**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** | **代码转换实验** |  | | **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，让我自己编写了一个能够简单实现计算器功能的汇编程序。  具体来说，为完成本次实验，我首先观察了题目，发现需要进行计算器的简单实现，这里需要用到代码转换相关的设计，且既需要用LOOP、JZ等操作进行循环、分支的实现，也需要使用ADD、IMUL等操作进行相关运算。其次，对于子程序的设计，要按标准进行编写使用PROC、ENDP等关键字。  综上所述，本次实验极大地增加了我的观察问题、发现问题、分析问题的能力。 | | | | | | |

**实验六**

**（一）实验内容：**

**题目：**

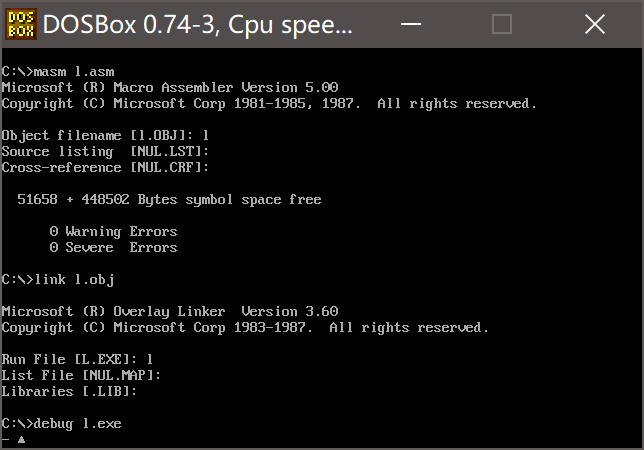
编写一个能实现“\*/+-”功能的程序

**要求：**

原始数据从键盘输入，结果在显示器上显示。

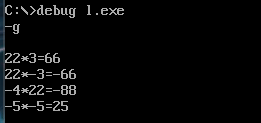
**（二）实验步骤：**

①编写程序，并进行汇编、连接，进入调试模式。。



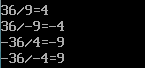
②下面对每个计算符进行**正~正，正~负，负~正，负~负**运算

③执行乘法功能



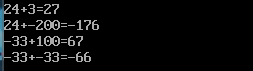
可以看到对应输出的结果都是正确的

④执行除法功能



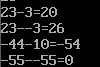
可以看到对应输出的结果都是正确的

⑤执行加法功能



可以看到对应输出的结果都是正确的

⑥执行减法功能



可以看到对应输出的结果都是正确的

⑦退出程序



**（三）遇到问题及思考：**

①程序比较大，不方便调试

答：设置断点，在数据段专门设计调试数据方便调试过程中进行查看

②如何让负数也参与运算

答：一种是采用IMUL等进行运算，最后判断目标数是否小于0，小于0则取补前面加负号；此外对除法要对两个数分别进行判断。

③出现jump out of range…的错误

答：条件跳转的目标指令在内存中的地址，与该条件转移指令之间的距离，不能超过 －128 ~+127的范围。

④数据输出错误

答：忘记重置对数判断的标志区域了

⑤负数情况没考虑全

答：一开始的流程想的不对，画一个流程图之后重新架构程序就考虑的十分周全了

**附录：**

1. **源代码:**

**实验六:**

DATA SEGMENT

DIVISORS DW 10000, 1000, 100, 10, 1

RESULTS DB 0,0,0,0,0,"$" ;存放五位数ASCII码

COUNT DB 0;

NUM1 DW 99H;

NUM2 DW 88H

COUNT2 DB 0;

data ends

SSEG SEGMENT STACK

    STK DB 200 DUP(0)

SSEG ENDS

CODE SEGMENT

ASSUME CS:CODE,SS:SSEG;

START:

    MOV AX,DATA

    MOV DS,AX

    MOV AX,SSEG;

    MOV SS,AX;

JISUAN:                    ;判断加减乘除

    MOV COUNT,0

CLEAR:                  ;换行+初始化

    MOV DL,0DH

    MOV AH,2

    INT 21H

    MOV DL,0AH

    MOV AH,2

    INT 21H

    MOV AX,0

    MOV BX,0

MAI:                    ;判断进行计算的类别

    CALL INPUT

    MOV NUM1,BX

    PUSH BX

    CMP AL,'+'

    JZ PLUS             ;加

    CMP AL,'-'

    JZ MINUS            ;减,需要判断是中间符号还是数字

    CMP AL,'\*'

    JZ BY               ;乘

    CMP AL,'/'

    JZ DIVD             ;除

    CMP AL,'Q'

    JZ MEXIT1            ;按Q退出

    CMP AL,'q'

    JZ MEXIT1

PLUS:

    CALL INPUT          ;读取第二个数

    MOV NUM2,BX

    POP AX              ;取出第一个数

    ADD AX,BX           ;两数相加

    CMP AX,0

    JL  FUP

    JMP NEXT            ;进行输出并读取下一个数

MINUS:

    CALL INPUT

    POP AX              ;取出第一个数

    CMP AX,BX           ;由于减法需要考虑负数，故先比较

    JL FU               ;负数跳转

    SUB AX,BX

    JMP NEXT

FU:

    SUB AX,BX           ;减法

    NEG AX              ;取补(负)

    PUSH AX             ;结果入栈

    MOV DL,'-'          ;显示负号

    MOV AH,02H

    INT 21H

    POP AX              ;结果出栈

    JMP NEXT            ;进行输出并读取下一个数

MEXIT1:

    JMP MEXIT

BY:

    CALL INPUT          ;读取第二个数

    POP AX              ;执行乘法

    IMUL BX

    CMP AX,0

    JL FUP

    JMP NEXT            ;进行输出并读取下一个数

DIVD:

    CALL INPUT          ;读取第二个数

    POP AX

    CMP AX,0

    JL  FU1

    JMP ZHENG1

FU1:                    ;第一个数是负数

    NEG AX

    CMP BX,0

    JL  FU2

    JMP FUZHENG

FU2:                ;两个数都是负数

    NEG BX          ;标记第二个数是负数

    MOV COUNT,0     ;都是负数不用加负号

    JMP CHU

ZHENG1:      ;第一个数是正数

    CMP BX,0

    JL ZHENGFU      ;一正一负加负号

    JMP CHU         ;都是正数不用加负号

ZHENGFU:

    NEG BX

    MOV COUNT,2

    JMP CHU

FUZHENG:

    MOV COUNT,2

    JMP CHU

CHU:

    DIV BX

    CMP COUNT,2

    JZ  FUP2

    JMP NEXT

FUP:

    NEG AX

    PUSH AX             ;结果入栈

    MOV DL,'-'          ;显示负号

    MOV AH,02H

    INT 21H

    POP AX              ;结果出栈

    JMP NEXT

FUP2:

    PUSH AX             ;结果入栈

    MOV DL,'-'          ;显示负号

    MOV AH,02H

    INT 21H

    POP AX              ;结果出栈

NEXT:

    CALL OUTPUT         ;输出

    JMP  JISUAN            ;进行下一轮计算

MEXIT:

    POP BX              ;出栈

    MOV AH,4CH          ;退出程序

    INT 21H

;输入功能

INPUT PROC NEAR

SST:

    MOV BX,0

NUM:

    MOV AH,1

    INT 21H

                         ;遇到符号进行第二个数的读取

    CMP AL,'+'

    JZ EXIT

    CMP AL,'-'           ;负号要进行情况判断

    JZ FLAG

    CMP AL,'\*'

    JZ EXIT

    CMP AL,'/'

    JZ EXIT

    CMP AL,'Q'           ;输入有Q直接跳转结束程序

    JZ EXIT

    CMP AL,'q'

    JZ EXIT

    SUB AL,30H

    JL EXIT

    CMP AL,9

    JG EXIT

    CBW

    XCHG AX,BX           ;模拟进位

    MOV CX,10

    MUL CX

    XCHG AX,BX

    ADD BX,AX

    MOV COUNT2,1

    JMP NUM

FLAG:

    INC COUNT

    CMP COUNT,2         ;-xx-的情况

    JZ EXIT2

    CMP COUNT2,1        ;xx-的情况

    JZ RETURN1

    JMP SST

EXIT:

    CMP COUNT,1         ;-xx\*/+的情况

    JZ EXIT2

RETURN1:

    MOV COUNT,0

    MOV COUNT2,0

    RET

EXIT2:

    MOV COUNT,0

    MOV COUNT2,0

    NEG BX

    RET

INPUT ENDP

OUTPUT PROC NEAR        ;输出

    MOV SI, OFFSET DIVISORS

    MOV DI, OFFSET RESULTS

    MOV CX,5

JINWEI:

    MOV DX,0

    DIV word ptr [si]

    ADD al,30H

    MOV byte ptr [di],al

    INC di

    ADD si,2

    MOV ax,dx

    LOOP JINWEI

    MOV cx,4

    MOV di, OFFSET RESULTS

BI0:

    CMP BYTE PTR [di],'0'

    JNE PRINT

    INC DI

    LOOP BI0

PRINT:

    MOV DX,DI

    MOV AH,9

    INT 21H

    RET

OUTPUT ENDP

CODE ENDS

    END START