

National Computer Rank Examination

全国计算机等级考试



上机考试习题集

三级

汇编语言程序设计

教育部考试中心 编

供 2002 年下半年和 2003 年全年使用

赠送
参考答案
光盘

南开大学出版社

全国计算机等级考试

上机考试习题集

三级汇编语言程序设计

教育部考试中心 编

南开大学出版社

天 津

图书在版编目(CIP)数据

全国计算机等级考试上机考试习题集. 三级. 汇编语言程序设计/教育部考试中心编. —2版. —天津:南开大学出版社, 2002. 7

ISBN 7-310-01630-0

I. 全... II. 教... III. ①电子计算机—水平考试—习题②汇编语言—程序设计—水平考试—习题
IV. TP3—44

中国版本图书馆 CIP 数据核字(2002)第 037297 号

出版发行 南开大学出版社

地址:天津市南开区卫津路 94 号

邮编:300071 电话:(022)23508542

出版人 肖占鹏

承 印 河北昌黎人民胶印厂印刷

经 销 全国各地新华书店

版 次 2002 年 7 月第 2 版

印 次 2002 年 7 月第 2 次印刷

开 本 787mm×1092mm 1/16

印 张 12.25

字 数 306 千字

印 数 25001 — 30000

定 价 23.00 元

大力推行全国计算机等级考试 为发展知识经济、信息产业和培养计算机专门人才作出贡献 (代序)

**中国科学院院士 北京大学信息与工程科学学部主任
全国计算机等级考试委员会主任委员 杨芙清**

当今，人类正在步入一个以智力资源的占有和配置，知识生产、分配和使用为最重要因素的知识经济时代，也就是小平同志提出的“科学技术是第一生产力”的时代。科教是经济发展的基础，知识是人类创新的源泉。基础研究的科学发现、应用研究的原理探索和开发研究的技术发明，三者之间的联系愈来愈紧密，转换周期日趋缩短。世界各国的竞争已成为以经济为基础、以科技（特别是高科技）为先导的综合国力的竞争。

在高科技中，信息科学技术是知识高度密集、学科高度综合、具有科学与技术融合特征的学科。它直接渗透到经济、文化和社会的各个领域，迅速改变着人们的观念、生活和社会的结构，是当代发展知识经济的支柱之一。

在信息科学技术中，微电子是基础，计算机硬件及通信设施是载体，计算机软件是核心。软件是人类知识的固化，是知识经济的基本表征，软件已成为信息时代的新型“物理设施”。人类抽象的经验、知识正逐步由软件予以精确地体现。在信息时代，软件是信息化的核心，国民经济和国防建设、社会发展、人民生活都离不开软件，软件无处不在。软件产业是增长最快的朝阳产业，是具有高额附加值、高投入高产出、无污染、低能耗的绿色产业。软件产业的发展将推动知识经济的进程，促进从注重量的增长向注重质的提高方向发展，是典型的知识型产业。软件产业是关系到国家经济安全和文化安全，体现国家综合实力，决定 21 世纪国际竞争地位的战略产业。

为了适应知识经济发展的需要，大力推动信息产业的发展，需要在全民中普及计算机的基本知识，广开渠道，培养和造就一批又一批能熟练运用计算机和软件技术的各行各业的专门人才。

1994 年，原国家教委（现教育部）推出了全国计算机等级考试，它是一种重视应试人员对计算机和软件的实际掌握能力的考试。它不限制报考人员的学历背景，任何年龄段的人员都可以报考。这就为培养各行各业计算机的应用人才，开辟了一条广阔的道路。

1994 年是推出计算机等级考试的第一年，当年参加考试的有 1 万余人；而 2001 年，报考人数已达 172 万余人。截止至 2001 年底，全国计算机等级考试共开考 14 次，考生人数累计达 590 万人，其中，有 214 万人获得了各级计算机等级证书。

事实说明,鼓励社会各阶层的人士通过各种途径掌握计算机应用技术,并运用等级考试对他们的才干予以认真的、有权威性的认证,是一种较好的人才培养的有效途径,是比较符合我国具体情况的。等级考试也为用人单位录用和考核人员提供了一种测评手段。从有关公司对等级考试所作的社会抽样调查结果看,不论是管理人员还是应试人员,对该项考试的内容和形式都给予了充分肯定的评价。

计算机等级考试所取得的良好效果,也同全国各有关单位专家们在等级考试的大纲编写、试题设计、阅卷评分及效果分析等等多项工作中所付出的大量心血和辛勤的劳动密切相关,他们为这项工作的顺利开展作出了重要的贡献。

计算机与软件技术是一项日新月异的高新技术。计算机等级考试大纲有必要根据计算机与软件技术在近年的新发展,进行适当的修正,从而使等级考试更能反映当前计算机与软件技术的应用实际,使培养计算机应用人才的基础工作更健康地向前发展。计算机等级考试的形式也有必要根据形势的发展进行改革和完善,本书的出版,是对该考试的上机考试形式进行改革的一次积极尝试,有利于提高培训水平,有利于维护考试公平、公正。

从面临知识经济的机遇与挑战这样一个社会大环境的背景出发,考察全国计算机等级考试,就会看到,这一举措是符合知识经济和发展信息产业的方向的,是值得大力推行的。

我们相信,在 21 世纪知识经济和加快发展信息产业的形式下,在教育部考试中心的精心组织领导下,在全国各有关专家们的大力配合下,全国计算机等级考试一定会以更新的面貌出现,从而为我国培养计算机应用专门人才的宏大事业作出更多的贡献。

2002 年 4 月

前 言

在知识经济时代，知识在国家经济建设和社会发展的过程中，发挥着越来越重要的作用。而信息技术在知识经济的发展中，已经成为不可或缺的关键性因素。现在，计算机这个现代科技的结晶，正成为信息技术大众化的先进工具，它的应用在推动经济发展和社会进步方面正发挥着越来越大的作用。

根据我国经济发展的需要，为顺应市场经济体制的建立，促进人们对计算机的使用，九年前国家教委（现教育部）推出了全国计算机等级考试。至 2001 年底，这项考试的考生人数累计已达 590 多万，其中 214 多万人获得了证书。实践证明，这个同工作岗位培训密切相关的社会证书考试，促进了不同群体人们对计算机应用技术的学习和掌握，并且为用人单位提供了权威性的水平认证，也同样受到了广大考生的欢迎，受到了用人单位的广泛认可。

计算机等级考试需要考察考生的实际操作能力，实践性很强。因此，经全国计算机等级考试委员会专家的论证，以及教育部考试中心有关方面的研究，决定编写出版《全国计算机等级考试上机考试习题集》，供考生考前学习使用。这套习题集包括了一级、二级 QBASIC、二级 C、二级 FoxBASE、三级汇编语言、三级 C 共六个上机考试科目。该习题集的编写、出版和发行，对先行考试的考生同后续考试的考生交流上机考题的不正当现象是一种积极疏导，对某些人在上机考试期间利用网络进行试题炒作是一种化解，因而有利于保证这项考试的公平、公正，真实地考核出考生的计算机操作水平和编程能力。

编写这样一套习题集，是参照同类考试的做法，其内容同实际考试内容相近，所以这实际上也是对上机考试形式改革的一次尝试。许多专家参加了本书的具体编写工作，或者给予其他形式的支持，特别是，著名计算机专家杨芙清院士在百忙中为本书写了序言，给予充分肯定和支持。对于专家们的热情支持和辛勤工作，我谨代表教育部考试中心表示衷心的感谢，对为本书编写、出版和发行作出努力的其他有关同志也一并致谢。

教育部考试中心主任 赵亮宏

2002 年 4 月

1. 请编制程序, 其功能是: 将内存中由 SOURCE 指示的 40 个字节有符号数组成的数组分成正数和负数两个数组, 并求这两个数组的数据个数, 结果存放在 RESULT 指示的内存区域。存放形式为正数个数在前, 其后跟正数数组元素, 然后是负数个数及负数数组元素。

例如:

内存中有 1EH, 91H, 74H, 91H, 42H, 30H, 81H, F3H, 18H, 25H

结果为 06H, 1EH, 74H, 42H, 30H, 18H, 25H, 04H, 91H, 91H, 81H, F3H

部分程序已经给出, 其中原始数据由过程 LOAD 从文件 INPUT1.DAT 中读入 SOURCE 开始的内存单元中, 转换结果要求从 RESULT 开始存放, 由过程 SAVE 保存到文件 OUTPUT1.DAT 中。

填空 BEGIN 和 END 之间已经给出的一段源程序使其完整, 填空处已经用横线标出, 每个空白一般只需要填一条指令或指令的一部分(指令助记符或操作数), 也可以填入功能相当的多条指令, 或删除 BEGIN 和 END 之间原有的代码并自行编程来完成要求的功能。

对程序必须进行汇编, 并与 IO.OBJ 链接产生执行文件, 最终运行程序产生结果。调试中若发现整个程序中存在错误之处, 请加以修改。

试题程序:

```

                EXTRN     LOAD:FAR, SAVE:FAR
N               EQU       40
STAC            SEGMENT   STACK
                DB        128 DUP (?)
STAC            ENDS
DATA            SEGMENT
SOURCE          DB        N DUP(0)
RESULT          DB        N+2 DUP(0)
NAME0           DB        'INPUT1.DAT', 0
NAME1           DB        'OUTPUT1.DAT', 0
NDATA          DB        N DUP(0)
PDATA          DB        N DUP(0)
DATA            ENDS
CODE            SEGMENT
                ASSUME     CS:CODE, DS:DATA, SS:STAC
START           PROC      FAR
                PUSH       DS
                XOR        AX, AX
                PUSH       AX
                MOV        AX, DATA
                MOV        DS, AX
                MOV        ES, AX                ; 置附加段寄存器

                LEA        DX, SOURCE            ; 数据区起始地址
```

```

        LEA        SI, NAME0          ; 原始数据文件名
        MOV        CX, N              ; 字节数
        CALL       LOAD               ; 从 'INPUT1.DAT'中读取数据
; **** BEGIN ****
        LEA        SI, SOURCE
        (1)        DI, OFFSET PDATA   ;PDATA 为正数数组存放缓冲区首址
        MOV        BX, OFFSET NDATA   ;NDATA 为负数数组存放缓冲区首址
        XOR        DX, DX
        MOV        CX, N
        CLD
MAIN1:    LODSB
        TEST       AL, (2)
        JZ         MAIN2
        INC        DH                  ; -
        MOV        [BX], AL
        INC        BX
        (3)
MAIN2:    INC        DL
        MOV        [DI], AL           ; +
        INC        DI
MAIN3:    (4)        MAIN1
        LEA        SI, PDATA
        LEA        DI, RESULT
        MOV        [DI], DL
        INC        DI
        XOR        CX, CX
        MOV        CL, DL
MAIN4:    MOV        AL, (5)
        MOV        [DI], AL
        INC        DI
        INC        SI
        LOOP       (6)
        MOV        [DI], DH
        INC        DI
        XOR        CX, CX
        MOV        CL, DH
        MOV        BX, OFFSET NDATA
MAIN5:    MOV        AL, [BX]
        MOV        [DI], AL

```

```

                INC        DI
                _____
                (7)
LOOP            MAIN5
; **** END ****
                LEA        DX, RESULT    ; 结果数据区首址
                LEA        SI, NAME1     ; 结果文件名起始地址
                MOV        CX, N+2       ; 字节数
                CALL       SAVE          ; 保存结果到'OUTPUT1. DAT'文件中
                RET
START          ENDP
CODE           ENDS
                END        START

```

★★

2. 请编制程序, 其功能是: 内存中连续存放着 10 个无符号 8 位格雷码表示的数, 现将此十个数转换成十个 8 位二进制数, 结果存入内存。其转换方法为二进制数的最高位 d_7 与格雷码的最高位 g_7 相同, 二进制数的其余七位 d_k ($k=6, \dots, 0$) 分别为格雷码的位 g_k ($k=6, \dots, 0$) 与二进制数的位 d_{k+1} ($k=6, \dots, 0$) 异或的结果。

例如:

内存中有 00H, 03H, 2BH, 67H, 0CH, 15H, 54H, 02H, D8H, C7H

结果为 00H, 02H, 32H, 45H, 08H, 19H, 67H, 03H, 90H, 85H

部分程序已给出, 其中原始数据由过程 LOAD 从文件 INPUT1. DAT 中读入 SOURCE 开始的内存单元中。运算结果要求从 RESULT 开始存放, 由过程 SAVE 保存到文件 OUTPUT1. DAT 中。

填空 BEGIN 和 END 之间已给出的源程序使其完整(空白已用横线标出, 每行空白一般只需一条指令, 但采用功能相当的多条指令亦可), 或删除 BEGIN 和 END 之间原有的代码并自行编程来完成要求的功能。

对程序必须进行汇编, 并与 IO. OBJ 链接产生执行文件, 最终运行程序产生结果。调试中若发现整个程序中存在错误之处, 请加以修改。

试题程序:

```

                EXTRN      LOAD:FAR, SAVE:FAR
N              EQU        10
STAC          SEGMENT    STACK
                DB        128 DUP (?)
STAC          ENDS
DATA          SEGMENT
SOURCE        DB          N DUP (?)      ; 顺序存放 10 个字节数
RESULT        DB          N DUP (0)      ; 存放结果
NAME0         DB          'INPUT1. DAT', 0
NAME1         DB          'OUTPUT1. DAT', 0

```

```

DATA      ENDS
CODE      SEGMENT
          ASSUME      CS:CODE, DS:DATA, SS:STAC
START     PROC        FAR
          PUSH        DS
          XOR          AX, AX
          PUSH        AX
          MOV          AX, DATA
          MOV          DS, AX
          LEA          DX, SOURCE      ; 数据区起始地址
          LEA          SI, NAME0      ; 原始数据文件名
          MOV          CX, N          ; 字节数
          CALL         LOAD          ; 从 'INPUT1.DAT'中读取数据
;  **** BEGIN ****
          LEA          DI, RESULT
          LEA          SI, SOURCE
          MOV          CX, 10
AGNO:     MOV          AL, [SI]
          _____
                (1)
          MOV          CX, 8
          MOV          BX, 0
AGN1:     MOV          AH, 0
          SHL          BL, 1
          _____
                (2)
          RCL          AH, 1
          CMP          AH, _____
                (3)
          _____
                (4)
          JMP          NEXT
SET_ONE:  OR           BL, 01H
NEXT:     MOV          _____, BL
                (5)
          _____
                (6)
          LOOP         AGN1
          _____
                (7)
          MOV          [DI], BL
          INC          SI
          INC          DI
          LOOP         AGNO
;  **** END  ****
          LEA          DX, RESULT      ; 结果数据区首址

```

```

        LEA      SI, NAME1          ; 结果文件名
        MOV      CX, N              ; 结果字节数
        CALL     SAVE              ; 保存结果到文件
        RET
START   ENDP
CODE    ENDS
        END      START

```

★★

3. 请编制程序，其功能是：内存中连续存放着 20 个十六位二进制无符号数序列，请将它们排成升序(从小到大)。

例如：

内存中有 7001H, 7004H, 7002H... (假设后 17 个字均大于 7004H)

结果为 7001H, 7002H, 7004H... (后跟 17 个字，按从小到大的顺序排列)

部分程序已给出，其中原始数据由过程 LOAD 从文件 INPUT1.DAT 中读入 SOURCE 开始的内存单元中。运算结果要求从 RESULT 开始存放，由过程 SAVE 保存到文件 OUTPUT1.DAT 中。填空 BEGIN 和 END 之间已给出的源程序使其完整(空白已用横线标出，每行空白一般只需一条指令，但采用功能相当的多条指令亦可)，或删除 BEGIN 和 END 之间原有的代码并自行编程来完成要求的功能。

对程序必须进行汇编，并与 IO.OBJ 链接产生执行文件，最终运行程序产生结果。调试中若发现整个程序中存在错误之处，请加以修改。

试题程序：

```

        EXTRN    LOAD:FAR, SAVE:FAR
N       EQU      20
STAC    SEGMENT STACK
        DB       128 DUP (?)
STAC    ENDS
DATA    SEGMENT
SOURCE  DW       N DUP (?)
RESULT  DW       N DUP (0)
NAME0   DB       'INPUT1.DAT', 0
NAME1   DB       'OUTPUT1.DAT', 0
DATA    ENDS
CODE    SEGMENT
        ASSUME   CS:CODE, DS:DATA, SS:STAC
START   PROC     FAR
        PUSH     DS
        XOR      AX, AX
        PUSH     AX

```

```

        MOV     AX, DATA
        MOV     DS, AX
        LEA     DX, SOURCE      ; 数据区起始地址
        LEA     SI, NAME0      ; 原始数据文件名
        MOV     CX, N*2        ; 字数
        CALL    LOAD           ; 从 'DAT'中读取数据
; ***** BEGIN *****
        LEA     SI, SOURCE
        LEA     DI, RESULT
        MOV     CX, N
NEXT0:   MOV     AX, [SI]
        MOV     [DI], AX
        ADD     SI, (1)
        (2)
        LOOP    (3)
        CLD
        MOV     BX, N-1
MAL1:   LEA     SI, RESULT
        MOV     CX, (4)
NEXT:   LOD (5)
        CMP     [SI], AX
        JAE     CONT
        XCHG    [SI], (6)
        MOV     [SI-2], AX
CONT:   LOOP    (7)
        (8)
        (9)      MAL1
; ***** END *****
        LEA     DX, RESULT      ; 结果数据区首址
        LEA     SI, NAME1      ; 结果文件名
        MOV     CX, N*2        ; 结果字数
        CALL    SAVE           ; 保存结果到文件
        RET
START   ENDP
CODE    ENDS
        END      START

```

★★

4. 请编制程序，其功能是：内存中连续存放着 24 个无符号二进制字序列，字的最高 3 位为 000，此序列对应某一信号在一段时间内的连续变化，现对第 21 个二进制字前的 20 个二

进制字进行移动平均处理,其方法为:将要处理的字 X_i 用以它为开始的连续五个字的平均数 $(X_i+X_{i-1}+X_{i+2}+X_{i+3}+X_{i+4})/5$ 代替(余数舍去),得到新的 20 个无符号二进制字序列,结果存入内存。

例如:

内存中有 0100H, 0200H, 0300H, 0400H, 0500H, 0600H...

结果 0300H, 0400H...

部分程序已经给出,其中原始数据由过程 LOAD 从文件 INPUT1.DAT 中读入 SOURCE 开始的内存单元中,转换结果要求从 RESULT 开始存放,由过程 SAVE 保存到文件 OUTPUT1.DAT 中。

填空 BEGIN 和 END 之间已经给出的一段源程序使其完整,填空处已经用横线标出,每个空白一般只需要填一条指令或指令的一部分(指令助记符或操作数),也可以填入功能相当的多条指令,或删去 BEGIN 和 END 之间原有的代码并自行编程来完成要求的功能。

对程序必须进行汇编,并与 IO.OBJ 链接产生执行文件,最终运行程序产生结果。调试中若发现整个程序中存在错误之处,请加以修改。

试题程序:

```

        EXTRN      LOAD:FAR, SAVE:FAR
N        EQU        24
STAC     SEGMENT   STACK
        DB         128 DUP (?)
STAC     ENDS
DATA     SEGMENT
SOURCE   DW         N DUP (?)          ; 顺序存放 24 个字
RESULT   DW         20 DUP (0)         ; 存放结果
NAME0    DB         'INPUT1.DAT', 0
NAME1    DB         'OUTPUT1.DAT', 0
DATA     ENDS
CODE     SEGMENT
        ASSUME     CS:CODE, DS:DATA, SS:STAC
START    PROC
        PUSH       DS
        XOR        AX, AX
        PUSH       AX
        MOV        AX, DATA
        MOV        DS, AX
        LEA        DX, SOURCE          ; 数据区起始地址
        LEA        SI, NAME0           ; 原始数据文件名
        MOV        CX, N*2             ; 字节数
        CALL       LOAD                ; 从 'INPUT1.DAT'中读取数据
;  **** BEGIN ****
```

```

        MOV     DI, 0
        MOV     SI, 0
        MOV     CX, 20
        MOV     BX, (1)
AGNO:   MOV     AX, SOURCE[SI]
        PUSH    SI
        PUSH    CX
        MOV     (2), (3)
AGN1:   INC     SI
        INC     SI
        ADD     AX, SOURCE[SI]
        LOOP    AGN1

        (4)
        DIV     BX
        MOV     RESULT[DI], AX
        INC     DI
        (5)
        POP     CX
        POP     SI
        INC     SI
        (6)
        LOOP    AGNO
; **** END ****
        LEA     DX, RESULT      ; 结果数据区首址
        LEA     SI, NAME1      ; 结果文件名
        MOV     CX, 40         ; 结果字节数
        CALL    SAVE           ; 保存结果到文件
        RET
START   ENDP
CODE    ENDS
        END     START

```

★★

5. 请编制程序，其功能是：内存中从 SOURCE 开始连续存放着 21 个八位有符号数(补码)，其相邻两数之间差值不超过-8 至 7。对这种变化缓慢的数据可采用差分方法进行压缩。即第一个数据不变, 其后的数据取与前一数据的差值并用四位二进制补码表示，两个差值拼成一个字节，前一个差值放在高四位, 后一个差值放在低四位。

例如：

原数据 (X[n])： 23H, 27H, 2AH, 29H, 22H...

压缩后 (Y[n])： 23H, 43H, F9H...

编程按上述方法进行压缩，结果保存在 RESULT 开始的内存单元中。

部分程序已给出，请填空 BEGIN 和 END 之间已给出的源程序使其完整(空白已用横线标出，每行空白一般只需一条指令，但功能相当的多条指令亦可)或删除 BEGIN 和 END 之间原有的代码并自行编写程序片段来完成要求的功能。

原始数据由过程 LOAD 从文件 INPUT1.DAT 中读入 SOURCE 开始的内存单元中，结果要求从 RESULT 开始存放，由过程 SAVE 保存到文件 OUTPUT1.DAT 中。

对程序必须进行汇编，并与 IO.OBJ 链接产生执行文件，最终运行程序产生结果。调试中若发现整个程序中存在错误之处，请加以修改。

试题程序：

```
        EXTRN        LOAD:FAR, SAVE:FAR

N        EQU          10

STAC      SEGMENT      STACK
          DB           128 DUP (?)
STAC      ENDS

DATA      SEGMENT
SOURCE    DB           2*N+1    DUP (?)
RESULT    DB           N+1      DUP (0)
NAME0     DB           'INPUT1.DAT', 0
NAME1     DB           'OUTPUT1.DAT', 0
DATA      ENDS

CODE      SEGMENT
          ASSUME        CS:CODE, DS:DATA, SS:STAC
START     PROC          FAR
          PUSH          DS
          XOR            AX, AX
          PUSH          AX
          MOV            AX, DATA
          MOV            DS, AX
          MOV            ES, AX          ; 置附加段寄存器

          LEA            DX, SOURCE      ; 数据区起始地址
          LEA            SI, NAME0       ; 原始数据文件名起始地址
          MOV            CX, 2*N+1       ; 字节数
          CALL            LOAD           ; 从'INPUT1.DAT'中读取数据
```

```

; **** BEGIN ****

        LEA     SI, SOURCE
        LEA     DI, RESULT
        CLD
        MOVSB                     ; Y[0]=X[0]
        XOR     BX, BX             ; FLAG=0
        MOV     DX, N*2           ; COUNTER

COMPRESS:
        LODSB
        SUB               (1)           ; X[n]-X[n-1]
                  (2)           ; FLAG=NOT FLAG
        J   (3)   LOW_HEX
        MOV     CL, 4
          (4)   AL, CL
        MOV     AH, AL
        JMP     NEXT

LOW_HEX:
                  (5)          
        OR      AL, AH
        STOSB

NEXT:              (6)          
        JNE     COMPRESS

; **** END ****

        LEA     DX, RESULT      ; 结果数据区首址
        LEA     SI, NAME1       ; 结果文件名起始地址
        MOV     CX, N+1         ; 字节数
        CALL    SAVE            ; 保存结果到'OUTPUT1.DAT'文件中
        RET

START    ENDP
CODE     ENDS
        END      START

```

★★

6. 请编制程序，其功能是：内存中存放着 20 个数字 0~9 之间的 ASCII 字符(包括数字 0 和 9 的 ASCII 字符)或 SP 字符(20H)，请将数字 0~9 之间的 ASCII 字符（包括数字 0 和 9 的 ASCII 字符）转换为相应的八位二进制数，并将 SP 字符转换为\$字符(24H)。将按上述方法处理后得到的 20 个字节存入内存中。

例如：

内存中有 20H('SP'), 30H('O'), 31H('1'), 31H('1'), 31H('1'), 20H('SP'), 32H('2'), ..., 39H('9') (共 20 个 ASCII 字符)

结果为 24H('\$'), 00H, 01H, 01H, 01H, 24H('\$'), 02H('2'), ..., 09H('9') (共 20 个字节)

部分程序已给出, 其中原始数据由过程 LOAD 从文件 INPUT1.DAT 中读入 SOURCE 开始的内存单元中。运算结果要求从 RESULT 开始存放, 由过程 SAVE 保存到文件 OUTPUT1.DAT 中。

填空 BEGIN 和 END 之间已给出的源程序使其完整 (空白已用横线标出, 每行空白一般只需一条指令, 但采用功能相当的多条指令亦可), 或删除 BEGIN 和 END 之间原有的代码并自行编程来完成要求的功能。

对程序必须进行汇编, 并与 IO.OBJ 链接产生执行文件, 最终运行程序产生结果。调试中若发现整个程序中存在错误之处, 请加以修改。

试题程序:

```
                EXTRN      LOAD:FAR, SAVE:FAR
N                EQU       20
STAC             SEGMENT   STACK
                DB         128 DUP (?)
STAC            ENDS
DATA            SEGMENT
SOURCE          DB         N DUP (?)
RESULT          DB         N DUP (0)
NAME0           DB         'INPUT1.DAT', 0
NAME1           DB         'OUTPUT1.DAT', 0
DATA            ENDS
CODE            SEGMENT
                ASSUME     CS:CODE, DS:DATA, SS:STAC
START           PROC       FAR
                PUSH       DS
                XOR        AX, AX
                PUSH       AX
                MOV        AX, DATA
                MOV        DS, AX
                LEA        DX, SOURCE          ; 数据区起始地址
                LEA        SI, NAME0          ; 原始数据文件名
                MOV        CX, N              ; 字节数
                CALL       LOAD               ; 从 'INPUT1.DAT'中读取数据
; ***** BEGIN *****
                _____ (1) _____
MOV            DI, 0
                _____ (2) _____
```