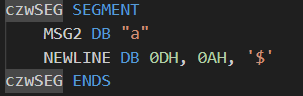
输出ASCII表中的小写字母部分，每13个一行

1. 使用LOOP指令实现

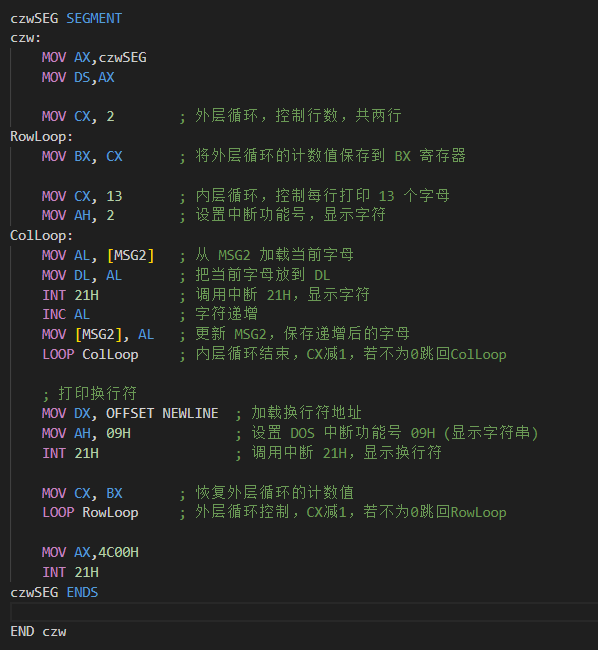
数据段：初始化字母’a’，并定义换行符。



假设段：当前代码段使用 czwSEG 段。



代码段：

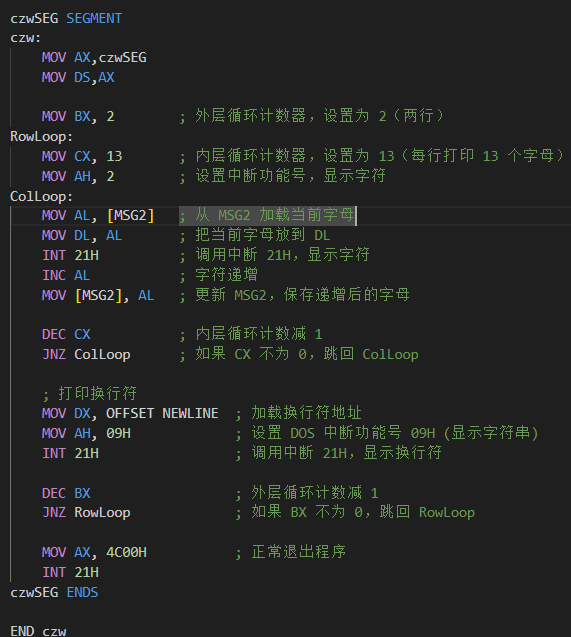


程序通过双重循环实现了字符的逐行输出，外层循环控制输出行数，内层循环控制每行输出的字母数量。

1. 使用跳转指令实现

数据段和假设段与方法一相同，代码段有所区别。

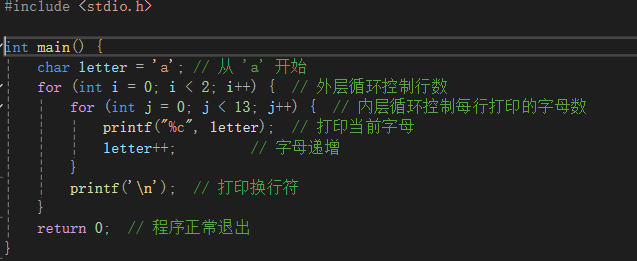
代码段：



在此程序中，使用DEC和JNZ来实现循环控制。DEC减去1并设置标志位，JNZ根据标志位判断是否跳回循环。这种方法更灵活，可以根据需要实现更复杂的循环条件（例如，基于其他寄存器或变量的值）。

1. 用C语言实现并反汇编

C语言代码：



反汇编结果：

00007FF6A3E03BA0 push rbp

00007FF6A3E03BA2 push rdi

00007FF6A3E03BA3 sub rsp,148h

00007FF6A3E03BAA lea rbp,[rsp+20h]

00007FF6A3E03BAF lea rcx,[\_\_843D2DC9\_第二次作业\print\_letters@c (07FF6A3E11008h)]

00007FF6A3E03BB6 call \_\_CheckForDebuggerJustMyCode (07FF6A3E01357h)

00007FF6A3E03BBB nop

**保存当前基指针到栈中，并为局部变量分配空间。**

00007FF6A3E03BBC mov byte ptr [letter],61h

**初始化变量letter。**

00007FF6A3E03BC0 mov dword ptr [rbp+24h],0

00007FF6A3E03BC7 jmp main+31h (07FF6A3E03BD1h)

00007FF6A3E03BC9 mov eax,dword ptr [rbp+24h]

00007FF6A3E03BCC inc eax

00007FF6A3E03BCE mov dword ptr [rbp+24h],eax

00007FF6A3E03BD1 cmp dword ptr [rbp+24h],2

00007FF6A3E03BD5 jge main+79h (07FF6A3E03C19h)

**初始化 i 为 0，并通过条件跳转控制外层循环，判断 i 是否小于 2。**

00007FF6A3E03BD7 mov dword ptr [rbp+44h],0

00007FF6A3E03BDE jmp main+48h (07FF6A3E03BE8h)

00007FF6A3E03BE0 mov eax,dword ptr [rbp+44h]

00007FF6A3E03BE3 inc eax

00007FF6A3E03BE5 mov dword ptr [rbp+44h],eax

00007FF6A3E03BE8 cmp dword ptr [rbp+44h],0Dh

00007FF6A3E03BEC jge main+6Ch (07FF6A3E03C0Ch)

**初始化 j 为 0，并通过条件跳转控制内层循环，判断 j 是否小于 13。**

00007FF6A3E03BEE movsx eax,byte ptr [letter]

00007FF6A3E03BF2 mov edx,eax

00007FF6A3E03BF4 lea rcx,[string "%c" (07FF6A3E09BD8h)]

00007FF6A3E03BFB call printf (07FF6A3E013BBh)

00007FF6A3E03C00 nop

**将 letter 的值传递给 printf 函数进行打印。**

00007FF6A3E03C01 movzx eax,byte ptr [letter]

00007FF6A3E03C05 inc al

00007FF6A3E03C07 mov byte ptr [letter],al

}

00007FF6A3E03C0A jmp main+40h (07FF6A3E03BE0h)

**将 letter 的值递增。**

00007FF6A3E03C0C mov ecx,0Ah

00007FF6A3E03C11 call printf (07FF6A3E013BBh)

00007FF6A3E03C16 nop

}

00007FF6A3E03C17 jmp main+29h (07FF6A3E03BC9h)

**打印换行符。**

00007FF6A3E03C19 xor eax,eax

**将 0 返回给操作系统，表示正常退出。**

00007FF6A3E03C1B lea rsp,[rbp+128h]

00007FF6A3E03C22 pop rdi

00007FF6A3E03C23 pop rbp

00007FF6A3E03C24 ret

**恢复栈指针和基指针，结束函数。**

C语言反汇编生成的代码，首先需要保存基指针，为局部变量分配空间，然后初始化变量，再通过条件跳转指令实现循环打印字母表。