

《汇编语言程序设计》

实验 指导 书

计算机学院

目 录

第 1 部分 实验预备知识

1.1 汇编语言程序的上机操作过程

1.2 常用 DEBUG 命令

第 2 部分 汇编语言实验

实验 1	DEBUG 命令.....	8
实验 2	汇编语言上机过程.....	9
实验 3	顺序结构程序设计.....	10
实验 4	分支程序设计.....	11
实验 5	循环程序设计.....	12
实验 6	中断调用程序设计.....	15
实验 7	串指令程序设计.....	16
实验 8	子程序设计.....	17
实验 9	汇编综合程序设计.....	20

第 1 部分 实验预备知识

汇编语言程序设计是一门实践性很强的课程。编写程序、上机调试、运行程序是进一步学习和掌握汇编语言程序设计的必要手段。建立程序、调试程序需要一些应用程序的帮助。下面先介绍一些与上机操作有关的应用程序，以为第二部分的实验做出必要的准备。

1.1 汇编语言程序的上机操作过程

一、所需文件

为运行汇编语言程序至少要在磁盘上建立以下文件：

EDIT.COM（编辑程序）或其他编辑软件

MASM.EXE（汇编程序）

LINK.EXE（连接程序）

DEBUG.EXE（调试程序，DOS 外部命令文件）


二、运行汇编语言程序的操作步骤

1. 用编辑程序建立 .ASM 源文件；
2. 用汇编程序把 ASM 文件转换成 .OBJ 文件；
3. 用连接程序把 OBJ 文件转换成 .EXE 文件；
4. 用 DOS 命令直接输入文件名就可执行该程序。

三、汇编语言源程序上机操作过程

第一步：建立汇编源程序（即：建立文件名.asm）


这个过程就相当于我们在纸上编写源程序代码一样，只不过是纸变为了计算机，这个过程也称源代码录入。将源程序代码录入计算机的方法很多，下面将介绍具体方法。

(1) 通过 windows 自带的 EDIT.EXE 文本编辑器进行输入。双击图标  或在 DOS 提示符下键入:EDIT 回车, 这时如果你系统内可调用时, EDIT 的操作画面便会出现于屏幕上, 你就可在提示下进行录入了, 当录入完毕后, 选择存盘并给你输入的文件起一个文件名, 形式: filename.asm ; (其中 filename 为你起的文件名, 由 1-8 个字符组成), asm 是为汇编程序识别而必须加上去的, 不可更改。

(2) 如果你的系统中没有 EDIT, 也可用你所熟悉的文本编辑器进行录入、编辑, 如可用 c 语言和 pascal 语言的文本编辑器来编辑, 最后将文件存为 filename.asm 的形式即可。

第二步：汇编目标文件（即：编译为.obj .lst .crf 文件）

这个过程计算机将把你编的正确的源代码编译为机器语言、程序清单及交叉引用表的目标文件。如果此时你的程序有语法错误, 系统将报错, 并指出在第几行, 什么类型的错误, 你可根据提示去逐一修改。现介绍具体过程:

双击图标  或在 DOS 提示符下键入 MASM filename 回车。这时汇编程序的输出文件可以有三个（分别: .obj .lst .crf), 便会出现三次提问, 在这可以一路回车即可。下面显示的信息是源程序中的错误个数, 如果为 0 则表示顺利通过, 就可进行进程 c。但如果不为 0 就说明有错

误，并指出错误出现的行，可依据这个提示去进行修改。但如果错误太多还未等看清就显示过去了，可用如下命令将错误信息存于一个你指定的文件，再用文本编辑器去查看。MASM filename >filen (filen 为你起的一个没用过的文件名，用以存放出错信息) 以后可查看 filen 来得到出错信息。

第三步：编译不通过，重新修改（错误类型：源程序语句不合法）

在执行过第二步后，如有出错信息时，就不能跳跃到第四步去，如果强行执行第四步，将无任何有效结果。

现在就开找错吧！首先要清楚，在第二步中检测出的错误均为每一条语句的语法或用法错误，它并不能检测出程序的逻辑设计（语句安排位置）错误，所以就要记好出错的行号。在记录行号后，就应再次执行第一步，这时的操作应是打开已编好的源程序（以 EDIT 为例：在 DOS 提示符下键入：EDIT filename.asm 回车），依据行号进行修改并存盘，再次进行汇编，直至第二步通过为止。便可继续向下执行第四步。

第四步：连接为可执行文件（即：连接为 .exe 或 .com 文件）

在这个过程中一般没有意外，如果有也就是文件名打错了。

格式：在 DOS 提示符下：LINK filename 回车

第五步：运行编译好的可执行文件

当第四步通过后，会产生一个可执行文件，这时只需运行这个程序，看它是否按你所想象那样得出结果。在试运行期间，要尽量试一些临界状态，看程序是否运行稳定、结果是否正确。如一切正常，便可进入第六步了。

可最怕的是不 OK，程序产生一些莫名其妙的结果（你可不要以为是你的计算机不听你的指挥，其实它是在一丝不苟地按照你编的程序执行。我以前总以为我的计算机出了毛病），如果是在考场上这时千万不要慌，稳住自己的情绪，先不要看计算机，静几分钟（反正时间多得是）。这时就要用到最关键、最常用的一步了，进行第六步。

第六步：用调试工具调试，重新修改（逻辑错误）

1.2 常用 DEBUG 命令的功能

一、DEBUG 的主要用途及 DEBUG 的调用

DEBUG 是为汇编语言设计的一种调试工具，它通过单步执行、设置断点等方式为汇编语言程序员提供了非常有效的程序调试手段。DEBUG 可以直接用来检查和修改内存单元、装入、存储及启动运行程序、检查及修改寄存器，也就是说 DEBUG 可深入到计算机的内部，可使用户更紧密地与计算机中真正进行的工作相联系。不仅如此，对汇编语言初学者来说，DEBUG 也是练习使用汇编指令的一种有效工具。初学者可以直接在 DEBUG 环境下执行汇编指令。然而，在 DEBUG 下运行汇编语言源程序也受到了一些限制，它不宜汇编较长的程序，不便于分块程序设计，不便于形成以 DOS 外部命令形式构成的 .EXE 文件，不能使用浮动地址，也不能使用 ASM 和 MASM 提供的绝大多数伪指令。

在 DOS 系统中，DEBUG 是以 DOS 外部命令文件形式提供给用户的，名为 DEBUG.EXE。

进入 DEBUG 的提示符是符号“-”。即，出现提示符“-”就表示可以接受 DEBUG 命令了。

当进入 DEBUG 时，寄存器和标志设成以下数值，这些值用于 DEBUG 调试中的程序。

段寄存器 CS，DS，ES 和 SS 均指向 DEBUG 末尾的第一个段。

IP 寄存器置为 0100H。栈指针 SP 指向尾部或装入程序的暂存部分的底部。

其余寄存器皆取零值，但若用户调用时含文件说明，则 CX 含文件长度（长度大于 64K 时 BX 含长度的高位）；标志为各自的复位值；驱动器传送地址在代码段位移 80H 处。

注意：若 DEBUG 装入扩展名为 .EXE 的文件，则 DEBUG 需重定位且设置段寄存器指示器为文件中所定义的值。但 DS，ES 指向最低可用段处的程序区前缀。BX 和 CX 为文件容量值。而 .EXE 文件如果在连接时选择了装入内存高处的参数，则该程序装入高处。

二、DEBUG 的主要命令功能与格式

DEBUG 命令是在 DEBUG 提示符“-”下，由键盘键入的。每条命令以单个字母的命令符开头，然

后是命令的操作参数，操作参数与操作参数之间，用空格或逗号隔开，操作参数与命令符之间用空格隔开，命令的结束符是回车键 Enter。命令及参数的输入可以是大小写的结合。Ctrl+Break 键可中止命令的执行。Ctrl+Num Lock 键可暂停屏幕卷动，按任一键继续。所用数均为十六进制数，且不必写 H。

* 1. 汇编命令 A

格式：A [[<段寄存器名> / <段地址>:] <段内偏移>]

上式等价于：

- (1) A <段寄存器名>:<段内偏移>
- (2) A <段地址>:<段内偏移>
- (3) A <段内偏移>
- (4) A

功能：键入该命令后显示段地址和段内偏移并等待用户从键盘逐条键入汇编命令，逐条汇编成代码指令，顺序存放到段地址和段内偏移所指定的内存区域，直到显示下一地址时用户直接键入回车键返回到提示符“-”。

注：其中(1)用指定段寄存器的内容作段地址，(3)用 CS 的内容作段地址，(4)以 CS:100 作地址。以后命令中提及的各种‘地址’形式，均指(1)、(2)、(3)中 A 后的地址形式。

2. 比较命令 C

格式：C <源地址范围>, <目标地址>

其中<范围>是由<起始地址> <终止地址>或者是由<起始地址> L <长度>指出的一片连续单元。

功能：从<源地址范围>的起始地址单元起逐个与目标起始地址以后的单元顺序比较单元的内容，直至源终止地址为止。遇有不一致时，以<源地址> <源内容> <目标内容> <目标地址>的形式显示失配单元及内容。

* 3. 显示内存命令 D

格式：D [<地址> / <范围>]

上式等价于：

- (1) D <地址>
- (2) D <范围>
- (3) D

功能：以两种形式显示指定范围的内存内容。一种形式为十六进制内容，一种形式为以相应字节的内容作为 ASCII 码的字符，对不可见字符以‘.’代替。

注：其中(1)以 CS 为段寄存器。(3)显示 CS:100 起始的一片内容。

* 4. 修改内存命令 E

格式：E <地址> [<单元内容表>]

上式等价于：

- (1) E <地址>
- (2) E <地址> <单元内容表>

其中<单元内容表>是以逗号分隔的十六进制数，或用‘或’括起来的字符串，或者是二者的组合。

功能：(1)不断显示地址，可连续键入修改内容，直至新地址出现后键入回车 Enter 为止。(2)将<单元内容表>逐一写入由<地址>开始的一片单元。

5. 填充内存命令 F

格式：F <范围> <单元内容表>

功能：将单元内容表中的值逐个填入指定范围，单元内容表中内容用完后重复使用。

例如：-F 5BC:200 L 10 B2, ‘XYZ’, 3C <Enter>

* 6. 执行命令 G

格式：G [= <地址>[, <断点>]]

功能：执行内存中的指令序列

注：(1)从 CS:IP 所指处开始执行

(2)从指定地址开始执行

(3)从指定地址开始执行，到断点自动停止。

* 7. 读盘命令 L

格式: L <地址> <驱动器号> <起始逻辑扇区> <所读扇区个数 n>

其中<地址>的缺省值为 CS:100。逻辑扇区可由物理扇区号换算得到,以双面双密度盘为例:物理扇区是按 0 面 0 道 1 区, 0 面 0 道 2 区, …… , 0 面 0 道 9 区, 0 面 1 道 1 区, …… , 0 面 39 道 9 区, 1 面 0 道 1 区, …… , 1 面 39 道 9 区排列。而逻辑扇区与物理扇区号的对应关系为物理扇区 0 面 0 道 1 扇区至 9 扇区, 逻辑扇区号为 0—8; 物理扇区 1 面 0 道 1 扇区至 9 扇区, 逻辑扇区号为 9—11H; 物理扇区 0 面 1 道 1 扇区至 9 扇区, 逻辑扇区号为 12—1AH; ……。这样每道先 0 面后 1 面一直排下去。

其中<驱动器号>为 0、1 或 2, 0 表示 A 驱, 1 表示 B 驱, 2 表示硬盘。

功能: 将<驱动器号>指定的盘上, 从<起始逻辑扇区>起, 共 n 个逻辑扇区上的所有字节顺序读入指定内存地址开始的一片连续单元。当 L 后的参数缺省时, 必须在 L 之前由 N 命令指定 (或进入 DEBUG 时一并指出) 所读驱动器文件名。此时 L 执行后将该文件装入内存。

例如: -N EXAMPLE <Enter>

-L <Enter>

将当前驱动器上的 EXAMPLE 文件装入 CS:100 起始的一片内存单元。

* 8. 命名待读 / 写文件命令 N

格式: N <文件名说明>

功能: 为 L / W 命令指定待装入 / 写盘文件

注: 其它形式参考 DOS 手册

9. 端口输出命令 O

格式: O <端口地址> <字节>

功能: 将该<字节>从指定<端口地址>输出。

例如: -O 2F 4F <Enter>

将 4FH 从端口 2FH 输出

* 10. 结束 DEBUG 返回 DOS 命令 Q

格式: Q

功能: 返回 DOS 提示符下

* 11. 显示修改寄存器命令 R

格式: R [<寄存器名>]

上式等价于:

(1) R

(2) R <寄存器名>

功能: (1) 显示当前所有寄存器内容, 状态标志及将要执行的下一指令的地址, 代码及汇编语句形式。其中对状态标志 FLAG 以每位的形式显示, 详见表 1-1。

表 1-1 状态标志显示形式

标志位	溢出 OF	方向 DF	中断 IF	符号 SF	零 ZF	辅助 AF	奇偶 PF	进位 CF
状 态	有 / 无	减 / 增	开 / 关	负 / 正	零 / 非	有 / 无	偶 / 奇	有 / 无
显 示	OV / NV	DN / UP	EI / DI	NG / PL	ZR / NZ	AC / NA	PE / PO	CY / NC

(2) 显示指定寄存器内容

例如: -R AX <Enter>

-R F <Enter>

* 12. 跟踪命令 T

格式: T [=<地址>] [<条数>]

功能: 执行由指定地址起始的、由<条数>指定的若干条命令。其中<地址>的缺省值是当前 IP 值, <条数>的缺省值是一条。

例如: -T <Enter>

执行当前指令并显示状态

-T 10 <Enter>

从当前指令始执行 10H 条指令

* 13. 反汇编命令 U

格式: U [<地址>/<地址范围>]

上式等价于:

- (1) U <地址>
- (2) U <地址范围>
- (3) U

功能：将指定范围内的代码以汇编语句形式显示，同时显示地址及代码。注意，反汇编时一定要确认指令的起始地址后再作，否则将得不到正确结果。地址及范围的缺省值是上次 U 指令后下一地址的值。这样可以连续反汇编。

* 14. 写盘命令 W

格式：W <地址> <盘号> <起始逻辑扇区> <所写逻辑扇区数 n>

功能：与 L 命令不同的地方是将内存从<地址>起始的一片单元内容写入指定扇区。只有 W 而没有参数时，与 N 命令配合使用将文件写盘。

注：要求读者对其中打“*”的 DEBUG 命令必须能熟练使用。

三、使用 DEBUG 调试和运行可执行文件

用户程序经过编辑、汇编、连接后得到一个可执行文件（.EXE），这时借助于调试程序 DEBUG 对用户程序进行调试，查看程序是否能完成预定功能。对于初学者，如何选用 DEBUG 中各命令，有效地调试与运行程序，需要一个学习过程。在初次使用 DEBUG 时，可参照下列步骤进行。

1. 调用 DEBUG，装入用户程序

可以在调用 DEBUG 是直接装入用户程序可执行文件，也可以在进入 DEBUG 环境后使用 N 命令和 L 命令装入用户程序可执行文件。无论用哪种方法，装入用户程序可执行文件时，一定要指定文件全名（即文件名和扩展名）。

2. 观察寄存器初始状态

程序装入内存后，用 R 命令查看寄存器内容。从各段寄存器现在的内容，便能了解用户程序各逻辑段（代码段，堆栈段等）在内存的分布及其段基值。R 命令亦显示了各通用寄存器和标志寄存器的初始值，显示的第三行就是即将执行的第一条指令。

3. 以单步工作方式开始运行程序

首先用 T 命令顺序执行用户程序的前几条指令，直到段寄存器 DS 和 / 或 ES 已预置为用户的数据段。在用 T 命令执行程序时，每执行一条指令，显示指令执行后寄存器的变化情况，以便用户查看指令执行结果。

4. 观察用户程序数据段初始内容

在第 3 步执行后 DS 和 / 或 ES 已指向用户程序的数据段和附加段，这时用 D 命令可查看用户程序的原始数据。

5. 继续以单步工作方式运行程序

对于初学者，一般编写的程序比较短，用 T 命令逐条执行指令，可清楚地了解程序的执行过程：现在执行的是什么指令，执行后的结果在哪里（寄存器，存储单元）？所得结果是否正确？等等。在逐次使用 T 命令时，若有需要，可选用 D 命令了解某些内存单元的变化情况。

用 T 命令逐条执行程序时，如遇上用户程序中的软中断指令 INT（如 INT 21H），这时，通常不要用单步工作方式执行 INT 指令。因为系统提供的软中断指令 INT 是以中断处理子程序形式实现功能调用，且这种处理子程序常常是较长的。若用 T 命令去执行 INT 指令，那么将跳转到相应的功能调用子程序中，要退出该子程序需要化费较多时间。如果既要执行 INT 指令，又要跳过这段功能调用子程序，则应使用连续工作方式（G 命令），且设置断点，其断点应为 INT 指令的下一条指令。例如要以单步工作方式执行下面一段程序：

```
10B0: 0022      MOV  DX, 0010
10B0: 0026      MOV  AH, 09
10B0: 0028      INT   21
10B0: 002A      MOV  CX, 00
```

当用 T 命令完成“MOV AH, 09”指令后，应使用 G 命令：

-G 002A <Enter>

这样，以连续工作方式实现功能调用后，即暂停在偏移量为 002A 的“MOV CX, 00”指令处（未执行），如同用单步工作方式完成 INT 指令的执行一样。

6. 连续工作方式运行程序

在用单步工作方式运行程序后，可再用连续工作方式从头开始运行程序，查看运行结果。在用

G 命令时，**注意**指定运行程序的起始地址。若 G 命令中未指定起始地址，就隐含为从当前 CS:IP 指向的指令开始。

7. 修改程序和数据

经过上面几步后，若发现程序有错，则需要适当进行修改。这时，如果仅需作个别修改，可在 DEBUG 状态下，使用 A 命令。这种修改仅仅是临时修改内存中的可执行文件，未涉及源程序。当确认修改正确后，应返回至编辑程序，修改源程序，然后再汇编、连接。

为了确认用户程序的正确性，常常需用几组不同的原始数据去运行程序，查看是否都能获得正确结果。这时，可用 E 命令在用户程序的数据段和附加段中修改原始数据，然后再用 T 命令或 G 命令运行程序，查看运行结果，直到各组数据都能获得正确结果为止。

8. 运用断点调试程序

如果已确认程序是正确的，在连续工作方式下，可快速地运行程序；如果已知程序运行结果不正确，用 G 命令运行程序，中途不停，很难查找错误。改用 T 命令，虽然可以随意暂停程序的执行，但是运行速度慢，如果运用断点，可快速查找错误。这里的“断点”是程序连续运行时要求暂停的指令位置（地址），用要求暂停的一条指令首字节地址表示。当程序连续运行到这断点地址时，程序就暂停，并显示现在各寄存器内容和下面将要执行的指令（即断点处指令）。为了准确设置断点，可用反汇编命令 U 察看源程序。运用断点，可以很快地查找出错误发生在哪一个程序段内，缩小查找错误的范围。然后在预计出错的范围内，再用 T 命令仔细观察程序运行情况，确定出错原因和位置，完成程序的调试。

第 2 部分 汇编语言程序设计实验

实验 1 Debug 基本命令及汇编基本指令

一. 实验目的

- 1、掌握 DEBUG 的基本命令及其功能，学会用 DEBUG 调试程序；
- 2、掌握 8086 CPU 指令的功能，体会寄存器的作用；
- 3、了解数据在内存中的存放方式和内存操作数的几种寻址方式；
- 4、了解简单指令的执行过程。

二. 实验内容

- 1、设 AX=3000H, BX=5000H；请编一程序段将 AX 和 BX 的内容进行交换。

要求：

- (1) 分别用 3 种方法实现；
- (2) 用 DEBUG 进行汇编与调试；
- (3) 记录每种方法执行结果

2、分别执行以下指令，比较立即寻址和直接寻址间的区别，寄存器寻址、寄存器间接寻址和相对寄存器寻址间的区别。

- (1) `Mov ax, 1000H`
`Mov ax, [1000h]`
- (2) `mov bx, 2000H`
`mov ax, bx`
`mov ax, [bx]`
`mov ax, 30[bx]`

要求：(1) 在执行以上指令时，记下当时 DS 数据段的值，写出每条指令执行后 AX 的结果。

(2) 用 E 命令修改指令偏移地址“1000h”处的值，再次执行“`Mov ax, [1000h]`”，指令，记录执行结果。

3、已知有如下程序段：

```
MOV AX, 5678H
MOV CL, 4
STC          ; 设置 CF=1
```

在以上程序段的基础上，分别执行以下指令，观察 AX 值的变化。

```
ROL AX, CL
ROR AX, CL
SHL AX, CL
SHR AX, CL
SAR AX, CL
RCL AX, CL
RCR AX, CL
```

试在 DEBUG 下用 A 命令汇编以上程序段，用 T 命令跟踪，观察以下内容：

- (1) 每条指令执行完后，AX 寄存器的内容是什么？
- (2) 每条指令执行完后，进位位 CF、符号位 SF 和零标志位 ZF 的值是什么？

三. 实验要求

- 1、预习 DEBUG 常用命令，预习所学指令。

2、实验前要做好充分准备，包括汇编程序清单、调试步骤、调试方法，以及对程序结果的分析等。

3、本实验只要求在 DEBUG 调试程序状态下进行，包括汇编程序、调试程序和执行程序。

四．实验报告书写要求

1、写出在 DEBUG 状态下编写、运行程序的过程以及调试中所遇到的问题是如何解决的，并对调试过程中的问题进行分析，对执行结果进行分析。

2、写出源程序段清单和执行结果。

3、写出本次上机的体会或收获。

实验2 汇编语言上机过程

一. 实验目的

- 1、掌握汇编语言源程序的结构
- 2、熟悉汇编语言源程序的编辑、汇编、连接、调试过程
- 3、理解和掌握变量的定义方法，系统如何为变量分配空间
- 4、理解和掌握汇编指令（LEA、ADD）的功能和用法

二. 实验内容

1、已知有以下变量定义，请将各变量分别放在源程序中进行调试，观察为每变量所分配的存储空间及初始化的数据值。

(1) BR DB 'Hello',68,-20,3 DUP(4)

(2) WR DW 3456H,0AFH,0A123H

(3) X DW 1,2,\$+4,3,4,\$+4

要求：分析、观察变量的数据分配，记录变量存储情况。

2、执行下列指令后，AX 寄存器中的内容是什么？

TABLE DW 10H,20H,30H,40H,50H

BUF DW 4

⋮

LEA BX,TABLE

ADD BX,BUF

MOV AX,[BX]

⋮

要求：(1) 将以上程序段补充成完整的汇编源程序，并调试运行。

(2) 将以上程序段中的 TABLE DW 10H,20H,30H,40H,50H 修改为 TABLE DW 10,20,30,40,50,再补充成完整的汇编程序并调试运行，观察并记录 AX 的内容。

三. 实验准备与要求

- 1、预习第3章中的所有指令，熟悉顺序程序设计方法；
- 2、预习 DEBUG 调试程序的使用方法；
- 3、根据实验内容要求，编写好实验源程序。
- 4、实验前要做好充分准备，包括汇编程序清单、调试步骤、调试方法，以及对程序结果的分析等。

四. 实验报告要求

- 1、列出源程序清单。
- 2、分析实验中所遇到的一些问题，分析错误原因。
- 3、说明本实验中是如何使用 DEBUG 进行调试的。
- 4、写出本次上机的体会或收获。

实验3 顺序程序设计

一、实验目的

- 1、熟悉运算类指令对标志位的状态影响以及标志位状态的表示方法；
- 2、熟悉逻辑类指令的用法；
- 3、掌握最基本的程序设计方法。

二、实验内容

- 1、求内存单元中已定义的两个数据（855CH 与 AB43H）之差值。下面已给出一种方法，请用定义字的方法改写实现。

```
DATA    SEGMENT
A  DB   5CH, 85H
B  DB   43H, 0ABH
DATA    ENDS
CODE    SEGMENT
        ASSUME  CS:CODE, DS:DATA
START:
        MOV     AX, DATA
        MOV     DS, AX
        MOV     SI, 0
        MOV     AL, A[SI]
        SUB     AL, B[SI]
        MOV     A[SI], AL
        INC     SI
        MOV     AL, A[SI]
        SBB     AL, B[SI]
        MOV     A[SI], AL
        MOV     AH, 4CH
        INT     21H
CODE    ENDS
        END     START
```

- 2、试编写一程序，实现将一存放在 DX、AX 中的 32 位操作数循环右移 4 位。

三、实验准备与要求

- 1、复习数据传送和减法等运算类指令
- 2、请运行以上源程序，并在 DEBUG 状态下观察程序运行结果。

四、实验报告要求

- 1、程序说明。说明程序的功能、结构。
- 2、调试说明。包括上机调试的情况、上机调试步骤、调试所遇到的问题是怎样解决的，并对调试过程中的问题进行分析，对执行结果进行分析。
- 3、写出源程序清单和执行结果，必要时画出流程图。

实验4 分支程序设计

一、实验目的

- 1、熟悉运算类指令对标志位的状态影响以及标志位状态的表示方法；
- 2、掌握条件转移、无条件转移指令的使用方法；
- 3、掌握分支程序设计、编写、调试和运行的方法。

二、实验内容

1、编写程序计算 $|X-Y|$ 的值，其中：X和Y为存放于X单元和Y单元的16位操作数，要求将结果存入result单元中。部分程序如下，要求填充完整的程序段：

```
        :  
        MOV AX,X  
        SUB AX,Y  
        JNS NONNEG  
        NEG AX  
NONNEG: MOV RESULT,AX  
        :  
        :
```

2、编一汇编语言程序，实现统计DX数据中所含1的个数。

三、实验准备与要求

- 1、复习条件转移指令和无条件转移指令；
- 2、熟悉分支程序设计方法；
- 3、实验前要做好充分准备，包括汇编程序清单、调试步骤、调试方法，以及对程序结果的分析等。

四、实验报告要求

- 1、程序说明。说明程序的功能、结构。
- 2、调试说明。包括上机调试的情况、上机调试步骤、调试所遇到的问题是怎样解决的，并对调试过程中的问题进行分析，对执行结果进行分析。
- 3、写出源程序清单和执行结果，画出流程图。

实验 5 循环程序设计

一、实验目的

- 1、掌握循环指令 LOOP、LOOPZ、LOOPNZ 的使用方法；
- 2、掌握实现单重、多重循环程序设计、编写、调试和运行的方法。
- 3、掌握串指令的应用。

二、实验内容

1、编写程序，实现将内存偏移地址为 1000H 开始的连续 100 个字节送往偏移地址为 2000H 开始的连续 100 个内存单元中。

2、在数据段变量名为 A 的数据区内有 10 个字符，编程实现将这 10 个字符以相反次序传送到附加段变量名为 B 的内存区中。

源程序已部分给出，请将程序补充完整，并调试运行结果。

```
data segment
A db '1234567890'
n equ $-a
B db  n dup(?)
data ends

code segment
    assume cs:code,ds:data
start:
    mov ax,data
    mov ds,ax
    lea si,a
    lea di,b
    add di,_____
    mov cx, ____

move:
    mov al,[si]
    mov [di],al
    inc si

    _____
    loop move
    mov ah,4ch
    int 21h
code ends
    end start
```

3、数据段开始区域中，连续存放着 10 个无符号数，编程序找出这 10 个数中最大的一个数，并将其存到该数据区的后面。

三、实验准备与要求

- 1、复习循环指令 LOOP；
- 2、熟悉循环程序设计方法，写出以上程序的源程序清单。
- 2、仔细观察在 DEBUG 下的调试过程。

四、实验报告要求

- 1、列出程序清单和执行结果；
- 2、分析实验中所遇到的一些问题。包括上机调试的情况、上机调试步骤、调试所遇到的问题是
如何解决的，并对调试过程中的问题进行分析，对执行结果进行分析。

实验 6 DOS 中断调用程序设计

一. 实验目的

- 1、了解 DOS 中断功能调用的概念；
- 2、掌握使用 INT 21H 中断的 1#、2#、9# 和 10# 功能调用。

二. 实验内容与要求

1、以下程序是一 9# 和 10# 功能相结合的例子, 具体实现: 从键盘上读入一串指定长度的字符, 然后利用 9 号系统功能调用显示输出该串字符。

请通过实际操作, 理解 10# 功能及部分关键指令。

```

Data    SEGMENT
message db 'please input a string:$'
buf      DB 20,?,20 DUP(0)
data    ends
code segment
        ASSUME CS:code,DS:Data
GO:     MOV AX,Data
        MOV DS,AX
        mov dx,offset message
        mov ah,9
        int 21h
        mov dx,offset buf
        mov ah,10
        int 21h
        mov ah,2
        mov dl,0ah
        int 21h
        mov dl,0dh
        int 21h
        mov bl,buf+1
        mov bh,0
        mov byte ptr buf+2[bx], '$'
        mov dx,offset buf+2
        mov ah,9
        int 21h
        MOV AH,4CH
        INT 21H
code    ENDS
        END GO

```

2、编程实现: 从键盘输入一个字符, 判断其是不是大写字母, 如果是则请输出这个大写字母, 如果不是, 请输出“这不是一个大写字母”的英文信息 (要求: 能连续输入)。

三. 实验准备与要求

- 1、复习 1#、2#、9# 和 10# 功能调用；
- 2、仔细观察在 DEBUG 下的调试过程。

四. 实验报告要求

- 1、程序说明。说明程序的功能、结构。
 - 2、调试说明。包括上机调试的情况、上机调试步骤、调试所遇到的问题是
- 如何解决的，并对调试过程中的问题进行分析，对执行结果进行分析。
- 3、写出源程序清单和执行结果，并给出程序流程图。
 - 4、分析实验结果及所遇到的问题的解决方法。
 - 5、体会和意见。

实验 7 串指令程序设计

一. 实验目的

- 1、掌握通过串传送数据与 MOV 指令间的区别；
- 2、理解串指令特点；
- 3、理解串指令与高级语言中数组间关系

二. 实验内容

1、编写程序，实现将内存偏移地址为 1000H 开始的连续 100 个字节送往偏移地址为 2000H 开始的连续 100 个内存单元中。要求使用二种不同的方法（用单一的串操作指令、用带重复前缀的串操作指令）。

2、编程实现：对 str1 和 str2 两个字符串进行比较，若两串相同，在 result 单元中置 1，否则置 -1。

三. 实验要求

复习相关的串操作指令

四. 实验报告要求

- 1、程序说明。说明程序的功能、结构。
- 2、调试说明。包括上机调试的情况、上机调试步骤、调试所遇到的问题是怎样解决的，并对调试过程中的问题进行分析，对执行结果进行分析。
- 3、写出源程序清单和执行结果，并给出程序流程图。
- 4、分析实验结果及所遇到的问题的解决方法。
- 5、体会和意见。

实验 8 子程序设计

一. 实验目的

- 1、掌握主程序与子程序之间的调用关系及其调用方法；
- 2、掌握子程序的调用与返回的方法；
- 3、掌握 CALL 指令的使用方法。掌握子程序设计、编写、调试和运行的方法；
- 4、了解子程序的嵌套与递归。
- 5、了解参数传递的三种不同方法

二. 实验内容与要求

- 1、编程求下列和值：

$$S = (1+2+3) + (1+2+3+4) + (1+2+3+4+5)$$

将和值存在数据段偏移量为 0100H 的单元中。

- 2、设有 10 个学生成绩分别是 76、69、84、90、73、88、99、63、100 和 80 分。试编制一个子程序统计 60~69 分，70~79 分，80~89 分，90~99 分和 100 分的人数并分别存放到 S6、S7、S8、S9 和 S10 单元中。

三. 实验准备

复习子程序设计的基本方法，根据实验内容要求编写出实验程序。

四. 实验报告要求

- 1、程序说明。说明程序的功能、结构。
- 2、调试说明。包括上机调试的情况、上机调试步骤、调试所遇到的问题是怎样解决的，并对调试过程中的问题进行分析，对执行结果进行分析。
- 3、写出源程序清单和执行结果，并给出程序流程图。
- 4、分析实验结果及所遇到的问题的解决方法。
- 5、体会和意见。

实验 9 汇编综合程序设计

一、实验目的

- 1、通过前面所学的汇编语言结构（顺序、分支、循环和子程序结构）以及 DOS 功能调用等基本的程序结构来设计一个综合程序，达到学以致用，举一反三；
- 2、掌握模块化程序的设计方法；
- 3、掌握综合程序的编制及调试方法。

二、实验内容

- 1、编程实现由键盘输入任意一个字符，将该字符的 ASCII 码值显示在屏幕上（要求：分别以二进制和十六进制形式显示）。
- 2、编写一个程序, 实现把存放在 BX 寄存器内的二进制数以十六进制数的形式显示在屏幕上。
- 3、现在 block 数据区有 20 个字数据，按要求编写程序：
 - (1) 将正数和负数分开存放；
 - (2) 统计其中的正数和负数个数，并将结果放在各区前面。
- 4、在以 BUF 为首址的字存储区中存放有 N 个有符号数，现将它们按由小到大的顺序排列后再放回 BUF 存储区中，试编程实现。

三、实验要求

- 1、复习涉及到的相关内容
- 2、总结前面的理论知识、实验过程和方法

四、实验报告要求

- 1、程序说明。说明程序的功能、结构。
- 2、调试说明。包括上机调试的情况、上机调试步骤、调试所遇到的问题是怎样解决的，并对调试过程中的问题进行分析，对执行结果进行分析。
- 3、写出源程序清单和执行结果，并给出程序流程图。
- 4、分析实验结果及所遇到的问题的解决方法。
- 5、体会和建议。