实践课项目二

一、按下列步骤进行上机实践，并回答问题。

1.按如下步骤搭建汇编开发环境

第一步： 从公共邮箱文件中心下载汇编编译器 masm 和连接器 link。

第二步： 在 c 盘下创建 masm 文件夹。

第三步： 将下载的 masm 和 link 文件复制到 masm 文件夹中。

第四步： 运行 command，键入 【cd\】 回车，键入 【cd masm】 回车。

第五步： 同时按下【alt】和【enter】 键，进入全屏模式。

第六步： 键入 dir 命令，查看 masm 文件下的 masm.exe 和 link.exe 文件。

第七步： 同时按下【alt】和【enter】 键，退出全屏模式。

(**注：**若计算机操作系统是 64 位操作系统，则先下载本邮箱文件中心中的 Dosbox+Debug 工具并安装到电脑上（注意：内附使用说明），然后将文件中心的 masm5.zip 解压，将解压后的文件夹重命名为 masm，拷贝到 Dosbox 安装目录。然后进行实验。没有 edit.exe 的，可以使用 windows 系统自带的编辑器：记事本（notepad）。)

2.编辑、编译、连接汇编程序

第一步：运行 command。 同时按下【alt】和【enter】 键，进入全屏模式。

第二步：键入 【cd\】 回车，键入【cd masm】 回车，切换到我们的工作目录。

第三步：键入【edit simple.asm】 回车, 建立 simple.asm 文件。

第四步：在 edit 中编写下面汇编程序。（编写过程中可用上下左右键调整光标位置。）

1）写程序框架：

```
    assume cs:codesg
codesg segment

codesg ends
end
```

2）在框架中填写如下功能代码：

```
    mov ax,1
add ax,ax
add ax,ax
add ax,ax
add ax,ax
```

3）继续添加返回语句

```
mov ax,4c00h
int 21h
```

4）完成上面两步后，edit 中完成的程序如下：

```
        assume cs:codesg
codesg segment
         mov ax,1
         add ax,ax
         add ax,ax
         add ax,ax
               add ax,ax

        mov ax,4c00h
        int 21h
        codesg ends
end
```

第四步：同时按下【alt】和【f】键，弹出 edit 的 file 菜单，也可以鼠标点击。

第五步：按下【s】键，保存我们所编写的程序，也可以鼠标点击。

第六步：同时按下【alt】和【f】键，弹出 edit 的 file 菜单，也可以鼠标点击。

第七步：按下【x】键，退出 edit 返回到 command，也可以鼠标点击。

第八步：键入【dir】，查看我们刚刚编写的 simple.asm 源文件，确认文件是否存在。

第九步：键入【masm simple;】 编译我们所写的源程序。若编译有错误则跳到第三步，检查自己的错误。

第十步：键入【link simple;】 连接我们所生成的目标文件，最终生成可执行文件。

第十一步：键入【dir】命令查看我们所生成的 simple.obj 和 simple.exe 文件。

simple.obj 和 simple.exe 这两个文件的大小是 71 ，528 。

3.调试程序

第一步：键入【debug simple.exe】命令，用 debug 加载我们的程序。

第一步：使用 R 命令观察初始寄存器的值。

AX= ffff ；BX= 0000 ；CX= 0010 ；DX= 0000 ；CS= 076A ；DS= 075A ；SS= 0769 ；ES= 075A ；SP= 0000 ；IP= 0000 ；

第一步：整个代码的长度是 10 字节。这是通过寄存器 cx 得到的。

第二步： 键入【u】回车，在 debug 下查看我们所写的程序。

第四步：键入【t】回车，单步执行一条程序。 查看相应寄存器的值的变化。

第五步：用 debug 的【t】命令执行完 int 21h 前的所有指令。在单步运行过程中，最后一条 add ax，ax

执行后，ax 值是 0010 。
第六步：执行到 int 21h 时，键入【p】回车。执行 int 21h 这条“特殊”指令。
第七步：键入【q】回车，终止 debug 运行，返回到 command。

二、编写程序，复制 PSP 的内容到安全空间中。

1.内存中的一段安全空间是：0:200-:0:2ff
2.在 edit 中书写下面的源程序。

```
assume cs:code
code segment

    mov ax,20h
    mov es,ax
    mov bx,0

    mov cx,256
s:  mov ah,ds:[bx]
    mov es:[bx],ah
    inc bx
    loop s

    mov ax,4c00h
    int 21h

code ends
end
```

3.编译连接生成.exe 文件。
4.用 debug 加载程序，查看 PSP 中的前 16 个字节用到的命令是： d 075a：0
将这 16 个字节的内容抄写如下：
CD 20 FF 9F 00 EA FF FF-AD AE 4F 03 A3 01 8A 03

5.程序执行后（注意：查看结果一定要在程序返回前查看，即执行 mov ax,4c00h int 21h 这两条指令前查看结果。），查看安全空间的前 16 个字节用到的命令是： d 0020:0
将这 16 个字节的内容抄写如下：
CD 20 FF 9F 00 EA FF FF-AD AE 4F 03 A3 01 8A 03
6.这个程序可以进行一些修改，使得程序运行速度更快，请写出修改后的程序：

mov es:[bx],ds:[bx]

三、编程实现如下功能：

把 FFFF0H 开始的 8 个字节单元按字节扩大到原来的 2 倍，然后将扩大后的数值存放到 00200H 开始的 8 个字单元中。

1.将调试成功的源程序写到下方空白处：

```
 assume cs:test3
test3 segment
mov ax,0ffffh
mov ds,ax
mov bx,0
mov ax,0020h
mov es,ax

  mov cx,8
s:  mov  al,ds:[bx]
    mov ah:0
    add ax, ax
    mov es:[bx],al
    inc bx
    loop s
   mov ax,4c00h
   int 21h
test3 ends
end
```

2.用 d 命令查看 FFFF0H 开始的 8 个字节单元的内容是：

3.程序执行前 200H 开始的 16 个字节单元内容是：

4.程序执行后 200H 开始的 16 个字节单元内容是：

四、编写程序完成如下功能：

1.统计 0：0～0：100h 这 256 个字节中， 10h 出现的个数，将结果存入 ax。

将调试成功的源程序写到下方空白处：

2. 读取内存空间 0：0～0：10h 中的 16 个字节的数据，将奇地址单元（字节单元）和偶地址单元（字节单元）的数据对换之后，存入安全地址空间。
将调试成功的源程序写到下方空白处：

3. 判断 ax 中的数，如果是 0，给 bx 赋 1，否则给 bx 赋 2。
将调试成功的源程序写到下方空白处：