

华中科技大学

课程实验报告

课程名称： 汇编语言程序设计实验

实验名称： 实验一 编程基础

实验时间： 2017-3-15, 14: 30-17: 30 实验地点： 南一楼 804 室 099 号实验台

指导教师： 张勇

专业班级： 计算机科学与技术 201307 班

学 号： U201314969 姓 名： 王镇宇

同组学生： 无 报告日期： 2017 年 3 月 22 日

原创性声明

本人郑重声明：本报告的内容由本人独立完成，有关观点、方法、数据和文献等的引用已经在文中指出。除文中已经注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品或成果，不存在剽窃、抄袭行为。

特此声明！

学生签名：

日期：2017.3.22

成绩评定

实验完成质量得分（70 分） （实验步骤清晰详细深入，实验记录真实完整等）	报告撰写质量得分（30 分） （报告规范、完整、通顺、详实等）	总成绩（100 分）

指导教师签字：

日期：

目录

1	实验目的与要求	2
2	实验内容	2
3	实验过程	5
3.1	任务 1	5
3.1.1	设计思想及寄存器分配	5
3.1.2	流程图	5
3.1.3	源程序	5
3.1.4	实验步骤	5
3.1.5	实验记录	5
3.2	任务 2	6
3.2.1	设计思想及存储单元分配	6
3.2.2	流程图	7
3.2.3	源程序	7
3.2.4	实验步骤	7
3.2.5	实验记录与分析	7
3.3	任务 3	8
3.3.1	设计思想及存储单元分配	8
3.3.2	流程图	8
3.3.3	源程序	8
3.3.4	实验步骤	9
3.3.5	实验记录	10
3.4	任务 4	10
3.4.1	设计思想及存储单元分配	10
3.4.2	流程图	11
3.4.3	源程序	11
3.4.4	实验步骤	15
3.4.5	实验记录	15
4	总结与体会	16
	参考文献	17

1 实验目的与要求

本次实验的主要目的与要求有下面 6 点，所有的任务都会围绕这 6 点进行，希望大家事后检查自己是否达到这些目的与要求。

- (1) 掌握汇编源程序编辑工具、汇编程序、连接程序、调试工具 TD 的使用；
- (2) 理解数、符号、寻址方式等在计算机内的表现形式；
- (3) 理解指令执行与标志位改变之间的关系；
- (4) 熟悉常用的 DOS 功能调用；
- (5) 熟悉分支、循环程序的结构及控制方法，掌握分支、循环程序的调试方法；
- (6) 加深对转移指令及一些常用的汇编指令的理解。

2 实验内容

（上机实验环境说明：在机房计算机上建议大家使用 VMWare Workstation 的虚拟机环境，在自己的计算机上可以使用 DOSBox 虚拟机。源程序编辑工具可以使用记事本、EDIT、或 C 语言的编辑器；汇编程序使用 MASM 6.0；连接程序使用 LINK；调试工具使用 TD；具体介绍可参见教材第 7 章及《80X86 汇编语言程序设计上机指南》，相关软件可以到教学网站上下载）

任务 1. 《80X86 汇编语言程序设计》教材中 P31 的 1.14 题。

要求：(1) 直接在 TD 中输入指令，完成两个数的求和、求差的功能。求和/差后的结果放在 (AH) 中。

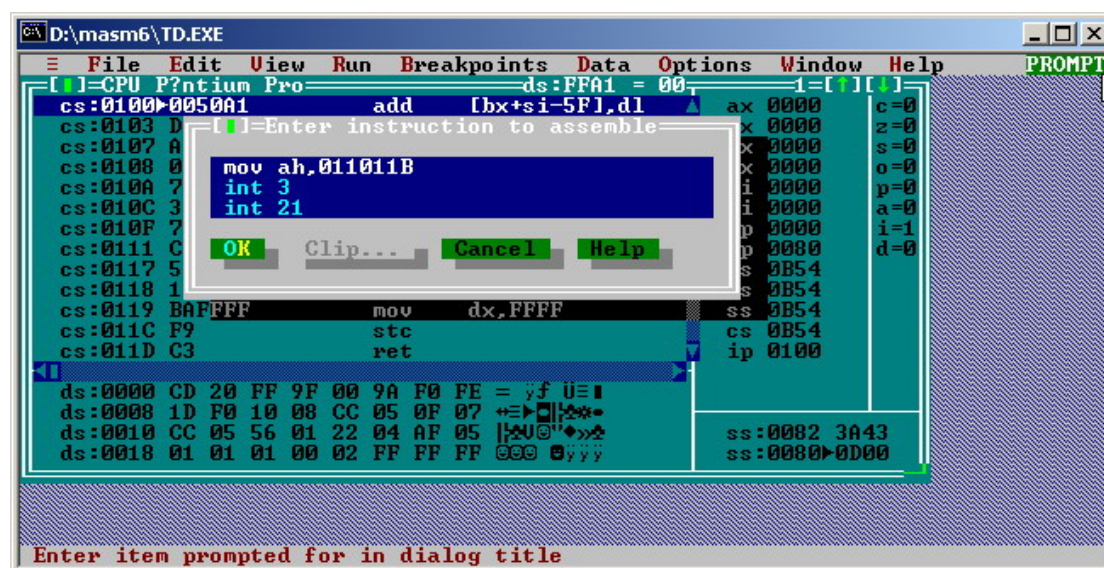
(2) 请事先指出执行指令后 (AH)、标志位 SF、OF、CF、ZF 的内容。

(3) 记录上机执行后的结果，与 (2) 中对应的内容比较。

(4) 求差运算中，若将 A、B 视为有符号数，且 $A > B$ ，标志位有何特点？

若将 A、B 视为无符号数，且 $A > B$ ，标志位又有何特点？

在 TD 中输入指令语句的操作提示：将 TD 中的代码显示区置为当前区域，光标移到期望修改的行后，直接输入汇编指令；当输入了第一个字符时，TD 自动弹出如下图所示的指令编辑窗口。每输入完一条指令，按回车键，这时输入的指令即可出现在光标处，同时光标自动下移一行，以便输入下一条指令。



任务 2. 《80X86 汇编语言程序设计》教材中 P45 的 2.3 题。

要求：（1）分别记录执行到“MOV CX, 10”和“INT 21H”之前的(BX), (BP), (SI), (DI)各是多少。

（2）记录程序执行到退出之前数据段开始 40 个字节的内容，指出程序运行结果是否与设想的一致。

（3）在标号 LOPA 前加上一段程序，实现新的功能：先显示提示信息“Press any key to begin!”，然后，在按了一个键之后继续执行 LOPA 处的程序。

操作提示：使用 TD.EXE 调试程序时，应先单步执行各个语句，每执行一条语句，都应观察数据段中的内容以及相应寄存器的变化。首先注意观察对 DS 寄存器的赋值过程，并在 TD 的数据窗口定位待观察的数据区位置。其次，单步执行循环体两遍且正确理解了循环体语句的含义后，可在“MOV AH, 4CH”处设置断点，然后直接执行到断点处，回答(1)和(2)的问题。

任务 3. 《80X86 汇编语言程序设计》教材中 P45 的 2.4 题的改写。

要求：（1）实现的功能不变，对数据段中变量访问时所用到的寻址方式中的寄存器改成 32 位寄存器。

（2）内存单元中数据的访问采用变址寻址方式。

（3）记录程序执行到退出之前数据段开始 40 个字节的内容，检查程序运行结果是否与设想的一致。

（4）在 TD 代码窗口中观察并记录机器指令代码在内存中的存放形式，并与 TD 中提供的反汇编语句及自己编写的源程序语句进行对照，也与任务 2 做对比。（相似语句记录一条即可，重点理解机器码与汇编语句的对应关系，尤其注意操作数寻址方式的形式）。

（5）观察连续存放的二进制串在反汇编成汇编语言语句时，从不同字节位置开始反汇编，结果怎样？理解 IP/EIP 指明指令起始位置的重要性。

操作提示：要让 TD 从任意指定地址开始反汇编，需要使用 TD 在代码显示区的 Goto 功能，即鼠标选中代码显示区，点击右键将显示带有 Goto 的菜单，选中 Goto 菜单项，输入 CS:XXXX 即可（XXXX 是你希望录入的偏移地址）。

任务 4. 设计实现一个学生成绩查询的程序。

1、实验背景

在以 BUF 为首址的字节数据存储区中，存放着 n 个学生的课程成绩表（百分制），每个学生的相关信息包括：姓名（占 10 个字节，结束符为数值 0），语文成绩（1 个字节），数学成绩（1 个字节），英语成绩（1 个字节），平均成绩（1 个字节）。

例如：

N EQU 30

BUF DB 'zhangsan', 0, 0 ; 学生姓名，不足 10 个字节的部分用 0 填充

DB 100, 85, 80, ? ; 平均成绩还未计算

DB 'lisi', 6 DUP(0)

DB 80, 100, 70, ?

DB N-3 DUP('TempValue', 0, 80, 90, 95, ?) ; 除了 3 个已经具体定义了学生信息的成績表以外，其他学生的信息暂时假定为一样的。

DB 'wangwu', 0, 0, 0, 0 ; 最后一个必须是自己名字的拼音

DB 85, 85, 100, ?

2、功能一：提示并输入待查询成绩的学生姓名

(1) 使用 9 号 DOS 系统功能调用，提示用户输入学生姓名。

(2) 使用 10 号 DOS 系统功能调用，输入学生姓名。输入的姓名字符串放在以 in_name 为首址的存储区中。

(3) 若只是输入了回车，则回到“(1)”处重新提示与输入；若仅仅输入字符 q，则程序退出，否则，准备进入下一步处理。

3、功能二：以学生姓名查询有无该学生

(1) 使用循环程序结构，在成绩表中查找该学生。

(2) 若未找到，就提示用户该学生不存在，并回到“功能一(1)”的位置，提示并重新输入姓名。

(3) 若找到，则将该学生课程成绩表的起始偏移地址保存到 POIN 字变量中。

提示：字符串比较时，当采用输入串的长度作为循环次数时，若因循环次数减为 0 而终止循环，则还要去判断成绩表中名字串的下一个字符是否是结束符 0，若是，才能确定找到了（这样做是为了避免输入的名字仅仅是数据段中所定义名字的子集的误判情况）。

4、功能三：计算所有学生的平均成绩

使用算数运算相关指令计算并保存每一个学生的平均成绩。

平均成绩计算公式： $(A*2+B+C/2)/3.5$ ，即将语文成绩 A 乘以权重 2、英语成绩 C 除以权重 2 后，与数学成绩 B 一起求和，再计算该生的平均成绩。要求避免溢出。

提示：使用循环程序结构，注意寻址方式的灵活使用。把小数 3.5 转换成分数后再运算避免使用浮点数指令。

5、功能四：将功能二查到的学生的平均成绩进行等级判断，并显示判断结果。

(1) 平均成绩等级显示方式：若平均成绩大于等于 90 分，显示“A”；大于等于 80 分，显示“B”；大于等于 70 分，显示“C”；大于等于 60 分，显示“D”；小于 60 分，显示“F”。

提示：使用分支程序结构，采用 2 号 DOS 系统功能调用显示结果。

(2) 使用转移指令回到“功能一(1)”处（提示并输入姓名）。

3 实验过程

3.1 任务 1

3.1.1 设计思想及寄存器分配

寄存器分配：一个 AH, 运算后的结果存储在 AH 中。

3.1.2 流程图

无

3.1.3 源程序

无

3.1.4 实验步骤

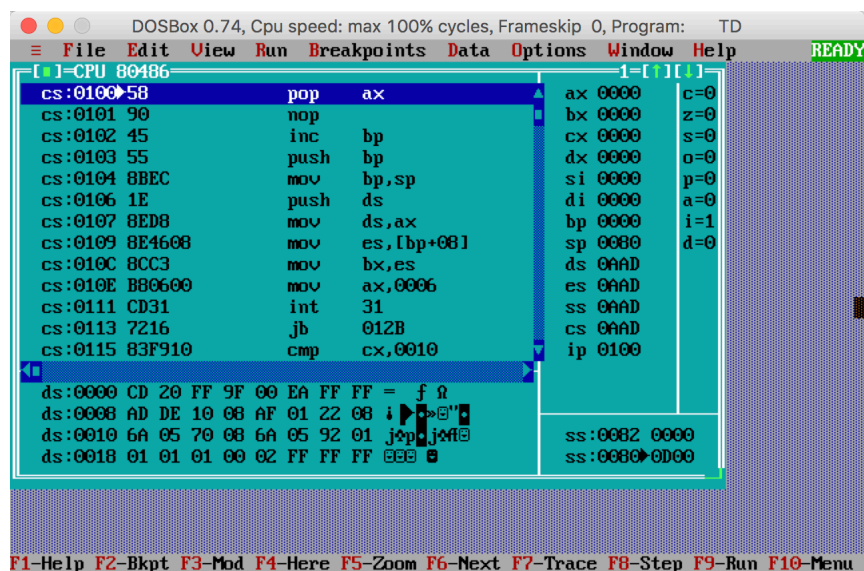
1. 使用编辑程序 EDIT.EXE 录入源程序，存盘文件名为 CUBE.ASM。
2. 使用 MASM5.0 或 6.0 汇编源文件。即 MASM CUBE;
3. 观察提示信息，若出错，则用编辑程序修改错误，存盘后重新汇编，直至不再报错为止。
4. 使用连接程序 LINK.EXE 将汇编生成的 CUBE.OBJ 文件连接成执行文件。
即 LINK CUBE;
5. 若连接时报错，则依照错误信息修改源程序。之后重新汇编和连接，直至不再报错并生成 CUBE.EXE 文件。
6. 执行该程序。即在命令行提示符后输入 CUBE 后回车，观察执行现象。

3.1.5 实验记录

- 1、实验环境条件：MacBook Pro：2.9 Ghz Intel Core i5，8G 内存；DosBox：EDIT.EXE 2.0；MASM.EXE 6.0；LINK.EXE 5.2；TD.EXE 5.0。
- 2、求和运算：

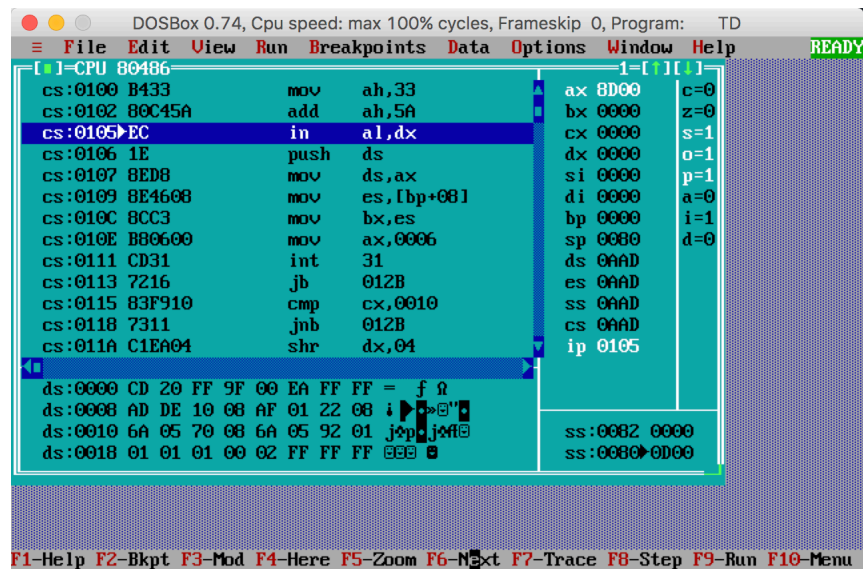
```
mov ah, 00110011b
add ah, 01011010b
```

执行前：



在 add ah, 01011010b 下一句设置一个断点运行至断点停止

执行后：



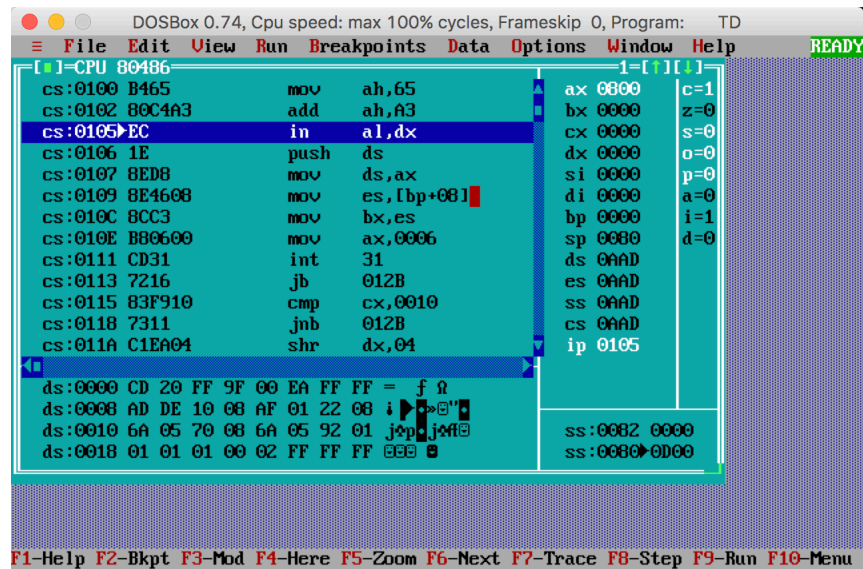
分析结果：右边标志寄存器区 o 变为 1，说明有溢出，c 为 0，说明没有进位

3、求差运算

mov ah, 01100101b

add ah, 10100011b

执行后：



分析结果：c 变为 1，说明产生进位

3.2 任务 2

3.2.1 设计思想及存储单元分配

用到 4 个数据段 BUF1-BUF4，一开始用 4 个寄存器 SI、DI、BX、BP 依次存储 BUF 第一个值，到后面用循环给 BUF1-BUF4 空间赋以不同的值。

3.2.2 流程图

无

3.2.3 源程序

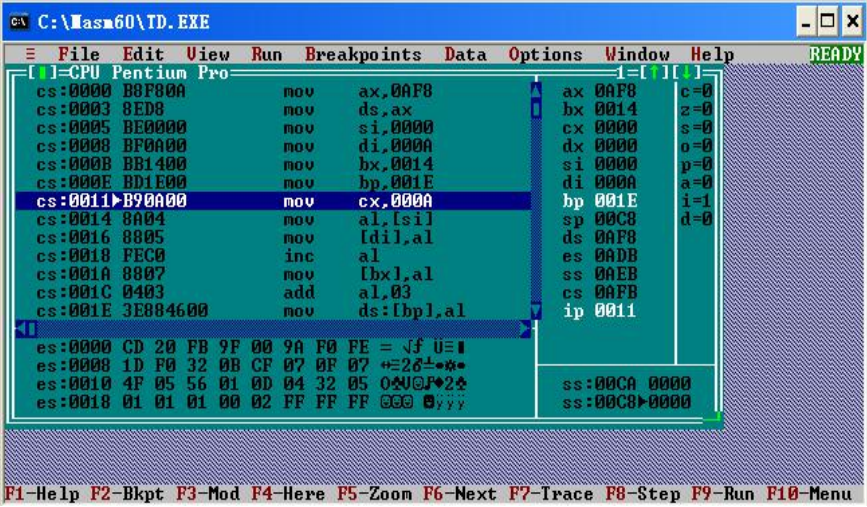
无

3.2.4 实验步骤

- 1. 使用编辑程序 EDIT.EXE 录入源程序，存盘文件名为 CUBE.ASM。
- 2. 使用 MASM5.0 或 6.0 汇编源文件。即 MASM CUBE；
- 3. 观察提示信息，若出错，则用编辑程序修改错误，存盘后重新汇编，直至不再报错为止。
- 4. 使用连接程序 LINK.EXE 将汇编生成的 CUBE.OBJ 文件连接成执行文件。
即 LINK CUBE；
- 5. 若连接时报错，则依照错误信息修改源程序。之后重新汇编和连接，直至不再报错并生成 CUBE.EXE 文件。
- 6. 执行该程序。即在命令行提示符后输入 CUBE 后回车，观察执行现象。

3.2.5 实验记录与分析

- 1、按 f8 进行单步调试，直到光标移到“MOV CX, 10”

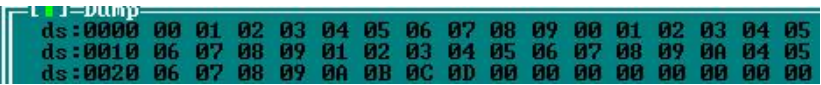


Bx 变为 0014，bp 变为 001E，si 为 0000，di 变为 000A

在 mov ah, 4c 一行按下 f2 设置断点，按下 f9，将运行至该行，再按下 f8，执行至 “INT 21H” 之前

Bx 变为 001E，bp 变为 0028，si 为 000A，di 变为 0014

- 2、记录程序执行到退出之前数据段开始 40 个字节的内容：



依次输出 0、1、2、3、4、5、6、7、8、9 0、1、
2、3、4、5、6、7、8、9 1、2、3、4、
5、6、7、8、9、10 4、5、6、7、8、9、
10、11、12、13

可见与预期结果相符。

3、在代码中改写：在数据段中加入

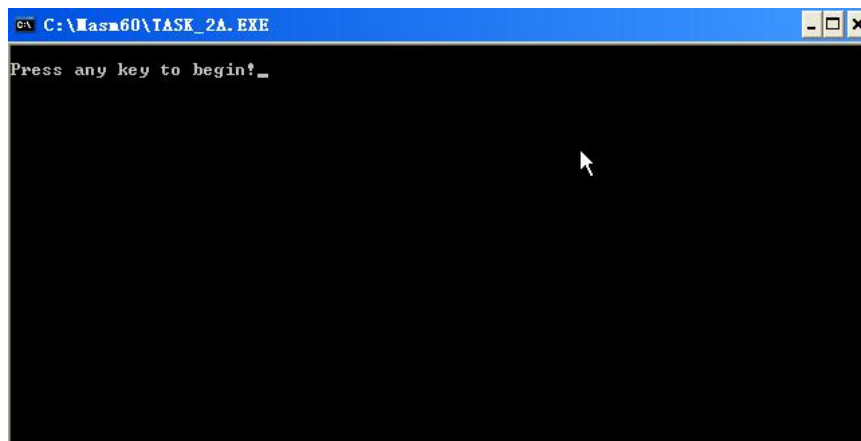
BUF5 DB 0AH,0DH,' Press any key to begin!\$'
再在 LOPA 前加上

```
LEA DX,BUF5  
MOV AH,9  
INT 21H
```

实现 9 号功能调用，在屏幕输出字符串在 LOPA 前加上

```
MOV AH,1  
INT 21H
```

实现 1 号功能调用，等待从键盘输入一个字符，一旦输入，继续程序执行，截图：



3.3 任务 3

3.3.1 设计思想及存储单元分配

AL 每循环一次+1，再赋给 BUF[ESI]相应的值，寄存器 CX 来控制循环次数。

3.3.2 流程图

无

3.3.3 源程序

```

. 386
STACK SEGMENT USE16 STACK
        DB 200 DUP(0)
STACK ENDS
DATA SEGMENT USE16
BUF1 DB 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
BUF2 DB 10 DUP(0)
BUF3 DB 10 DUP(0)
BUF4 DB 10 DUP(0)
DATA ENDS
CODE SEGMENT USE16
        ASSUME CS:CODE, DS:DATA, SS:STACK
START: MOV AX, DATA
        MOV DS, AX
        MOV ESI, 0
        MOV CX, 10
LOPA:  MOV AL, BUF1[ESI]
        MOV BUF2[ESI], AL
        INC AL
        MOV BUF3[ESI], AL
        ADD AL, 3
        MOV BUF4[ESI], AL
        INC ESI
        DEC CX
        JNZ LOPA
        MOV AH, 4CH
        INT 21H
CODE ENDS
        END START

```

3.3.4 实验步骤

1. 使用编辑程序 EDIT.EXE 录入源程序，存盘文件名为 CUBE.ASM。
2. 使用 MASM5.0 或 6.0 汇编源文件。即 MASM CUBE;
3. 观察提示信息，若出错，则用编辑程序修改错误，存盘后重新汇编，直至不再报错为止。

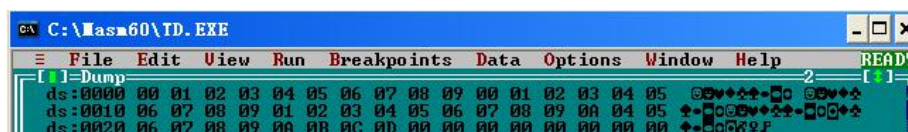
4. 使用连接程序 LINK.EXE 将汇编生成的 CUBE.OBJ 文件连接成执行文件。
即 LINK CUBE;
5. 若连接时报错，则依照错误信息修改源程序。之后重新汇编和连接，直至不再报错并生成

CUBE.EXE 文件。

6. 执行该程序。即在命令行提示符后输入 CUBE 后回车，观察执行现象。

3.3.5 实验记录

设断点在 int 21 那行，打开 dump 观察数据区的值如下：



对比任务 2 中结果可知符合。

3.4 任务 4

3.4.1 设计思想及存储单元分配

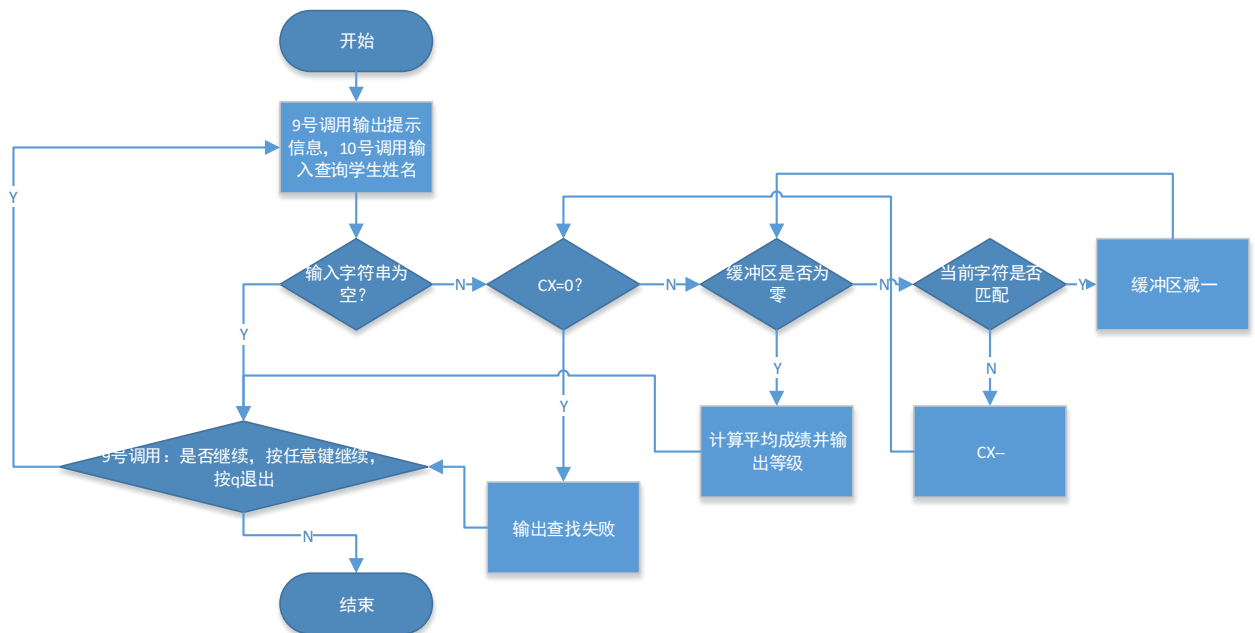
设计思想：

本实验主要是要解决 3 个问题：一是提示并输入学生姓名；二是如何进行字符串的比较从而查询学生；三是如何计算平均分从而分出等级。依据本次实验的要求，此处用分支、循环程序的结构及控制方法来解决。

寄存器分配：

- CX：存放学生数量（默认 3），计数器作用；
- BX：存放目标学生下标值，得到分数缓冲区首地址；
- BP:存放缓冲区基地址；
- AL:存放平均成绩；
- AX, DX, SI：临时寄存器；

3.4.2 流程图



3.4.3 源程序

```

. 386
STACK SEGMENT USE16 STACK
    DB 200 DUP(0)
STACK ENDS
DATA SEGMENT USE16
    N EQU 3
    BUF DB 'zhangsan', 0, 0
        DB 100, 85, 80, ?
        DB 'lisi', 6 DUP(0)
        DB 80, 100, 70, ?
        DB 'wangwu', 4 DUP(0)
        DB 85, 85, 100, ?
    MSG DB 'Input the name : $'
    CONTINU DB 'Enter any keys to continue(if enter q, exit!):$'
    FAIL DB 'Not Exist!$'
    INPUT DB 10
        DB ?
        DB 10 DUP(0)
    
```

```

DATA    ENDS

CODE    SEGMENT      USE16
        ASSUME  CS:CODE, DS:DATA, SS:STACK

START:  MOV     AX, DATA
        MOV     DS, AX
        JMP     BEGIN

FAILED:  LEA     DX, OFFSET FAIL      ; 查找失败
        MOV     AH, 9H
        INT     21H

LOOPA:   MOV     DL, 0AH              ;换行符
        MOV     AH, 2H
        INT     21H
        LEA     DX, OFFSET CONTINU   ;继续
        MOV     AH, 9H
        INT     21H
        MOV     DL, 0AH              ;换行符
        MOV     AH, 2H
        INT     21H
        MOV     DL, 0DH
        MOV     AH, 2H
        INT     21H
        MOV     AH, 8H
        INT     21H
        CMP     AL, 71H
        JE      OVER                 ;输入 'q', 退出程序

BEGIN:   MOV     CX, N                 ; 学生个数
        LEA     DX, OFFSET MSG        ;输出提示信息
        MOV     AH, 9H
        INT     21H
        MOV     DL, 0AH              ; 换行符
        MOV     AH, 2H
        INT     21H
        LEA     DX, OFFSET INPUT      ; 读入学生姓名
        MOV     AH, 0AH
        INT     21H

```

```

MOV     DL, 0AH           ; 换行符
MOV     AH, 2H
INT     21H
LEA     BP, OFFSET INPUT   ; 将 INPUT 基址存放至 BP
ADD     BP, 2
CMP     DS:BYTE PTR [BP-1], 0H ; 空字符串
JE      LOOPA
INC     CX
COMPA:  DEC     CX
        JE      FAILED      ; 查找失败, 重新输入
MOV     BX, N              ; 计算目标学生下标值, 存放至 BX
SUB     BX, CX
IMUL    BX, 14             ; 根据目标学生下标值, 找到分数缓冲区首地址
MOV     AX, 10             ; 临时计数器
MOV     SI, 0
COMPB:  MOV     DL, [BX + SI]
        MOV     DH, BYTE PTR DS:[BP + SI]
        CMP     DL, 0        ; 如果缓冲区姓名已结束, 说明查找成功
        JE      CAL         ; 跳转至平均成绩计算处
        CMP     DH, DL       ; 比较 当前缓冲区姓名 与 输入姓名 字符
        JNE     COMPA       ; 当前字符相同, 继续循环以比较下一字符
        INC     SI
        DEC     AX
        JNE     COMPB
CAL:    MOV     BX, N        ; 计算目标学生下标值, 存放至 BX
        SUB     BX, CX
        IMUL    BX, 14
        ADD     BX, 10       ; 根据目标学生下标值, 找到分数缓冲区首地址 BX =
m * 14 + 10
        MOV     AX, 0
        MOV     DX, 0
        MOV     AL, [BX]     ; 计算平均成绩
        ADD     AX, AX
        MOV     DL, [BX + 1]
        ADD     AX, DX

```



```

MOV     DL, [BX +2]
SAR     DL, 1
ADD     AX, DX      ;AL=A*2+B+C/2
SAL     AX, 1
MOV     DL, 7
IDIV    DL          ; AL = 2 * AL / 7
MOV     [BX + 3], AL; AVG = AL ( AL / 3.5)
CMP     AL, 90
JGE     LEVELA
CMP     AL, 80
JGE     LEVELB
CMP     AL, 70
JGE     LEVELC
CMP     AL, 60
JGE     LEVELD
JMP     LEVELF
LEVELA:MOV DL, 41H   ;输出成绩等级
        MOV     AH, 2H
        INT     21H
        JMP     LOOPA
LEVELB:MOV DL, 42H
        MOV     AH, 2H
        INT     21H
        JMP     LOOPA
LEVELC:MOV DL, 43H
        MOV     AH, 2H
        INT     21H
        JMP     LOOPA
LEVELD:MOV DL, 44H
        MOV     AH, 2H
        INT     21H
        JMP     LOOPA
LEVELF:MOV DL, 46H
        MOV     AH, 2H
        INT     21H

```

```

        JMP     LOOPA
OVER:    MOV     AH, 4CH
        INT     21H
CODE     ENDS
        END     START

```

3.4.4 实验步骤

1. 使用编辑程序 EDIT.EXE 录入源程序，存盘文件名为 shiyan.ASM。
2. 使用 MASM 6.0 汇编源文件。即 MASM shiyan ;
3. 观察提示信息，若出错，则用编辑程序修改错误，存盘后重新汇编，直至不再报错为止。
4. 使用连接程序 LINK.EXE 将汇编生成的 shiyan.OBJ 文件连接成执行文件。
即 LINK shiyan ;
5. 若连接时报错，则依照错误信息修改源程序。之后重新汇编和连接，直至不再报错并生成 shiyan.EXE 文件。
6. 执行该程序。即在命令行提示符后输入 shiyan 后回车，观察执行现象。
7. 让 9 号功能调用显示的信息放在自己希望的位置。
8. 在 9 号功能调用时，尝试带显示字符串的结尾没有 “\$” 结束符。
9. 10 号功能调用时，输入的字符数超过定义的数量时，它是如何处理的？

3.4.5 实验记录

- 1、实验环境条件 :P3 1GHz, 256M 内存 ;WINDOWS 2000 命令行窗口 ;EDIT.EXE 2.0 ;
MASM.EXE 6.0 ; LINK.EXE 5.2; TD.EXE 5.0。
- 2、按照流程图编写完程序，编译和连接都没问题，运行时出现

```

Write fault error writing device PRN
Abort, Retry, Ignore, Fail?_

```

后来发现是 9 号调用的字符串在之前定义时没有以\$结尾，改正后程序运行正确

- 3、程序运行正确，但格式很差，如下图：

```

Input the name : zhangsan$
A Enter any keys to continue(if enter q,exit!): Input the name : lisi$
B Enter any keys to continue(if enter q,exit!): Input the name : wanguu$
F Enter any keys to continue(if enter q,exit!): Input the name : lihua$
Not Exist! Enter any keys to continue(if enter q,exit!):
_

```

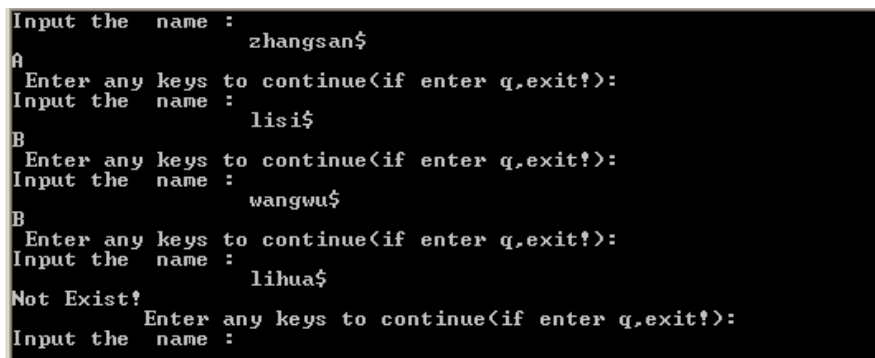
改进方法：在 LOOPA 里 1 号调用的换行符后面加上一个回车换行符，即

```
MOV     DL, 0AH           ;换行符
MOV     AH, 2H
INT     21H
```

后面加上：

```
MOV     DL, 0DH
MOV     AH, 2H
INT     21H
```

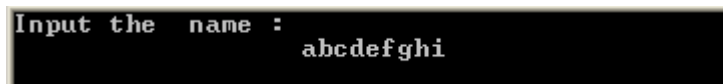
改正后运行如下图：



```
Input the name : zhangsan$
A
Enter any keys to continue(if enter q,exit!):
Input the name : lisi$
B
Enter any keys to continue(if enter q,exit!):
Input the name : wangwu$
B
Enter any keys to continue(if enter q,exit!):
Input the name : lihua$
Not Exist!
Enter any keys to continue(if enter q,exit!):
Input the name :
```

分析：存放的三个人平均成绩依次为 zhangsan :88 等级 A；lisi :83 等级 B；wangwu :90 等级 B，可见与输出结果相符合

4、对于输入字符超过定义的数量时，会出现无法再继续输入的情况如下图：



```
Input the name : abcdefghi
```

4 总结与体会

这是汇编的第一次上机，总体来说比较顺利，上机主要是熟悉编程和调试环境，学会如何将一个 .asm 文件编译成 .obj 文件，再将 .obj 文件链接成一个 .exe 文件，熟悉了 td 的环境及操作过程，对 td 的调试过程有一定的了解，主要是 go to 到指定的一行，和 f2 设置断点，以及单步执行，熟悉了 td 界面的各个区域及里面包含的内容，能够通过观察 td 里面的内容与代码联系起来分析。

通过实验 4 熟悉分支、循环程序的结构和控制方法，掌握了它们的调试方法，对转移指令有了更深的理解，是从比高级语言更底层的角度去理解了其含义，对其它的汇编指令有了初步的认识，相信在今后的几次实验中会对更多的指令更加的熟悉和掌握。

参考文献

- [1] 王元珍等. 80x86 汇编语言程序设计. 版本(第 1 版)
- [2] 王晓虹等. 汇编语言程序设计教程. 版本(第 1 版)