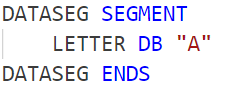
**输出 ASCII 表中的小写字母部分，每行 13 个字符**

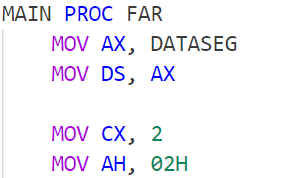
该代码通过双重循环，分别实现打印两行，每行 13 个字母的功能。外层循环控制行数，内层循环控制每行字符的数量。分别使用 loop 指令和条件跳转指令实现。

数据段：



存储第一个打印的字母，即 'A'。

初始化：



换行：



每行打印完后，通过向 DL 寄存器分别存入 0DH (回车) 和 0AH (换行) 来实现换行操作。

1. 用 loop 指令实现

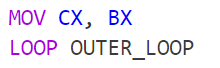
在汇编程序中，LOOP 用于控制循环。当 CPU 执行 LOOP 时，先将 CX 自动减 1，然后判断 CX 是否为 0，如果 CX 为 0 ，则退出循环，执行 LOOP 语句的后一句语句；否则跳转到指定的标签继续执行循环。

对于本程序，在外层和内层循环中，都使用了 CX 寄存器来决定循环的次数。如果不保存 CX 寄存器的值，则内层循环会覆盖外层循环的计数器，导致外层循环无法正确工作。因此，在多层循环中，我们需要注意保存和恢复 CX 寄存器，避免数据被覆盖。本程序中使用 BX 寄存器来保存 CX 寄存器的值。

在外层循环还没有进入内存循环时，保存 CX 寄存器的值到 BX 寄存器

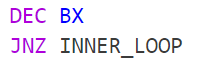


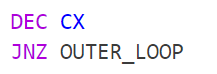
内层循环结束后，在外层循环中恢复 CX 寄存器的值



1. 用条件跳转指令实现

条件跳转指令在满足特定条件时执行跳转。本程序使用 JNZ 指令，该指令的意义为，如果不为零，则跳转。JNZ 指令会检查上一个运算之后的结果，如果结果不是零，JNZ 就会跳转到指定的标签。



这两条指令会自减存放循环次数的寄存器，并判断该值在自减之后是不是零。如果该值为零，则退出循环，否则跳转回 INNER\_LOOP 标签处。  
 

外层循环同理。