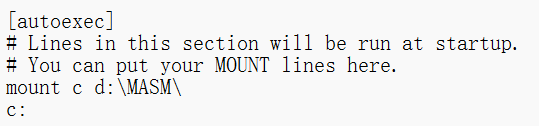
Assignment 1 Hello World

1. **环境配置**

1.1 安装DOSBox，在DOSBox的安装目录下打开DOSBox 0.74 Options.bat Windows 批处理文件，在打开的 dosbox-0.74.conf 配置文件的末尾添加如下内容，并保存。这些配置使得 DOSBox 启动后，直接挂载并进入。

1.2 在对应文件夹里添加相应配置程序

1. **传统编译方式**
   1. 在文件夹中创建 HELLO.ASM，汇编文件内容如下：

.MODEL SMALL                                ; 定义内存模型为small模式

.STACK 100h                                 ; 定义堆栈大小为256字节

.DATA

    Hello DB 'Hello world!', 0dh, 0ah, '$'  ; 定义要输出的字符串 'Hello world!'

                                            ; 0Dh = 回车符 (CR)

                                            ; 0Ah = 换行符 (LF)

                                            ; '$' 表示字符串的结束符，用于DOS的INT 21h功能9

.CODE

START:

    ; 初始化数据段寄存器

    MOV AX, @DATA                           ; 将数据段的基地址存入AX

    MOV DS, AX                              ; 将AX中的数据段地址加载到DS寄存器

    ; 调用DOS中断输出字符串

    MOV DX, offset Hello                    ; 将Hello字符串的地址存入DX

    MOV AH, 9                               ; DOS功能号9：输出字符串，字符串必须以'$'结束

    INT 21H                                 ; 触发中断21h，执行字符串输出

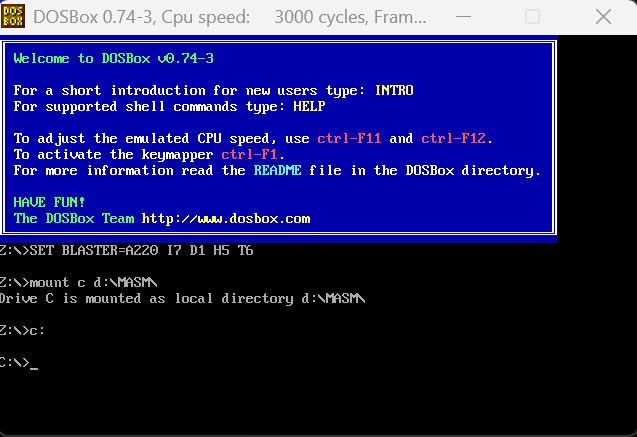
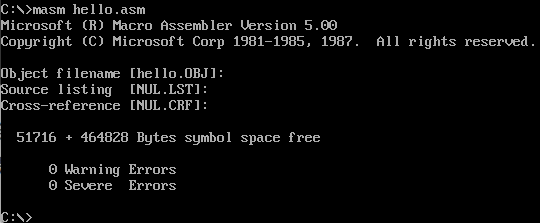
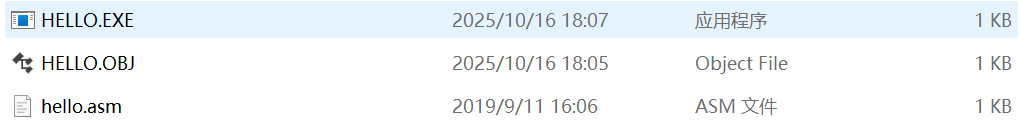
    ; 正常结束程序

    MOV AX, 4C00H                           ; 设置返回代码为0的结束程序指令

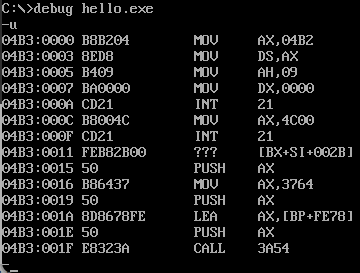
    INT 21h                                 ; 触发中断21h，返回到操作系统

END START                                   ; 标记程序结束，START是入口点

* 1. 编译并运行汇编程序

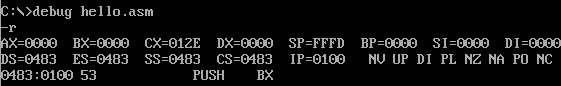
1. 启动 DOSBox
2. 执行命令masm hello.asm，Object filename 选项回车，表示接受默认文件名，Source listing 选项回车，表示不生成源代码列表文件，Cross-reference 回车，表示不生成交叉引用表。这一步的作用是使用编译器将汇编语言的 .asm 源文件编译成一个 .obj 文件，用于后续的链接步骤
3. 执行命令 link hello.obj，Run File 选项回车，表示接受默认文件名，List File 选项回车，表示不生成列表文件，Libraries 选项回车，表示不链接任何额外的库文件，采用默认设置。这一步通过链接器将编译生成的目标文件 hello.obj 链接为可执行文件。
4. 执行命令hello，执行可执行程序。



1. 执行命令 debug hello.exe，再使用 -u 命令，可以反汇编可执行文件的机器代码，逐条显示汇编指令。
2. 反汇编结果分析

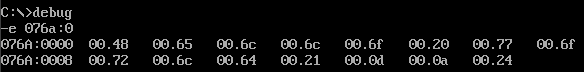
* 04B3:0000 MOV AX,04B2 将数值 04B2 放入 AX 寄存器。这个值 04B2 是程序的数据段（Data Segment）所在的内存段地址
* 04B3:0003 MOV DS,AX 将 AX 寄存器中的值（也就是 04B2）赋给 DS（数据段寄存器）
* 04B3:0005 MOV AH,09 将数值 09 放入 AH 寄存器。这是在为调用DOS中断做准备。AH=9 是DOS 21H 号中断的一个功能号，代表“显示字符串”
* 04B3:0007 BA0000 MOV DX,0000 将数值 0000 放入 DX 寄存器。DX 寄存器需要存放待显示字符串在数据段中的**偏移地址**（Offset Address）。这里的 0000 意味着字符串是从数据段的起始位置开始的，此时DS:DX 完整地指向了字符串的内存地址，即 04B2:0000
* 04B3:000A CD21 INT 21 执行 INT 21H 中断调用。因为此时 AH 的值是 09，所以CPU会去执行显示字符串的操作。它会找到 DS:DX 指向的地址，并将该地址开始的字符串内容显示在屏幕上，直到遇到 $ 符号为止
* 04B3:000C B8004C MOV AX,4C00 将4C00这个十六进制数放入AX寄存器
* 04B3:000F CD21 INT 21 再次执行 INT 21H 中断

1. **另类编译方式，直接将代码和数据使用debug -e写入内存执行**

3.1 启动 DOSBox，执行命令debug hello.asm，再使用 -r 命令，查看并显示 CPU 寄存器的当前状态。

3.2 写入数据：使用e (enter/edit) 命令，将字符串 "Hello world!" 及其结束符的 ASCII 码写入内存。我们选一个段地址076A，从偏移地址0开始写入。

48 65 6c 6c 6f 20 77 6f 72 6c 64 21 0d 0a 24

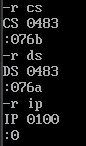
* 48 65 6c 6c 6f 是 "Hello"
* 20 是空格
* 77 6f 72 6c 64 21 是 "world!"
* 0d 0a 是回车换行
* 24 是字符串结束符 $

3.3 写入代码：使用e命令，将程序的机器码写入另一个内存地址。我们选择段地址 076B，从偏移地址0开始写入。

b8 6a 07 8e d8 ba 00 00 b4 09 cd 21 b8 00 4c cd 21

这段机器码对应的汇编指令是：

* b8 6a 07 -> MOV AX, 076A (将我们的数据段地址送入 AX)
* 8e d8 -> MOV DS, AX (设置数据段寄存器)
* ba 00 00 -> MOV DX, 0000 (设置字符串的偏移地址)
* b4 09 -> MOV AH, 09 (调用 DOS 9号功能：显示字符串)
* cd 21 -> INT 21H (执行 DOS 中断)
* b8 00 4c -> MOV AX, 4C00H (调用 DOS 4C号功能：带返回值退出)
* cd 21 -> INT 21H (执行 DOS 中断)

3.4 修改寄存器：使用r (register) 命令，告诉CPU代码段(CS)、数据段(DS)和指令指针(IP)在哪里。

3.5 执行程序：使用 g (go) 命令来执行我们刚刚写入内存的代码