**计算机科学与工程学院实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验课程名称** | | **汇编语言程序设计** | | | **实验总成绩** |  |
| **专业** | **计算机科学与技术** | | **班级** | **计算机1803** | **指导教师签字** |  |
| **学号** | **20184484** | | **姓名** | **胥卜凡** | **实验报告批改时间** |  |
| **实验报告分项成绩**   |  |  |  | | --- | --- | --- | | **序号** | **实验项目** | **成绩** | | **1** | **分支程序设计实验** |  | | **2** |  |  | | **3** |  |  | | **4** |  |  | | **5** |  |  | | | | | | | |
| **实验课程总结**  从以下方面总结：1.实验体现知识应用和初步研究能力；2. 反映基本观察、发现问题和分析问题能力；3. 实验项目内容或者实验课程是否存在问题及下一年度改进意见）4.其他方面  总体来看，本次实验极大地丰富了我的动手实践能力，让我对E指令进一步掌握，对于三个数的正负判断和对于分支结构的设计也有了更深的理解。  具体来说，为完成本次实验，我首先观察了题目，发现需要对3个数进行判断，这显然是分支程序的设计，需要用JMP等操作进行分支的实现。其次，对于数的正负判断有许多方法，鉴于代码的简单性和简洁性，我采用依次判断的方法。最后，实验要求使用e命令进行修改，我则充分运用了之前所学的内容，进行了修改，并成功输出结果。  综上所述，本次实验对我而言是有效的，是有益的，增加了我的观察问题、发现问题、分析问题的能力。 | | | | | | |

**实验一**

**（一）实验内容：**

**题目：**

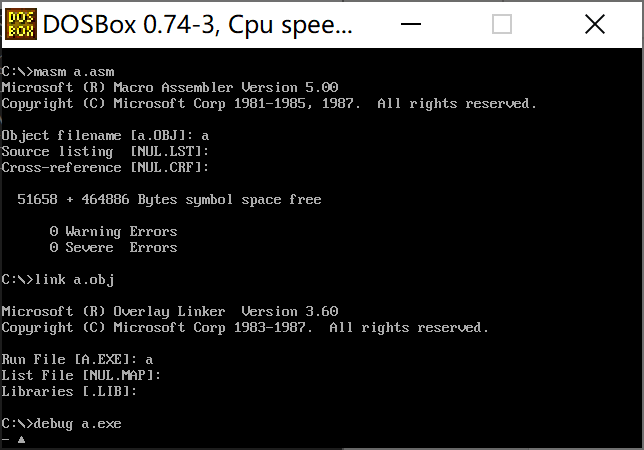
已知DATA单元开始存放三个带符号数，编制程序，检查三个数据是否同号，若同时为正，则在显示器上显示“+”；同时为负，则显示“-”；否则显示空格“ ”。

**要求：**

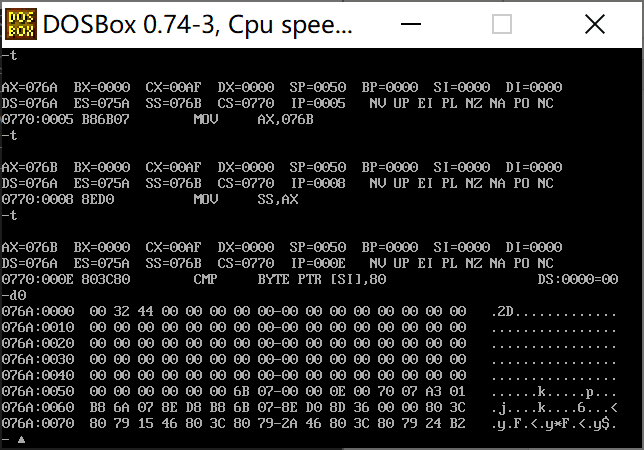
分别用三组数据（同时为正，同时为负，异号）调试程序，验证程序的正确性。第一组数据在源程序中给出，第二组和第三组数据要求在DEBUG下用E命令给出。

**（二）实验步骤：**

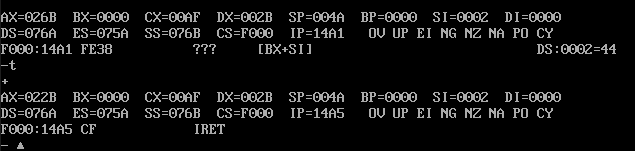
①编写程序，并进行汇编、连接，进入调试模式。。



②首先查看第一组数据（00H,32H,44H）

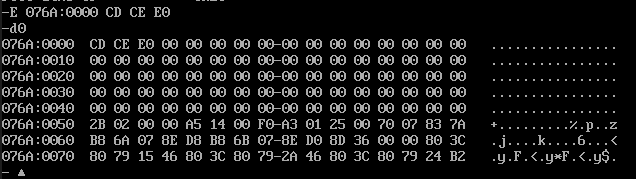


③执行到输出处



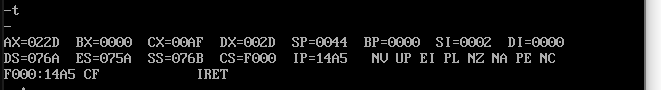
成功输出‘+’

④用E命令更改内存数据为（CD,CE,E0）



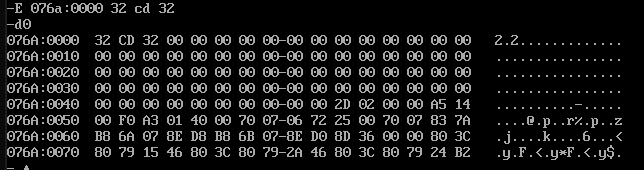
⑤从头开始重新执行到目标处



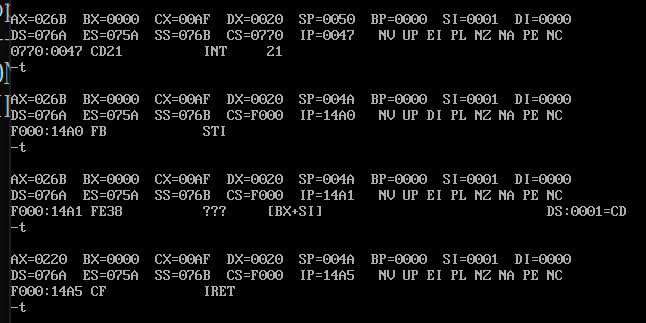


成功输出‘-’

⑥用E命令更改内存数据为（32,CD,32）



⑦从头开始执行到目标处



空格显示的不明显，但注意到AH=02，所以确实输出了空格

**（三）遇到问题及思考：**

①如何同时修改多个数据的内存

答：E 地址 数据 数据 数据

②如何判断一个带符号数是正（包括0）还是负的？

答：与80H进行CMP操作，SF=0则是负数，否则就是正数

③如何判断3个数的情况?

答：先判断前两个数的情况，再判断第三个数的情况，类似前缀和的求法。

**附录：**

1. **源代码:**

**实验三:**

DSEG SEGMENT

    DATA DB  33H,32H,44H;正数33H,32H,44H；负数CDH,CEH,BCH；

DSEG ENDS

SSEG SEGMENT STACK

    STK DB 80 DUP(0);

SSEG ENDS

CSEG SEGMENT

    ASSUME DS:DSEG,SS:SSEG,CS:CSEG;

START:

    MOV AX,DSEG;

    MOV DS,AX;

    MOV AX,SSEG;

    MOV SS,AX;

    LEA SI,DATA;

    CMP BYTE PTR [SI],80H;

    JNS FU1;

ZHENG1:

    INC SI;

    CMP BYTE PTR [SI],80H;

    JNS  ELS;

ZHENG2:

    INC SI;

    CMP BYTE PTR [SI],80H;

    JNS ELS;

ZHENG3:

    MOV DL,'+';

    MOV AH,2

    INT 21H

    JMP RETURN

FU1:

    INC SI;

    CMP BYTE PTR[SI],80H;

    JNS FU2;

    JMP ELS;

FU2:

    INC SI;

    CMP BYTE PTR[SI],80H;

    JNS FU3;

    JMP ELS;

FU3:

    MOV DL,'-';

    MOV AH,2

    INT 21H

    JMP RETURN;

ELS:

    MOV DL,' ';

    MOV AH,2

    INT 21H

RETURN:

    MOV AH,4CH;

    INT 21H;

CSEG ENDS

    END START