高等院校规划教材
计算机科学与技术系列

汇编语言程序设计实验与习题解答

何　超　主编

机械工业出版社
CHINA MACHINE PRESS

高等院校规划教材·计算机科学与技术系列

汇编语言程序设计实验与习题解答

何　超　主编
胡安明　王煜林　田桂丰　参编

机械工业出版社

本书是《汇编语言程序设计》（ISBN 978-7-111-27260-1）的配套教材。全书共2章，第1章介绍了汇编语言程序的开发过程，并给出了8个实验；第2章是《汇编语言程序设计》第1～7章的习题解答。本书的主要特点是：以模拟实际开发中的汇编程序设计为例，用通俗易懂、由浅入深、由简到繁、循序渐进的方式展开讲解。

本书可作为信息类专业（计算机、自动控制、电工电子等）本科生的教材，也可供从事相关技术工作的人员和感兴趣的读者作为参考书或自学读物。

图书在版编目（CIP）数据

汇编语言程序设计实验与习题解答 / 何超主编. —北京：机械工业出版社，2009.12

（高等院校规划教材·计算机科学与技术系列）

ISBN 978-7-111-28333-1

Ⅰ. 汇… Ⅱ. 何… Ⅲ. 汇编语言－程序设计－高等学校－教学参考资料 Ⅳ. TP313

中国版本图书馆CIP数据核字（2009）第165989号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）

责任编辑：陈 皓

责任印制：李 妍

北京汇林印务有限公司印刷

2010年4月·第1版第1次印刷

184mm×260mm·8.75印张·211千字

0001－3000册

标准书号：ISBN 978-7-111-28333-1

定价：17.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务

社服务中心：（010）88361066

销 售 一 部：（010）68326294

销 售 二 部：（010）88379649

读者服务部：（010）68993821

网络服务

门户网：http://www.cmpbook.com

教材网：http://www.cmpedu.com

封面无防伪标均为盗版

出版说明

计算机技术的发展极大地促进了现代科学技术的发展，明显地加快了社会发展的进程。因此，各国都非常重视计算机教育。

近年来，随着我国信息化建设的全面推进和高等教育的蓬勃发展，高等院校的计算机教育模式也在不断改革，计算机学科的课程体系和教学内容趋于更加科学和合理，计算机教材建设逐渐成熟。在“十五”期间，机械工业出版社组织出版了大量计算机教材，包括“21 世纪高等院校计算机教材系列”、“21 世纪重点大学规划教材”、“高等院校计算机科学与技术‘十五’规划教材”、“21 世纪高等院校应用型规划教材”等，均取得了可喜成果，其中多个品种的教材被评为国家级、省部级的精品教材。

为了进一步满足计算机教育的需求，机械工业出版社策划开发了“高等院校规划教材”。这套教材是在总结我社以往计算机教材出版经验的基础上策划的，同时借鉴了其他出版社同类教材的优点，对我社已有的计算机教材资源进行整合，旨在大幅提高教材质量。我们邀请多所高校的计算机专家、教师及教务部门针对此次计算机教材建设进行了充分的研讨，达成了许多共识，并由此形成了“高等院校规划教材”的体系架构与编写原则，以保证本套教材与各高等院校的办学层次、学科设置和人才培养模式等相匹配，满足其计算机教学的需要。

本套教材包括计算机科学与技术、软件工程、网络工程、信息管理与信息系统、计算机应用技术以及计算机基础教育等系列。其中，计算机科学与技术系列、软件工程系列、网络工程系列和信息管理与信息系统系列是针对高校相应专业方向的课程设置而组织编写的，体系完整，讲解透彻；计算机应用技术系列是针对计算机应用类课程而组织编写的，着重培养学生利用计算机技术解决实际问题的能力；计算机基础教育系列是为大学公共基础课层面的计算机基础教学而设计的，采用通俗易懂的方法讲解计算机的基础理论、常用技术及应用。

本套教材的内容源自致力于教学与科研一线的骨干教师与资深专家的实践经验和研究成果，融合了先进的教学理念，涵盖了计算机领域的核心理论和最新的应用技术，真正在教材体系、内容和方法上做到了创新。同时本套教材根据实际需要配有电子教案、实验指导或多媒体光盘等教学资源，实现了教材的“立体化”建设。本套教材将随着计算机技术的进步和计算机应用领域的扩展而及时改版，并及时吸纳新兴课程和特色课程的教材。我们将努力把这套教材打造成为国家级或省部级精品教材，为高等院校的计算机教育提供更好的服务。

对于本套教材的组织出版工作，希望计算机教育界的专家和老师能提出宝贵的意见和建议。衷心感谢计算机教育工作者和广大读者的支持与帮助！

机械工业出版社

前　言

本书是《汇编语言程序设计》的配套教材。全书共 2 章，第 1 章介绍了汇编语言程序的开发过程，并给出了 8 个实验，通过实验使读者进一步了解汇编语言程序的开发过程，熟悉汇编语言程序的调试方法；第 2 章是《汇编语言程序设计》第 1～7 章的习题解答，大量的习题（主要是编程训练）对于学习汇编语言是必不可少的，读者应该通过做题来尽快掌握相关知识。习题解答只是一个参考，请读者正确对待它，万万不可以抄袭来应付作业，程序的编写有多种形式，绝不止解答给出的一种，读者可以尝试其他的答案，看一看能否收到异曲同工的效果。

本书的主要特点是：以模拟实际开发中的汇编程序设计为例，用通俗易懂、由浅入深、由简到繁、循序渐进的方式展开讲解。

本书由何超主编，实验部分由胡安明和何超编写，习题解答部分主要由王煜林编写，胡安明、田桂丰也完成了部分习题解答的编写。

本书可作为计算机、自动控制、电工电子等信息类专业本科生的教材，也可供从事相关技术工作的人员和感兴趣的读者作为参考书或自学读物。

鉴于作者的水平有限，书中不妥之处在所难免，敬请广大读者批评指正。

编　者

目　录

第1章　汇编语言程序开发过程和实验

本章详细地介绍了汇编语言程序开发的过程，并给出了8个相关实验。

1.1　汇编语言程序开发过程

汇编语言程序设计的基础是编程。在设计程序时，要确保程序的正确性，同时要使程序结构合理，简洁明了，容易阅读和交流，便于使用和扩充，节省存储空间，具有较高的运行速度。

1.1.1　程序设计的步骤

汇编语言程序设计是用计算机的机器指令、伪指令和宏指令等编写解决某一问题的指令代码的过程。

用汇编语言进行程序设计的一般步骤为：分析问题，建立数学模型，选择算法，编写程序和上机调试。

1．分析问题

要分析问题，首先要对问题有一个确切理解，明确问题的来龙去脉，弄清已知条件，有哪些原始数据，最后应获得什么结果。另外，需要了解解决问题的环境限制，对运算精度、处理速度的要求等。总之，正确地分析问题是进行程序设计的基础。

2．建立数学模型

在确切理解问题的基础上，建立一个数学模型，即对问题用简洁而严明的数学方法进行严格或近似地描述，把一个实际问题化成一个计算机可以处理的问题。

有些实际问题比较简单，数学模型比较容易建立。有些实际问题比较复杂，没有现成的公式或模型可以套用，此时往往需要工作人员经过若干次实验，取得大量数据，再利用数理统计方法对客观现象和过程进行有限度的抽象（既要考虑其普遍性，又要考虑其特殊性），最后归纳总结成数学模型。对于这项工作的实施，有时程序设计人员难以胜任，需要由具有实践经验的工程技术人员或数学工作者来完成，但程序设计人员必须对已建立的模型有深刻而透彻的理解。

3．设计算法

数学模型建立后，就要找到算法，确定在计算机上由哪些逻辑步骤及顺序去实现它。算法是一组有穷的规则，它规定了解决某一特定类型问题的一系列运算。通俗一点说，就是把解决问题的方法与步骤具体化。如果有几种解决方法，在对方法进行选择时，需根据问题的要求，选择较优的算法，从而使处理逻辑简单，处理速度更快，且容易获得满意的结果。

设计或选择算法时，必须注意3点：

1）算法中的每一种运算必须有确切的定义，无二义性。

2）组成算法的处理步骤是有限的。如果某类问题不能用有限的步骤获得结果，则称这

类问题不存在算法，只存在过程。

3）算法的可行性。技术人员必须结合所使用计算机的软硬件资源（如存储空间，运行速度），来考虑算法是否可行。算法可以用自然语言、半自然语言、类程序设计语言或流程图来表述。

关于流程图的应用，应注意的问题包括：当问题比较简单时，不必用流程图来表示处理过程，直接编写程序即可；如果问题和算法比较复杂，尤其是在出现多个分支和逻辑判断时，一定要画出流程图表示处理过程；流程图的设计要尽量分层次、表述详细。

使用流程图的优点如下。

1）流程图描述的算法结构清晰、直观、形象，便于了解程序全貌和各程序段之间的逻辑关系；便于相互交流。

2）有助于查找和分析程序的逻辑错误。

流程图的缺点是：不能表示数据的类型和结构，也不便于作为文件保存在计算机外存中。

4．编写程序

用计算机机器指令助记符号或语句实现算法的过程，就是编写程序。为达到此目标，需要正确地使用各种指令和寻址方式，灵活地应用程序设计的各种方法与技巧。而这些知识很难全部掌握，只有在实践中逐步训练，排除错误，总结经验，才能不断提高程序设计的能力。

编写汇编语言源程序时，要考虑以下几方面的问题。

1）内存空间的分配。例如，程序中所使用的数据段、附加段、堆栈段和代码段存放在内存中的什么位置（如不明确指出，则由操作系统分配）；原始数据、中间结果及最终结果存放在内存中的什么位置，需要占用多大的存储空间；堆栈段需要多大空间等。根据设计要求，按操作数数值的大小和个数的多少来分配存储空间。操作数因其值的大小不同，有的可能需要占用一个字节，有的可能需要占用一个、两个或多个字的存储空间。例如，为 n 名学生的成绩登记表分配存储空间，假定每名学生的登记项包含学号、分数 1、分数 2、…、分数 9、总分。其中，学号不大于 65535，单科分数不高于 100，则每个学生的登记项定义如下：

```
XH    DW   学号
DKF   DB   分数1，分数2，…，分数9
SUM   DW   总分
```

由上述定义可知，学号占一个字，每个单科分占一个字节，总分占一个字。这样既不浪费存储空间，又能完整地表示各项的值。

若总分也用一个字节来存放，则可能放不下；若每个单科分数用一个字存放，虽然其值是正确的，但会造成存储空间浪费（因为一个单科分只占用一个字节）。

2）要选择合适的寻址方式，注意各种寻址方式之间的区别。例如，“ADD [SI]，SI”的功能是([SI])+(SI)→[SI]，而“ADD SI，SI”的功能是(SI)+(SI)→SI。前者未改变 SI 的内容，后者使 SI 的内容增加了 1 倍。如果每条语句执行前已知(SI)=100，(100)=200，则前一条语句的执行使(100) =300，但(SI) =100 不变。后一条语句的执行使(SI)=200。

3）选用合适的指令。选择合适的指令，使程序能正确运行。选用指令时，要注意以

下几点。

① 要注意字节操作与字操作的区别。

例如，累加字节数时，只能用字节数相加。而总和占一个字，则每累加一次字节数时要将其进位加到高字节中。

如果将“ADD AL，[BX]”误写成“ADD AX，[BX]”，则每次累加的不是一个字节操作数，而是一个字操作数。显然，这个结果是错误的。由于被累加的是字节操作数，故每累加一次后，BX 的内容增 1，指向下一字节数。如果不小心将 AL 写成 AX，累加一次后，指针 BX 增 2，则累加了两字节的内容，这也是错误结果。

又如，如果要往 SI 所指字节存储单元中送 12，相应的指令是“MOV [SI]，BYTE PTR 12”。若将“BYTE”写成了“WORD”，则 12 送入了 SI 所指的字单元，这不符合本来意愿。

② 要选用效率高的指令。

例如，用“INC AX”，而不用“ADD AX，1”；用“DEC AX”，而不用“SUB AX，1”。

③ 不要插入不必要的语句。

例如：

```
DEC     BX
CMP     BX，0
JNE     LOPA
```

此处，语句“CMP BX，0”是多余的。

又如：

```
MOV     AX，A
MOV     BX，B
CMP     AX，BX
```

此处，可将后两条语句改为一条：“CMP AX，B”。

④ 严格遵守寻址方式的格式。

例如，“MOV 100，AX”是错误的，目的操作数不允许用立即方式。

又例如，将 A、B 两个字单元中的操作数相加，不能用语句“ADD A，B”，因两个存储器操作数不能出现在同一条指令中。

再例如，“MOV [BX]，BYTE PTR 12”，其中，“BYTE PTR”是不可省的，因为当立即数与存储器操作数进行运算时，一定要指出其长度。

还例如，若 A 为字节变量，则语句“ADD AX，A”是错误的，因为两个操作数的类型要匹配，其中目的操作数只能是字节寄存器，“ADD AL，A”才是合法的。

4）合理安排源程序的格式。按正确的格式书写源程序，可提高编写程序的效率，便于检查、调试和相互交流。在书写程序时，应注意以下几点。

① 在每个程序的开始处应简介程序的功能，包括所用算法、使用的寄存器、重要的符号地址，以及输入/输出数据。

② 一般来说，每条语句后面都应有注释。对于不便阅读的部分，要给予完整的注释。

注释的内容要简洁明了，能清楚地表明此语句在程序中所起的作用。

如果注释内容一行写不下，则可分多行写。其中，每行注释均以分号开头。输入到机器中的注释对程序的目的代码无任何影响，只是便于阅读。

③ 语句与标号要有固定的间隔。语句中应有的空格和标志符号不可省略。

5）注意程序中所使用数据的表示方法（精度），输入/输出数据的方法、格式，以及输入/输出设备的种类。

5. 上机调试

源程序编写完后，送入计算机进行汇编、连接和调试。汇编程序可以检查源程序中的语法错误，调试人员根据提示出的语句错误，进行相应的修改，直至无语法错误为止，再利用纠错程序调试。

纠错程序（DEBUG）是调试程序的工具，它提供了极为方便的调试手段，如设置断点、逐条跟踪、检查修改内存单元和寄存器内容等。程序调试人员要准备一组或几组不同的数据使计算机能够执行组成程序的每条指令，以验证程序是否按预想的逻辑顺序执行并获得预期结果。如发现问题，修改程序直至达到设计要求为止。对于一个大型程序，编写程序与调试程序的时间大致相同，甚至调试程序比编写程序所花费时间还多。其具体操作方法在《汇编语言程序设计》一书中已有详述。

1.1.2 汇编语言开发的几种环境

虽然汇编语言在编写方式上和其他计算机语言有很大的不同，但是它仍属于计算机语言，同样需要开发环境的支持。大致上，汇编语言的开发环境可以分为 3 类。

1. Debug 调试环境

Debug 是 DOS 操作系统下的一个强有力的汇编语言调试、编写工具。在 Debug 中，不但可以很容易地调入任意一个.exe 文件进行反汇编，而且能对该源代码进行单步调试、跟踪等。在 Debug 中，可以直接输入汇编语言指令来编写程序。但是 Debug 毕竟只是调试环境，并不适合于编写较长的程序。

2. DOS 下的汇编语言开发工具

在 DOS 操作系统下的开发工具，主要以 MASM 为代表。MASM（Microsoft Macro Assembler）是微软公司为 x86 微处理器家族所写的一套集成开发环境。最初，它是用来开发 MS-DOS 下的软件的。它支持多种汇编语言格式的宏。它支持的宏很灵巧，既具广泛性又具多样性。在 DOS 下，汇编语言的主流开发环境一直都是 MASM。MASM 先后发展了多个版本，其中使用最为广泛的是 Masm 5.0 和 Masm 6.0。从 Masm 6.11 版本开始，微软公司已不在提供商业版本了，转为提供一款全免费的 MASM。至今，微软公司已多次更新 MASM 的版本，最近的几个版本为：

- 6.15 版，放在 Visual C++ 6.0 Processor Pack 中。
- 7.0 版，放在 Visual C++ .NET 2002 中。
- 7.1 版，放在 Visual C++ .NET 2003 中。
- 8.0 版，放在 Visual C++ .NET 2005 中，它也可以组译 x64 的程式码。

3. Windows 下的汇编语言开发环境

Windows 操作系统下的汇编语言开发环境非常多，但使用比较广泛的是 Emu 8086。

Emu 8086 Microprocessor Emulator 结合了先进的原始编辑器、组译器、反组译器、具有除错功能的软件模拟工具（虚拟 PC），以及一个循序渐进的指导工具。对初学汇编语言的人来说，这是一款很有用的工具。它会在模拟器中一步一步地编译程序码并执行，视觉化的工作环境更容易使用。用户可以在程序执行过程中检视寄存器、标志位及存储单元。汇编语言程序会在虚拟 PC 中执行程序，从而隔绝用户的程序，避免访问计算机。而在虚拟机器上执行组合程序，这让除错变得更加容易。本书后面的很多代码都是在 Emu 8086 下调试的。

1.2 实验一 在 Debug 环境下调试程序代码的方法

【实验课题】 在 Debug 中调试如下两段代码。

```
（1）MOV   AX，FEDC
     MOV   BX，CDAE
     ADD   AX，BX
     SUB   AX，BX
（2）MOV   AH，01
     INT   21H
     SUB   AL，20
     MOV   DL，AL
     MOV   AH，02
     INT   21 H
```

【实验目的】 熟悉和掌握 Debug 基本命令。

【实验步骤】

（1）第一段代码的调试过程

1）选择“开始”→“运行”命令，在弹出的“运行”对话框中输入 CMD，进入命令行模式（见图 1-1）。

2）在命令行窗口中输入 Debug，并按〈Enter〉键。

3）可以看到，Debug 模式的标示和命令行模式是不一样的。

4）这时，在 Debug 模式中输入命令：-a100，进入编辑状态（见图 1-1）。

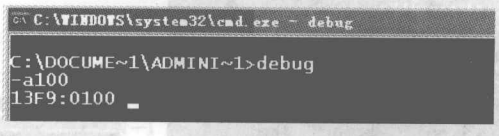

图 1-1 Debug 调试界面 1

5）可以看到，在窗口中显示出“13F9:0100”这样一个提示。其中，13F9 代表段地址，而 0100 代表偏移地址，编写的指令将从这个地址开始存放（注意，不同的计算机在不同的时间，所分配的段地址可能不同）。

6）输入第一段代码，输入完成后，按两次〈Enter〉键结束。接着输入命令-t=100 4，开始调试（见图 1-2）。

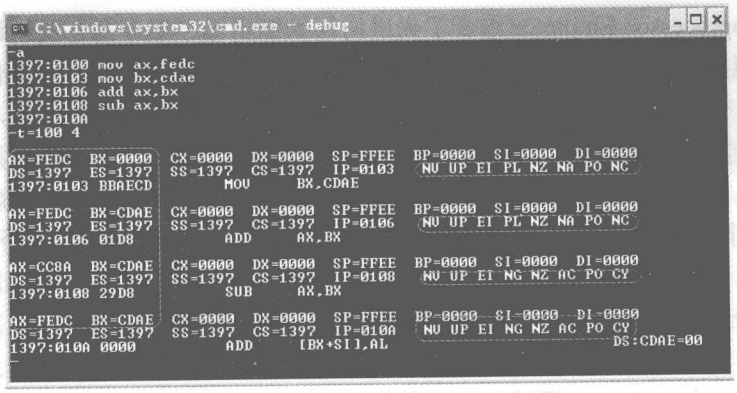

图 1-2　Debug 调试界面 2

7）实验记录：记录程序调试过程中出现的问题及解决过程。注意观察 AX、BX 及 PSW 寄存器的值的变化，并加以简要说明。

（2）第二段代码的调试过程

1）按照调试第一段代码的过程，使用命令–a100，输入第二段代码。

2）由于第二段代码中调用了中断，不能直接调试，需在 Debug 中编译存盘成.com 格式的文件才能运行，下面介绍这种调试方式。

输入完代码后，输入如下命令：

```
–r cx
:0c
–n  demo.com
–w
```

demo.com 是编译后存盘的文件名，可以自己随意定。当显示“Writing 0000C bytes”时，表示存盘成功。整个过程如图 1-3 所示。

3）退出 Debug 后，在 DOS 状态下调用该程序。

4）输入 v，输出 V，实现了大小写转换，如图 1-4 所示。

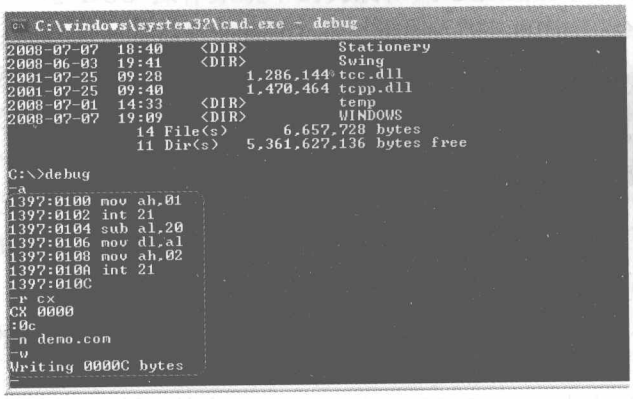

图 1-3　Debug 调试界面 3

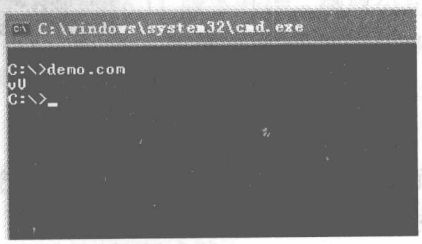

图 1-4　Debug 调试界面 4

1.3 实验二　分支程序设计与调试

1.3.1 分支程序设计基础知识

所谓程序，也就是按照一定流程执行的指令序列，而这一流程在计算机中可分为 3 类，分别是顺序、分支和循环。其中，顺序是按指令的排列顺序执行，如同记流水账一样。而后两类是根据计算机的逻辑判断能力来作出相应的处理，决定计算机的指令序列执行。

计算机进行逻辑判断有真、假两种判断结果，根据判别的真假，可执行两组不同的指令序列，实现这一功能的是转移指令。因此，所谓的分支和循环结构，是根据不同的逻辑判断值来作出不同的转移选择。

例如，设 AX 与 BX 中各存放着一个无符号字数据，将其中较小的数送入 MIN 单元，可编写出如下程序段：

```
        ⋮
        CMP   AX，BX      ；两数比较
        JB    MOVE        ；AX 中的数小时，转到 MOVE
        MOV   AX，BX      ；将较小的数存入 AX
MOVE:   MOV   MIN，AX     ；将较小的数存入 MIN 单元
        ⋮
```

上述程序中的“JB　MOVE”指令就是用来判别 AX 与 BX 的比较结果的，若 AX 低于 BX，则条件为真，转向 MOVE 执行；否则条件为假，顺序执行程序。

又例如，设 AX 中有一带符号数据，若 AX 中为正数，将其存入 PLUS 单元，否则存入 MINUS 单元，可编写如下程序：

```
        AND   AX，AX
        JS    MOVE        ；测试 AX 的符号，若为负，转到 MOVE
        MOV   PLUS，AX    ；AX 中为正数
        ⋮
MOVE:   MOV   MINUS，AX   ；AX 中为负数
```

从上述两个例子可以看出，要掌握分支和循环这两种程序结构，关键在于理解并掌握转移指令的使用。

转移指令在汇编语言中分成两大类：无条件转移和条件转移。

在分支程序设计中，有时会遇到多分支结构，如果用条件转移指令来实现，一条转移指令只有两个分支，N 个分支就需要 N–1 条条件转移指令，这样程序会显得很复杂。使用地址法，可以很容易地处理这一问题。

地址法的设计思路是：在数据段中定义一个地址列表，将各个分支程序的入口地址存放在所定义的列表中，然后根据不同的条件形成相对于地址表首地址的偏移地址。

1.3.2 分支程序设计与调试实验

1．单分支结构实验

【实验课题】 编写程序，计算下列公式。

$$Y=\begin{cases}1 & X>0\\0 & X=0\\-1 & X<0\end{cases}$$

【实验目的】 通过程序设计与调试，加深对单分支结构程序设计方法的认识，并掌握其调试方法。

【实验步骤】

1）编程思路分析：共分三种情况，需要用两个条件转移指令。判断完 X≥0 后，用 JGE 指令来处理，以分离出 X＜0 的情况。再判断 X＝0，还是 X＞0，用 JZ 指令来处理。

2）编写程序。

其具体参考程序如下。

```
DATA   SEGMENT
       X   DW   ?
       Y   DW   ?
DATA   ENDS
CODE   SEGMENT
           ASSUME   CS :CODE，DS :DATA
START: MOV   AX，DATA
           MOV   DS，AX
           MOV   X，－2                ；给 X 赋初值
           CMP   X，  0                ；判断 X≥0？
           JGE    BIN                  ；X≥0 时，转移到 BIN
           MOV   Y，－1
           JMP    NEXT
BIN:       JZ     EQUL
           MOV   Y，1
           JMP    NXET
EQUL:  MOV   Y，0
NEXT：MOV   AH，4CH
           INT    21H
CODE   ENDS
           END    START
```

3）程序调试：调试步骤类似实验一。

4）实验记录：记录程序调试过程中出现的问题及解决过程，并加以简要说明。

【练习】编写程序，实现将 3 个数按从小到大的顺序输出。

2．多分支结构实验

【实验课题】 用地址表示法编写多分支结构程序，通过键盘输入 0～9 之间的任意一个数，同时在屏幕上输出该数。

要求：

- 输入 0 时，转到 R0 标号处，同时屏幕输出‘0’。
- 输入 1 时，转到 R1 标号处，同时屏幕输出‘1’。
- 输入 2 时，转到 R2 标号处，同时屏幕输出‘2’。

【实验目的】 通过程序设计和调试，加深对多分支结构程序设计方法的认识，并掌握其调试方法。

【实验步骤】

1）编程思路分析：实现此课题，有以下两点很关键。

① 如何将标号地址的存放形成地址列表。

符号地址，也就是偏移地址，是一个 16 位数据，这时可以在数据段中定义一个 DW 型的列表变量，存放 R0～R9。即：

```
TABLE  DW  R0，R1，R2，R3，R4，R5，R6，R7，R8，R9
```

② 如何从键盘输入数据，以及如何根据输入的数据来访问地址列表。

可以通过调用 DOS 中断来读取键盘的输入数据，但在汇编语言中键盘输入的任何数据都是 ASCII 码，0 即 30H、1 即 31H，所以需要一定的处理。其实现代码如下：

```
MOV  AH，01H          ；调用 DOS 的 01H 中断，即从键盘读入一个字符
INT  21H
SUB  AL，30H          ；将读入的字符转换为数字，即减 30H
```

解决完数据输入后，便要来访问地址列表了。在访问时，由于 TABLE 的数据类型是 DW，一个偏移地址要占两个单元格，例如，访问 R3，其实 R3 是存放在第 6 和第 7 个格中的，那么访问 R3 时要将输入的数据乘以 2。其实现代码如下：

```
LEA  BX，TABLE        ；取 TABLE 的首地址
MOV  AH，0H           ；清空 AH
ADD  AX，AX           ；将 AL 中的数据×2
ADD  BX，AX
JMP  WORD  PTR  [BX]；转移访问
```

2）编写程序：其具体参考程序如下。

```
DATA  SEGMENT
      TABLE  DW  R0，R1，R2，R3，R4，R5，R6，R7，R8，R9
DATA  ENDS
CODE  SEGMENT
      ASSUME  CS:CODE，DS:DATA
START：MOV  AX，DATA
      MOV  DS，AX
      MOV  AH，01H
      INT  21H
      SUB  AL，30H
      CMP  AL，00H
      JB   EXIT
      CMP  AH，09H
      JA   EXIT
      MOV  BX，OFFSET  TABLE
      MOV  AH，0
```

```
        ADD  AX，AX
        ADD  BX，AX
        JMP  WORD  PTR  [BX]
R0:     MOV  DL，'0'
        JMP  PRINT
R1:     MOV  DL，'1'
        JMP  PRINT
R2:     MOV  DL，'2'
        JMP  PRINT
R3:     MOV  DL，'3'
        JMP  PRINT
R4:     MOV  DL，'4'
        JMP  PRINT
R5:     MOV  DL，'5'
        JMP  PRINT
R6:     MOV  DL，'6'
        JMP  PRINT
R7:     MOV  DL，'7'
        JMP  PRINT
R8:     MOV  DL，'8'
        JMP  PRINT
R9:     MOV  DL，'9'
        JMP  PRINT
PRINT:  MOV  AH，02H
        INT  21H
EXIT:   MOV  AH，4CH
        INT  21H
CODE    ENDS
        END  START
```

3）程序调试：调试步骤类似实验一。

4）实验记录：记录程序调试过程中出现的问题及解决过程，并加以简要说明。

1.4 实验三　循环程序设计与调试

1.4.1 循环程序设计基础知识

为了实现程序的循环，8086/8088 中提供了专门用于构成循环结构的重复控制指令。另外，还提供了串操作指令，并可借助于重复前缀，由硬件配合实现基本数据串指令的重复操作。掌握好上述指令的用法，是进行循环结构程序设计的基础。

1．循环程序的组成

一个循环结构的程序主要由以下 3 个部分组成。

（1）设置循环初值部分

循环程序的初始状态，需要在进入循环以前给出，一般称为设置循环初值。循环初值又

可分为循环工作部分的初值和循环结束条件的初值。

（2）循环工作部分

这是循环结构的基本部分。循环的工作部分又称为循环体，使用循环程序的目的就是要重复执行这段操作。不同的程序要解决的问题不同，因此工作部分的具体内容也有所不同。循环体一般包括重复操作的程序段和循环参数的修改，有些循环体还包含对循环控制参数的修改。循环程序的工作部分可以是顺序结构、分支结构，也可以是一个循环结构。当工作部分只有顺序结构或分支结构时，这样的循环结构称为单重循环结构；若工作部分又是一个新的循环程序，则该循环结构就称为多重循环结构。循环程序中最简单、最常用的是二重循环。在二重循环中，外面一层循环称外层循环，里面一层循环称内层循环。注意，在二重循环中内层循环可以有多个，但它们之间只能并列，不能再嵌套。如果二重循环中的内层循环又含有一重循环，则称为三重循环，依此类推，称为多重循环。

（3）循环控制部分

循环控制部分用于控制重复执行的次数，一般是检测循环结束条件。当循环结束条件不满足时，则继续重复执行循环体；当循环结束条件满足时，则退出循环执行循环结构外的后继语句。循环结构分为允许零次循环和不允许零次循环两种情况，允许零次循环结构是“先判断，后执行”，即先进入循环控制部分，判断循环的条件是否满足，所以有可能循环体一次也不执行；不允许零次循环则相反。

2．循环程序的设计步骤

设计循环程序一般分为以下几个步骤。

1）分析问题，确定选取几重循环结构，以及循环结束控制方法。

2）根据循环变化的规律，确定循环工作部分。

3）考虑循环参数的修改部分，分析并确定参数每次修改的方法。

4）设置循环控制部分及循环参数，可根据循环程序控制方法的不同，设置循环参数的初值。

3．循环控制方法

循环程序设计的核心问题是循环控制，常用的循环控制方法有计数控制法、条件控制法、开关控制法和逻辑尺控制法。

（1）计数控制法

对于循环体重复执行次数已知的情况，可以采用计数方法来控制循环。计数控制法又分为正计数法和倒计数法。

1）正计数法。所谓正计数法是将计数器的初值设置为 0，每执行一遍循环体，计数器加 1，然后与规定循环次数比较，若相等，则结束循环；否则，继续循环。这种方法使用的指令较多，需要用比较指令进行比较后再判断，不太方便，故一般不常用。

2）倒计数法。所谓倒计数法是先将计数器的初值设置为循环次数，每执行一遍循环体，计数器减 1，并测试是否为 0，若为 0，则结束循环；否则，继续循环。最常用于倒计数控制法的指令是 LOOP。

（2）条件控制法

在循环程序中，某些问题的循环次数预先不能确定，只能根据循环过程中某个特定条件是否满足来决定循环是否继续执行。对于这类问题，可以通过测试该条件来实现对循环的控