



【实验题目】快速排序实验

【实验目的】掌握 MASM 汇编语言编程的基本方法。

【实验说明】

- 安装 MASM5 (只有在 32 位 DOS 下可以用)。因此需要安装 DOSBox。
- 运行 DOSBox 并把 MASM5 所在的目录映射为 C 盘：

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Welcome to DOSBox v0.74

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount c: c:\masm5
Drive C is mounted as local directory c:\masm5\

Z:\>c:
C:\>_
```

- 对 C 盘下的汇编程序(first.asm)进行汇编、链接和执行：

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

C:\>masm first.asm
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987. All rights reserved.

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

51718 + 464826 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>link first.obj

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

Run File [FIRST.EXE]:
List File [NUL.MAP]:
Libraries [LIB1]:
LINK : warning L4021: no stack segment

C:\>first
Hello World
C:\>_
```

命令后加上分号，不会要求输入；也可以用批处理 m.bat 执行：C:>m first

- 参考源码：
 - first.asm 显示一个字符串
 - second.asm 输入字符并显示它。
- ASCII 码：
 - 0~9: 30H, ..., 39H
 - A~F: 41H, ..., 46H
 - 换行: 0AH
 - 回车: 0DH



- 默认的段寄存器

$$EA = \begin{cases} [BX] \\ [BP] \end{cases} + \begin{cases} [SI] \\ [DI] \end{cases} + \text{位移量}$$

BX默认使用DS
BP默认使用SS

- [BX]或[BP]、[SI]或[DI]、位移量三项中可以任意选择
- 如果没有[BX]和[BP]，[SI]和[DI]默认的段寄存器分别是DS和ES
- 如果只有位移量(标号)，看标号所在段，如果不是DS，则要加段超越前缀。

- 提供的源码仅供参考

【实验内容】

- 1、(qsort.asm)对数据进行快速排序（从大到小排，地址越小数据越大），最后把排序结果显示出来。

参考截屏：

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
15h 2fh 30h 55h 64h 67h 69h 78h 81h 9ah

C:\>m qsort
C:\>masm qsort;
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987. All rights reserved.

51680 + 464864 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>link qsort;

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

LINK : warning L4021: no stack segment

C:\>qsort
13h 34h 47h 58h 62h 70h 81h 93h
15h 2fh 30h 55h 64h 67h 69h 78h 81h 9ah

C:\>_
```

运行截屏：（从大到小排，地址越小数据越大）

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
9ah 81h 78h 69h 67h 64h 55h 30h 2fh 15h

C:\>m qsort
C:\>masm qsort;
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987. All rights reserved.

51604 + 464940 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>link qsort;

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

LINK : warning L4021: no stack segment

C:\>qsort
93h 81h 70h 62h 58h 47h 34h 13h
9ah 81h 78h 69h 67h 64h 55h 30h 2fh 15h

C:\>
```

源码：

;qsort.asm

data segment



```
cnt1 dw 8
d1 db 47h, 58h, 34h, 62h, 13h, 81h, 93h, 70h ; unsigned int
cnt2 dw 10
d2 db 67h, 30h, 64h, 69h, 2fh, 81h, 9ah, 78h, 55h, 15h ; unsigned int
data ends
```

code segment

```
assume cs:code, ds:data
```

start:

```
mov ax, data
mov ds, ax
mov es, ax
```

```
mov si, offset d1
mov cx, [cnt1]
mov di, si
add di, cx
dec di
call qsort
```

```
mov si, offset d1
mov cx, [cnt1]
call disp8
```

```
mov si, offset d2
mov cx, [cnt2]
mov di, si
add di, cx
dec di
call qsort
```

```
mov si, offset d2
mov cx, [cnt2]
call disp8
```

```
mov ah, 4ch ; 功能：结束程序，返回 DOS 系统
int 21h ; DOS 功能调用
```

; 用 16 进制显示从 si 开始的 cx 个字节

; si-base address, cx-count

disp8 proc near

```
mov bx, 0
mov bp, cx
```

next:

```
cmp bx, bp
```



```
jge    complete

mov     al, [si+bx]
mov     ch, al

mov     cl, 4
shr     al, cl
call    disp4           ;显示高 4 位

mov     al, ch
call    disp4           ;显示低 4 位

mov     dl, 104          ;显示 h
mov     ah, 02h
int     21h

mov     dl, 32           ;显示空格
mov     ah, 02h
int     21h

inc     bx
jmp     next

complete:
mov     dl, 10
mov     ah, 02h
int     21h

ret

disp8 endp

;用 16 进制数显示 al 低 4 位
disp4 proc near
mov     cl, 4
shl     al, cl
shr     al, cl

cmp     al, 10
jae     letter

add     al, 30h
mov     dl, al
mov     ah, 02h
int     21h
jmp     success

letter:
letter:
```



```
add    al, 57h
mov     dl, al
mov     ah, 02h
int     21h
```

```
success:
```

```
ret
```

```
disp4 endp
```

;si-第一个数的地址，di-最后一个数的地址，cx-总共排序多少个数

```
qsort proc near
```

```
    cmp     cx, 1
    jle     noneed
```

```
    mov     al, [si]
    mov     dx, 0
    mov     bp, 0
```

```
while:
```

```
    mov     bx, bp
    neg     bx
    add     bx, dx
    inc     bx
    cmp     bx, cx
    je      finish
```

```
highfind:
```

```
    mov     bx, bp
    neg     bx
    add     bx, dx
    inc     bx
    cmp     bx, cx
    je      lowfind
```

```
    mov     bx, bp
    mov     ah, [bx+di]
    cmp     ah, al
    ja      highmove
    dec     bp
    jmp     highfind
```

```
highmove:
```

```
    mov     bx, bp
    mov     ah, [bx+di]
    mov     bx, dx
    mov     [bx+si], ah
```



lowfind:

```
mov    bx, bp
neg     bx
add     bx, dx
inc     bx
cmp     bx, cx
je      findout
```

```
mov     bx, dx
mov     ah, [si+bx]
cmp     ah, al
jb      lowmove
inc     dx
jmp     lowfind
```

lowmove:

```
mov     bx, dx
mov     ah, [si+bx]
mov     bx, bp
mov     [di+bx], ah
```

findout:

```
jmp     while
```

finish:

```
mov     bx, dx
mov     [si+bx], al
```

```
push    di
push    cx
push    dx
push    si
```

```
add     di, bp
dec     di
mov     cx, di
sub     cx, si
inc     cx
call    qsort
```

```
pop     si
pop     dx
pop     cx
pop     di
```

```
add     si, dx
inc     si
mov     cx, di
```



```
sub    cx, si
inc     cx
call   qsort
```

noneed:

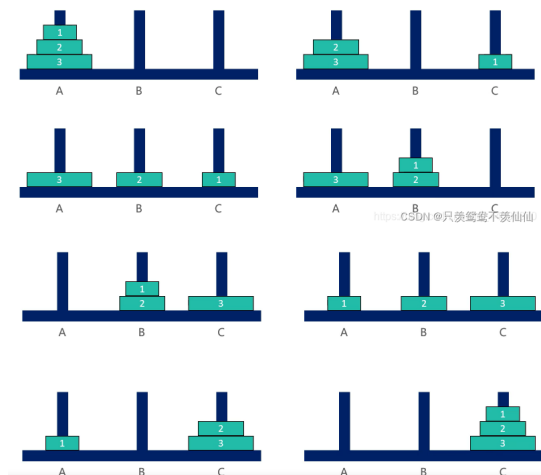
```
ret
```

```
qsort endp
```

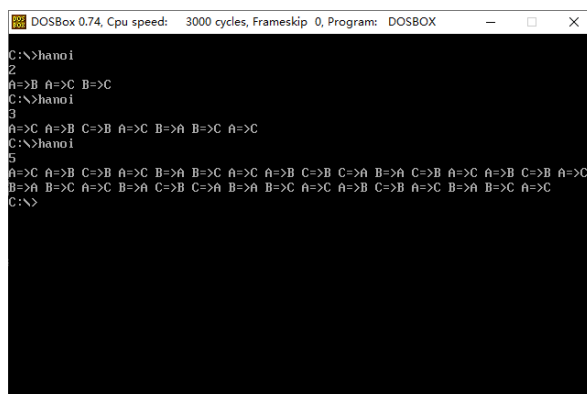
```
code ends
```

```
end start
```

- 2、(hanoi.asm)用递归的方法实现汉诺塔：把所有盘从A柱移到C柱，小盘必须在大盘之上，可以利用B柱。



参考截屏：



运行截屏：



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip: 0, Program: DOSBOX

0 Warning Errors
0 Severe Errors

C:\>link hanoi:

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

LINK : warning L4021: no stack segment

C:\>hanoi
2
A=>B A=>C B=>C
C:\>hanoi
3
A=>C A=>B C=>B A=>C B=>A B=>C A=>C
C:\>hanoi
4
A=>B A=>C B=>C A=>B C=>A C=>B A=>B A=>C B=>C B=>A C=>A B=>C A=>B A=>C B=>C
C:\>hanoi
5
A=>C A=>B C=>B A=>C B=>A B=>C A=>C A=>B C=>B C=>A B=>A C=>B A=>C A=>B C=>B A=>C
B=>A B=>C A=>C B=>A C=>B C=>A B=>A B=>C A=>C A=>B C=>B A=>C B=>A B=>C A=>C
C:\>
```

源码:

;hanoi.asm

code segment

assume cs:code

start:

```
mov    ah, 1
int     21h
and     al, 0fh
mov     ah, 0
mov     cx, ax
```

```
mov     dl, 0ah      ;回车
mov     ah, 02
int     21h
```

```
mov     ax, 'A'      ;source
mov     dx, 'C'      ;destination
mov     bx, 'B'      ;middle
```

```
;     mov cx, 3
call    hanoi
```

```
mov     ah, 4ch      ; 功能: 结束程序, 返回 DOS 系统
int     21h          ; DOS 功能调用
```

; 显示: ax => dx (单盘)

move proc near

```
push    ax
push    dx
```




```
        mov     dh, dl
        mov     dl, al           ;显示 ax
        mov     ah, 02h
        int     21h

        mov     dl, '='         ;显示=
        mov     ah, 02h
        int     21h
        mov     dl, '>'         ;显示>
        mov     ah, 02h
        int     21h

        mov     dl, dh           ;显示 dx
        mov     ah, 02h
        int     21h

        mov     dl, ' '         ;显示空格
        mov     ah, 02h
        int     21h

        pop     dx
        pop     ax
        ret

move endp

        ; ax =>dx, middle-bx, cx-count of layers
hanoi proc near
        push    ax
        push    bx
        push    cx
        push    dx

        cmp     cx, 1
        ja      digui
        call    move
        jmp     finish

digui:
        mov     bp, dx
        mov     dx, bx
        mov     bx, bp
        dec     cx
        call    hanoi

        mov     bp, dx
        mov     dx, bx
```



```
mov     bx, bp
call    move

mov     bp, ax
mov     ax, bx
mov     bx, bp
call    hanoi

finish:
pop     dx
pop     cx
pop     bx
pop     ax

ret
hanoi endp

code ends
end start
```

- 3、(hanoi2.asm)用非递归的方法实现汉诺塔。参考：通过参数进栈出栈实现；不用参考递归的做法，完全重新考虑；注意进出栈次序；每批参数中有分类参数。

运行截屏：

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip: 0, Program: DOSBOX

0 Warning Errors
0 Severe Errors

C:\>link hanoi2;

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

LINK : warning L4021: no stack segment

C:\>hanoi2
2
A=>B A=>C B=>C
C:\>hanoi2
3
A=>C A=>B C=>B A=>C B=>A B=>C A=>C
C:\>hanoi2
4
A=>B A=>C B=>C A=>B C=>A C=>B A=>B A=>C B=>C B=>A C=>A B=>C A=>B A=>C B=>C
C:\>hanoi2
5
A=>C A=>B C=>B A=>C B=>A B=>C A=>C A=>B C=>B C=>A B=>A C=>B A=>C A=>B C=>B A=>C
B=>A B=>C A=>C B=>A C=>B C=>A B=>A B=>C A=>C A=>B C=>B A=>C B=>A B=>C A=>C
C:\>
```

源码：

;hanoi.asm

code segment



```
assume cs:code
```

```
start:

    mov     ah, 1           ;输入值
    int     21h
    and     al, 0fh         ;去掉高 4 位
    mov     ah, 0
    mov     cx, ax          ;cx 为盘数

    mov     dl, 0ah         ;回车
    mov     ah, 02
    int     21h

    mov     ax, 'A'         ;source
    mov     dx, 'C'         ;destination
    mov     bx, 'B'         ;middle
; mov     cx, 3
    call    hanoi

    mov     ah, 4ch         ; 功能：结束程序，返回 DOS 系统
    int     21h             ; DOS 功能调用

; 显示: ax => dx (单盘)
move proc near
    push    ax
    push    dx

    mov     dh, dl
    mov     dl, al          ;显示 ax
    mov     ah, 02h
    int     21h

    mov     dl, '='         ;显示=
    mov     ah, 02h
    int     21h
    mov     dl, '>'         ;显示>
    mov     ah, 02h
    int     21h

    mov     dl, dh          ;显示 dx
    mov     ah, 02h
    int     21h

    mov     dl, ' '         ;显示空格
    mov     ah, 02h
    int     21h
```



```
        pop    dx
        pop    ax
        ret

move endp

; ax => dx, middle-bx, cx-count of layers
hanoi proc near
        mov    si, 0

        push   ax
        push   bx
        push   cx
        push   dx
        inc    si

while:
        cmp    si, 0
        je     finish

        pop    dx
        pop    cx
        pop    bx
        pop    ax
        dec    si

        cmp    cx, 1
        ja     divide
        call   move
        jmp    while

divide:
        mov    bp, ax
        mov    ax, bx
        mov    bx, bp
        dec    cx
        push   ax
        push   bx
        push   cx
        push   dx
        inc    si

        mov    bp, ax
        mov    ax, bx
        mov    bx, bp
        mov    bp, 1
```



```
        push    ax
        push    bx
        push    bp
        push    dx
        inc     si

        mov     bp, dx
        mov     dx, bx
        mov     bx, bp
        push    ax
        push    bx
        push    cx
        push    dx
        inc     si

        jmp     while

finish:
        ret

hanoi endp

code ends

end start
```

【完成情况】

是否完成以下步骤? (√完成 ×未完成)

1 [√] 2 [√] 3 [√]

【实验体会】

写出实验过程中遇到的问题，解决方法和自己的思考;并简述实验体会（如果有的话）。

指令格式虽然中有写 `bp+si+disp` 和 `bp+di+disp` 两种有效地址，但是实际使用 `bp` 寄存器为一些指令寻址会发生错误，网络上查到的一些资料里也只使用 `bx` 寄存器寻址，所以用 `bx` 配合 `si` 或 `di` 寻址更妥当。函数调用的时候会使用栈存储返回地址，所以在函数调用过程中判断栈空不能使用判断 `sp` 指针是否为 0 的方法。即使是在不同函数中，跳转指令跳转地址的标注名字也不能一样。

【交实验报告】

每位同学单独完成本实验内容并填写实验报告。

交作业地点: <http://172.18.187.251/netdisk/default.aspx?vm=21org>

编程实验/快速排序和汉诺塔

截止日期: 2023 年 1 月 13 日 23:00

上传文件: 学号_姓名_快速排序和汉诺塔.doc