

## 分支程序设计

### 一、实验目的：

- 1.加深对分支程序的理解，掌握分支程序的结构。熟悉运算类指令对标志位的状态影响以及标志位状态的表示方法；掌握条件转移、无条件转移指令的使用方法。
- 2.掌握分支程序设计、编写、调试和运行的方法。

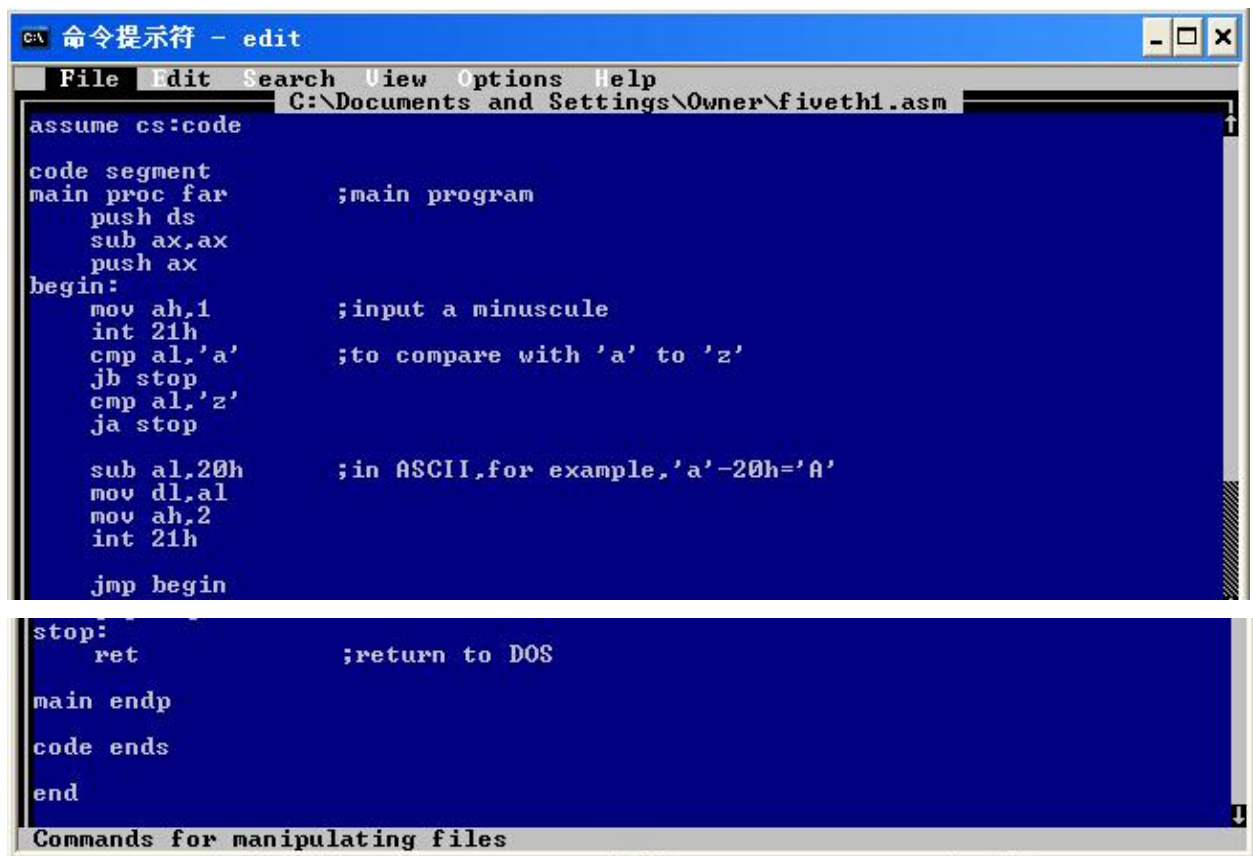
### 二、实验内容：

- 1.从试编写小程序：对键盘输入的小写字母用大写字母显示出来。
- 2.编制程序统计学生成绩。要求如下：设有 10 个学生的成绩分别为 56、69、84、82、73、88、99、63、100 和 80 分。试编制程序分别统计低于 60 分、60~69 分、70~79 分、80~89 分、90~99 分及 100 分的人数存放到 s5、s6、s7、s8、s9 及 s10 单元中。成绩分等部分采用分支结构，统计所有成绩则用循环结构完成。

### 三、实验主要步骤：

#### 1.程序一： 对键盘输入的小写字母用大写字母显示出来

##### (1) 源代码：



```
命令提示符 - edit
File Edit Search View Options Help
C:\Documents and Settings\Owner\fiveth1.asm

assume cs:code

code segment
main proc far           ;main program
    push ds
    sub ax,ax
    push ax
begin:
    mov ah,1             ;input a minuscule
    int 21h
    cmp al,'a'            ;to compare with 'a' to 'z'
    jb stop
    cmp al,'z'
    ja stop

    sub al,20h            ;in ASCII,for example,'a'-20h='A'
    mov dl,al
    mov ah,2
    int 21h

    jmp begin

stop:
    ret                  ;return to DOS

main endp
code ends
end

Commands for manipulating files
```

```

C:\DOCUME~1\Owner>masm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Source filename [ASM]: fiveth1
Object filename [fiveth1.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

49358 + 449215 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\DOCUME~1\Owner>link

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Object Modules [OBJ]: fiveth1
Run File [FIVETH1.EXE]:
List File [NUL.MAP]:
Libraries [LIB]:
LINK : warning L4021: no stack segment

```

## (2) 编译连接执行

按顺序输入 26 个小写字母，对应输出 26 个大写字母，结果正确。  
最后用“回车”作为结束符，程序结束，返回 dos

```

C:\DOCUME~1\Owner>fiveth1
aAbBcCdDeEfFgGhHiI jJkKlLmMnNoOpPqQrRsStTuUvVwWxXyYzZ

```

## (3) debug 调试

```

C:\DOCUME~1\Owner>debug fiveth1.exe
Microsoft (R) Symbolic Debug Utility Version 4.00
Copyright (C) Microsoft Corp 1984, 1985. All rights reserved.

Processor is [80286]
-r
AX=0000 BX=0000 CX=001B DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0000 NV UP EI PL NZ NA PO NC
0E80:0000 1E          PUSH    DS
-u
0E80:0001 2BC0          SUB     AX,AX
0E80:0003 50          PUSH    AX
0E80:0004 B401          MOV     AH,01
0E80:0006 CD21          INT     21
0E80:0008 3C61          CMP     AL,61                ;'a'
0E80:000A 720E          JB      001A
0E80:000C 3C7A          CMP     AL,7A                ;'z'
0E80:000E 770A          JA      001A
-u
0E80:0010 2C20          SUB     AL,20                ;' '
0E80:0012 8AD0          MOV     DL,AL
0E80:0014 B402          MOV     AH,02
0E80:0016 CD21          INT     21
0E80:0018 EBFA          JMP     0004
0E80:001A CB          RETF
0E80:001B 7989          JNS     FFA6
0E80:001D 798C          JNS     FFAB

```

反汇编

```

-g 0006
AX=0100 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0006 NU UP EI PL ZR NA PE NC
0E80:0006 CD21 INT 21 ;Read Keyboard and Echo
-t
aAX=0161 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0008 NU UP EI PL ZR NA PE NC
0E80:0008 3C61 CMP AL,61 ;'a'
-t
AX=0161 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=000A NU UP EI PL ZR NA PE NC
0E80:000A 720E JB 001A
-t
AX=0161 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=000C NU UP EI PL ZR NA PE NC
0E80:000C 3C7A CMP AL,7A ;'z'
-t
AX=0161 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=000E NU UP EI NG NZ AC PE CY
0E80:000E 770A JA 001A
-t
AX=0161 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0010 NU UP EI NG NZ AC PE CY
0E80:0010 2C20 SUB AL,20 ;' '
-t
AX=0161 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0010 NU UP EI NG NZ AC PE CY
0E80:0010 2C20 SUB AL,20 ;' '
-t
AX=0141 BX=0000 CX=001B DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0012 NU UP EI PL NZ NA PE NC
0E80:0012 8AD0 MOV DL,AL
-t
AX=0141 BX=0000 CX=001B DX=0041 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0014 NU UP EI PL NZ NA PE NC
0E80:0014 B402 MOV AH,02

```

以小写'a'为例子，'a'在'a'~'z'内，所以不跳转，执行 sub al,20h 此时 al 内容对应的 ASCII 就是'A'

```

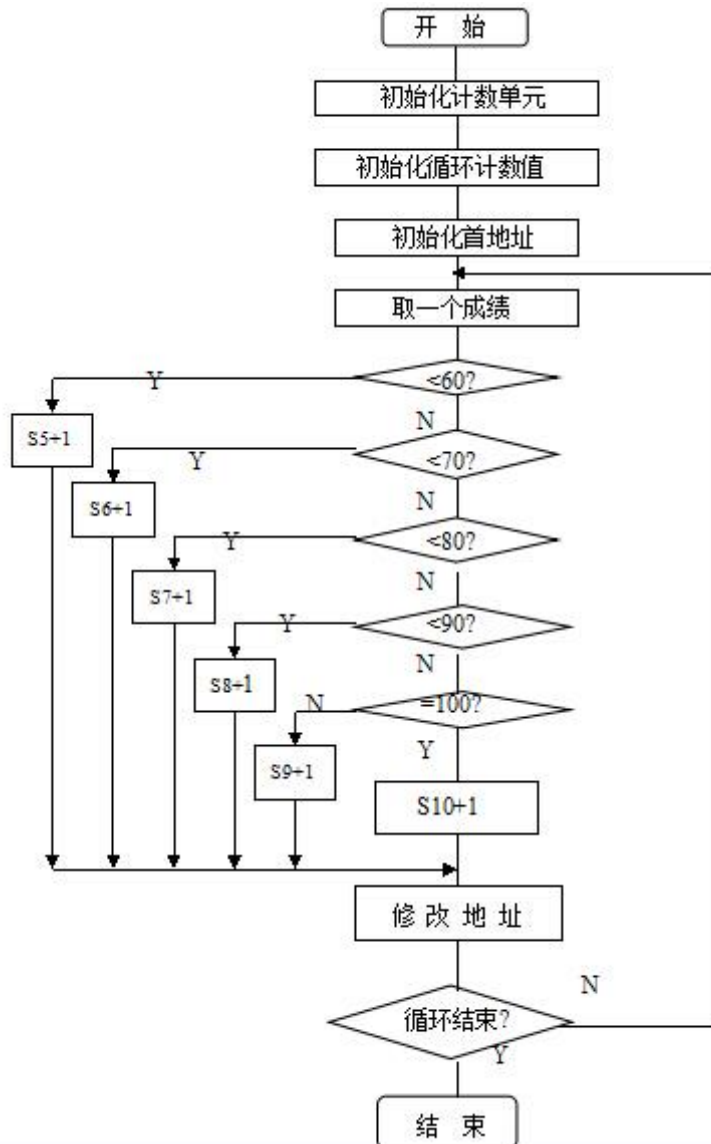
-t
AX=0241 BX=0000 CX=001B DX=0041 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0004 NU UP EI PL NZ NA PE NC
0E80:0004 B401 MOV AH,01
-t
AX=0141 BX=0000 CX=001B DX=0041 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0006 NU UP EI PL NZ NA PE NC
0E80:0006 CD21 INT 21 ;Read Keyboard and Echo
-t
DAX=0144 BX=0000 CX=001B DX=0041 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=0008 NU UP EI PL NZ NA PE NC
0E80:0008 3C61 CMP AL,61 ;'a'
-t
AX=0144 BX=0000 CX=001B DX=0041 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=000A NU UP EI NG NZ NA PO CY
0E80:000A 720E JB 001A
-t
AX=0144 BX=0000 CX=001B DX=0041 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E80 IP=001A NU UP EI NG NZ NA PO CY
0E80:001A CB RETF
-t
AX=0144 BX=0000 CX=001B DX=0041 SP=0000 BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0E70 IP=0000 NU UP EI NG NZ NA PO CY
0E70:0000 CD20 INT 20

```

大写'D'不在'a'~'z'内，所以跳转，结束程序。其他不在'a'~'z'范围内字符同理。

## 2.程序二： 编制程序统计学生成绩

### (1) 源代码和流程图



```

C:\ 命令提示符 - edit
File Edit Search View Options Help
C:\Documents and Settings\Owner\fiveh2.asm
dataarea segment ;define data segment
grade dw 56,69,84,82,73,88,99,63,100,80
s5 dw 0
s6 dw 0
s7 dw 0
s8 dw 0
s9 dw 0
s10 dw 0
dataarea ends

progran segment ;define code segment
main proc far ;main part of program
assume cs:progran, ds:dataarea
start: ;starting execution address
; set up stack for return
push ds ;save old data segment
sub ax,ax ;put zero in AX
push ax ;save it on stack
; set DS register to current data segment
mov ax,dataarea ;dataarea segment addr
mov ds,ax ;into DS register
  
```

```

; MAIN PART OF PROGRAM GOES HERE
    mov cx,10
    mov bx,0
s:    call compare
    add bx,2
    loop s

    ret                                ;return to DOS
main endp                             ;end of main part of program

compare proc near
    cmp word ptr[bx],60                ;defined grade is a word
                                        ;so we should explain the [bx] we need is a w
    jb bujige
    cmp word ptr[bx],70
    jb jige
    cmp word ptr[bx],80
    jb lianghao
    cmp word ptr[bx],90
    jb youxiu
    cmp word ptr[bx],100
    jb youyi
    inc si                               ;=100
    ret
bujige: inc si                          ;<60
        ret
jige:   inc si                          ;<70
        ret
lianghao: inc si                        ;<80
        ret
youxiu: inc si                          ;<90
        ret
youyi:  inc si                          ;<100
        ret
compare endp

prognam ends

    end start                          ;end assembly

```

Commands for manipulating files

## (2) 编译连接运行

```

C:\DOCUME~1\Owner>masm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Source filename [LASM1]: fiveth2
Object filename [fiveth2.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

49206 + 447319 Bytes symbol space free

0 Warning Errors
0 Severe Errors
C:\DOCUME~1\Owner>link

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Object Modules [L.OBJ]: fiveth2
Run File [FIVETH2.EXE]:
List File [NUL.MAP]:
Libraries [L.LIB]:
LINK : warning L4021: no stack segment

C:\DOCUME~1\Owner>fiveth2

```

正常执行并返回 dos

```

C:\DOCUME~1\Owner>debug fiveth2.exe
Microsoft (R) Symbolic Debug Utility Version 4.00
Copyright (C) Microsoft Corp 1984, 1985. All rights reserved.

Processor is I80286I
-r
AX=0000 BX=0000 CX=006F DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0E70 ES=0E70 SS=0E80 CS=0EB2 IP=0000  NU UP EI PL NZ NA PO NC
0E82:0000 1E          PUSH    DS
-u
0E82:0001 2BC0          SUB     AX,AX
0E82:0003 50          PUSH    AX
0E82:0004 B8800E      MOV     AX,0E80
0E82:0007 8ED8      MOV     DS,AX
0E82:0009 B90A00      MOV     CX,000A
0E82:000C BB0000      MOV     BX,0000
0E82:000F E80600      CALL    0018
0E82:0012 83C302      ADD     BX,+02
-u
0E82:0015 E2F8      LOOP    000F
0E82:0017 CB          RETF
0E82:0018 833F3C      CMP     Word Ptr [BX],+3C
0E82:001B 7219      JB      0036
0E82:001D 833F46      CMP     Word Ptr [BX],+46
0E82:0020 7219      JB      003B
0E82:0022 833F50      CMP     Word Ptr [BX],+50
0E82:0025 7219      JB      0040
-u
0E82:0027 833F5A      CMP     Word Ptr [BX],+5A
0E82:002A 7219      JB      0045
0E82:002C 833F64      CMP     Word Ptr [BX],+64
0E82:002F 7219      JB      004A
0E82:0031 FF061E00    INC     Word Ptr [001E]
0E82:0035 C3          RET
0E82:0036 FF061400    INC     Word Ptr [0014]
0E82:003A C3          RET
-u
0E82:003B FF061600    INC     Word Ptr [0016]
0E82:003F C3          RET
0E82:0040 FF061800    INC     Word Ptr [0018]
0E82:0044 C3          RET
0E82:0045 FF061A00    INC     Word Ptr [001A]
0E82:0049 C3          RET
0E82:004A FF061C00    INC     Word Ptr [001C]
0E82:004E C3          RET
-u
0E82:004F 94          XCHG    AX,SP
0E82:0050 43          INC     BX
0E82:0051 0B5050      OR      DX,[BX+SI+50]
0E82:0054 0420      ADD     AL,20
0E82:0056 49          DEC     CX
0E82:0057 41          INC     CX
0E82:0058 48          DEC     AX
0E82:0059 42          INC     DX

```

反汇编：  
旨在得到  
数据段 ds  
的地址。  
ds=0E80  
H

```

-g 0009
AX=0E80 BX=0000 CX=006F DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E80 ES=0E70 SS=0E80 CS=0EB2 IP=0009  NU UP EI PL ZR NA PE NC
0E82:0009 B90A00      MOV     CX,000A
-d ds:0
0E80:0000 38 00 45 00 54 00 52 00-49 00 58 00 63 00 3F 00  B.E.T.R.I.X.c.?.
0E80:0010 64 00 50 00 00 00 00 00-00 00 00 00 00 00 00 00  d.P.....
0E80:0020 1E 2B C0 50 B8 80 0E 8E-D8 B9 0A 00 BB 00 00 E8  .+@P8...X9...h
0E80:0030 06 00 83 C3 02 E2 F8 CB-83 3F 3C 72 19 83 3F 46  ...C.bxK.?<r..?F
0E80:0040 72 19 83 3F 50 72 19 83-3F 5A 72 19 83 3F 64 72  r..?Pr..?Zr..?dr
0E80:0050 19 FF 06 1E 00 C3 FF 06-14 00 C3 FF 06 16 00 C3  ....C....C....C
0E80:0060 FF 06 18 00 C3 FF 06 1A-00 C3 FF 06 1C 00 C3 94  ....C....C....C
0E80:0070 43 0B 50 50 04 20 49 41-48 42 4C 42 48 43 4C 0A  C.PP. IAHBLBHCL.

```

查看在源代码数据段定义的数据：

高地址存放高位，低地址存放低位，所以第一个分数是 0038H（即 56D），其余 9 个分数同理。最后一个分数是 0050H（80D）后紧接着是 s5,s6,s7,s8,s9,s10 的值，均为 0000H

```

-g
Program terminated normally (0)
-r
AX=0E80 BX=0000 CX=006F DX=0000 SP=FFFC BP=0000 SI=0000 DI=0000
DS=0E80 ES=0E70 SS=0E80 CS=0E82 IP=0009 NU UP EI PL ZR NA PE NC
0E82:0009 B90A00 MOV CX,000A
-d ds:0
0E80:0000 38 00 45 00 54 00 52 00-49 00 58 00 63 00 3F 00 B.E.T.R.I.X.c.?.
0E80:0010 64 00 50 00 01 00 02 00-01 00 04 00 01 00 01 00 d.P.....
0E80:0020 1E 2B C0 50 B8 80 0E 8E-D8 B9 0A 00 BB 00 00 E8 .+eP8...X9...h
0E80:0030 06 00 83 C3 02 E2 F8 CB-83 3F 3C 72 19 83 3F 46 ...C.bxK.?<r..?F
0E80:0040 72 19 83 3F 50 72 19 83-3F 5A 72 19 83 3F 64 72 r..?Pr..?Zr..?dr
0E80:0050 19 FF 06 1E 00 C3 FF 06-14 00 C3 FF 06 16 00 C3 ....C....C....C
0E80:0060 FF 06 18 00 C3 FF 06 1A-00 C3 FF 06 1C 00 C3 94 ....C....C....C
0E80:0070 43 0B 50 50 04 20 49 41-48 42 4C 42 48 43 4C 0A C.PP. IAHBLBHCL.

```

执行后数据段内容:

前 10 个字单元依然是分数, 不变。变化的是从第十一个字单元开始, 变化如下:

0000→0001; 0000→0002; 0000→0001; 0000→0004; 0000→0001; 0000→0001;

分别对应的是 s5,s6,s7,s8,s9,s10 的值, 即低于 60 分的有 1 人, 60~69 分的有 2 人, 70~79 分的有 1 人, 80~89 分的有 4 人, 90~99 分的有 1 人, 100 分的有 1 人。

结果正确

#### 四、实验结果与分析:

程序一是一个很简单的小程序, 通过编写, 熟悉主函数和子函数的基本知识。以前实验的程序都没有主函数, 直接定义数据段和代码段就开始写代码, 而此次实验的程序一要求用 ret 指令返回 dos, 那么就要把代码以主程序形式出现。这样编程的习惯更有利于思维的清晰和代码的重用。通过程序一的实验, 知道要在主函数用 ret 返回 dos 需要在之前做准备工作: push ds→sub ax,ax→push ax。

程序二的算法也不难。众多的判断, 众多的条件转移, 需要我们编程时思维要清晰。需要注意的一点是我们定义分数时是以 word 的形式定义的。如果我们希望寄存器间接寻址方式对定义分数和分界线分数比较时, 需要写明是 word ptr [reg], 如果不写明是 word ptr, 系统不知道是以 byte 还是 word 就会报错。当然也可以把定义的分分数通过 mov 指令传送到一个 16 位的寄存器, 然后通过立即寻址方式比较。