcx,[string "1+2+...+%d = %d\n" (07FF7F813AD70h)] ；第一个参数: 格式字符串
00007FF7F8131A32  call        printf (07FF7F813119Fh) ；调用printf
```

```
return 0;
```

```
00007FF7F8131A37  xor         eax,eax      ；返回值清零(eax = 0)

; 函数尾声 - 清理栈帧
}
00007FF7F8131A39  mov         edi,eax      ；保存返回值到edi
00007FF7F8131A3B  lea         rcx,[rbp-20h] ；第一个参数: 局部变量区域
00007FF7F8131A3F  lea         rdx,[xt_z+260h (07FF7F813AD00h)] ；第二个参数: 检查数据
00007FF7F8131A46  call        _RTC_CheckStackVars (07FF7F8131316h) ；运行时栈变量检查
00007FF7F8131A4B  mov         eax,edi      ；恢复返回值到eax
00007FF7F8131A4D  mov         rcx,qword ptr [rbp+138h] ；加载保存的保护值
00007FF7F8131A54  xor         rcx,rbp      ；验证栈保护cookie
00007FF7F8131A57  call        security_check_cookie (07FF7F81311B8h) ；安全检查调用
00007FF7F8131A5C  lea         rsp,[rbp+148h] ；恢复栈指针
00007FF7F8131A63  pop         rdi          ；恢复rdi寄存器
00007FF7F8131A64  pop         rbp          ；恢复基址指针
00007FF7F8131A65  ret                      ；函数返回
```