

**计算机组成原理与汇编语言**

**课程设计报告**

姓名：王云鹏

班级：物联网工程1802

学号：8213180228

学院：计算机学院

2020.9.19

# 实验目的与任务

课程设计是计算机原理与汇编语言教学过程中的重要环节。本课程设计主要目的是使计算机专业学生深入学习计算机原理与汇编语言知识，进一步提高学生计算机原理与汇编语言综合能力和程序设计技能，锻炼运用计算机原理与汇编语言解决实际问题的能力。

# 实验要求

1. 认真查阅资料，独立完成设计任务，每道题都必须上机通过。
2. 独立思考，培养综合分析问题解决问题和调试程序的能力。
3. 按时完成课程设计，写出课程设计报告。
4. 独立完成

# 实验环境

硬件：

处理器：1.1 GHz 四核Intel Core i5

内存：8 GB 3733 MHz LPDDR4X

软件：

系统：macOS 10.15.6

调试工具：DOSBox

实验一

# **一、实验题目**

求100以内的素数，要使用到子程序，完成这些素数的输出显示和个数统计以及求和计算，均以十进制形式输出。

# 背景知识

素数：素数一般指质数。质数是指在大于1的自然数中，除了1和它本身以外不再有其他因数的自然数。

# 思路分析

求100以内素数：遍历从2到100的所有自然数i，对每个数i验证其是否为素数，即验证从2到i-1是否有i的因数。若i有因数，则i不是素数，若i没有因数，则i是素数。并在此过程中求这些素数的个数以及和。

输出显示十进制：因为内存中对数字的存储为16进制，为了符合阅读习惯，输出时应转化为10进制。

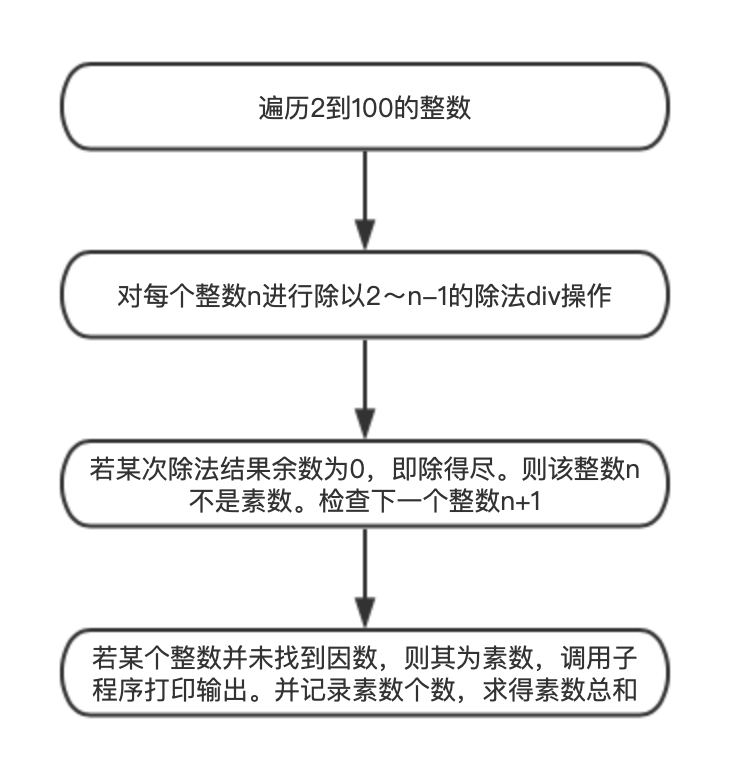
16进制每一位上可以是从小到大为0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F16个大小不同的数，即逢16进1，其中用A，B，C，D，E，F（字母使用大写）这六个字母来分别表示10，11，12，13，14，15。[16进制](https://baike.baidu.com/item/16%E8%BF%9B%E5%88%B6" \t "/Users/mine/Documents\\x/_blank)数的第0位的[权值](https://baike.baidu.com/item/%E6%9D%83%E5%80%BC" \t "/Users/mine/Documents\\x/_blank)为16的0次方，第1位的权值为16的1次方，第2位的权值为16的2次方……所以，在第N（N从0开始）位上，如果是是数 X （X 大于等于0，并且X小于等于 15，即：F）表示的大小为 X \* 16的N次方。

在往屏幕上打印十进制时，需使用ASCII (American Standard Code for Information Interchange)码，其中48～57为0到9十个阿拉伯数字。即输出调用中断的时候需要在原本的数字上加上48。

在数字大于一位数的时候，需要对于每位数字挨个输出其ASCII 字符，即循环除以每位数的权值，输出余数的ASCII码。

调用子程序：子程序既可以写在同一个code segment中，作为proc near存在，在主程序中只需用call调用即可。也可以写在另外的代码文件中，两份文件各自masm编译，link的时候需要将两个obj文件一起链接，运行的时候只运行主程序即可。

# 实现步骤



# 实验结果

汇编

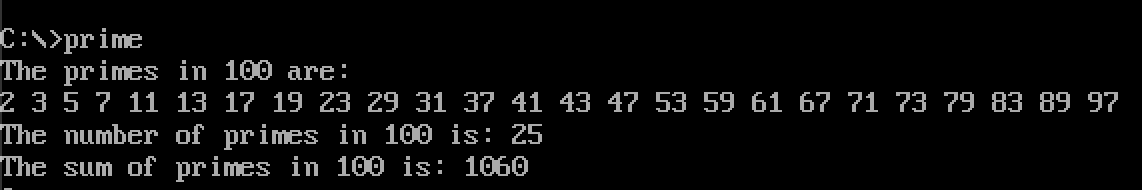




链接：



运行：



# 感想体验

由于一个假期都未接触汇编语言，因此初次开始写时有些生疏。但好在之前的课程学习认真，所以很快掌握了code的要点，通过教材与搜索引擎结合的方式，迅速复习了8086的汇编语法。

在这道题中，我复习了汇编循环与条件的语法套路，温习了用DOSBox单步执行调试的要点，为之后更加困难的题目奠定了基础。

# **七、源代码**

Prime.asm

extrn decout : far

data segment

prime\_array dw 100 dup(?)*;存放素数的数组*

prime\_num dw 0*;素数个数*

prime\_sum dw 0*;素数总和*

prime\_array\_msg db "The primes in 100 are: ",0dh,0ah,'$'*;输出素数提示*

prime\_num\_msg db 0dh,0ah,"The number of primes in 100 is: ",'$'*;输出素数个数提示*

prime\_sum\_msg db 0dh,0ah,"The sum of primes in 100 is: ",'$'*;输出素数总和提示*

space db " ",'$'*;空格*

data ends

code segment

main proc far

assume cs:code,ds:data

start:

push ds

sub ax,ax

push ax

mov ax,data

mov ds,ax

lea dx,prime\_array\_msg*;循环中会输出素数*

mov ah,9

int 21h

mov si,0*;数组下标*

mov cx,1*;从2开始遍历到100，找素数S*

circu\_out:

cmp cx,100

je out\_put

inc cx

mov bx,1*;对于每一个数i，除以从2到i-1，有因数不是素数*

circu\_in:

inc bx

cmp bx,cx

je is\_prime*;没找到因数,是素数*

mov ax,cx

div bl

cmp ah,0

jne circu\_in*;余数不为0说明此数不为因数，接着寻找*

je circu\_out*;余数为0说明此数是因数，看下一个数*

is\_prime:

mov prime\_array[si],cx

mov bx,cx

call far ptr decout*;输出素数*

lea dx,space*;输出空格*

mov ah,9

int 21h

add si,2*;以字节为单位*

inc prime\_num

add prime\_sum,cx

jmp circu\_out

out\_put:*;计算结束，输出个数和总数*

lea dx,prime\_num\_msg

mov ah,9

int 21h

mov bx,prime\_num*;输出素数个数*

call far ptr decout

lea dx,prime\_sum\_msg

mov ah,9

int 21h

mov bx,prime\_sum*;;输出素数总和*

call far ptr decout

ret

main endp

code ends

end start

Dout.asm

public decout

data segment

zero db 0

data ends

code segment

decout proc far

assume cs:code,ds:data

push ds

push ax *; 参数准备*

push bx

push cx

push dx

mov ax,data

mov ds,ax

cmp bx,0

jne next

mov dl,'0'

mov ah,2

int 21h

jmp return

next:

mov zero,0

mov cx,10000 *; 为除10000预置参数*

call dout *; 调用dout过程，输出万位*

*; 依次输出千位、百位、十位、各位....*

mov cx,1000

call dout

mov cx,100

call dout

mov cx,10

call dout

mov cx,1

call dout

return:

pop dx

pop cx

pop bx

pop ax

pop ds

ret

decout endp

dout proc near

mov dx,0 *; dx清0,除cx时，被除数为dx,ax*

mov ax,bx *; 将bx值（第一次为输入的数，随后为余数）赋值给ax*

div cx *; (dx,ax)，实际为ax（dx==0）除以cx（cx值在调用程序前设置，作为参数传递进来）*

xchg ax,dx *; ax与dx交换内容。交换后：ax中为余数，dx中为商*

mov bx,ax *; 将ax值（余数）赋予bx（进入下一轮运算）*

*; 如果用户前面输入65535，那么在第一轮除以10000后，dx中值为6，bx中值为5535*

cmp dl,0

jne outanum *; 如果dx中值不为0，则直接输出相应的数值*

cmp zero,0 *; 如果dx中值为0,那么判断是前面无意义的0，还是中间有意义的0。*

*; 如305，那么如果不进行次判断将输入00305。通过此位可以不输出前面两个0，但是输出中间0。*

je con *; 如果是前面无意义的0 ，则不输出*

outanum:

mov zero,1 *; 如果输出了一个大于0的数字，则置标志位为1，使得其后所有0都会被输出*

add dl,30h *; dl中数值加上30h，变成对应的ASCII码。*

mov ah,2

int 21h *; 输出该数字*

con:

ret *; 如果要求可以输入负数？输入0会输出吗？*

dout endp

code ends

end

实验二

1. **实验题目**

用递归计算50以内Fibonacci数,以十进制数输出.

# 背景知识

Fibonacci数：斐波那契数列（Fibonacci sequence），又称[黄金分割](https://baike.baidu.com/item/%E9%BB%84%E9%87%91%E5%88%86%E5%89%B2/115896" \t "/Users/mine/Documents\\x/_blank)数列、因[数学家](https://baike.baidu.com/item/%E6%95%B0%E5%AD%A6%E5%AE%B6/1210991" \t "/Users/mine/Documents\\x/_blank)[莱昂](https://baike.baidu.com/item/%E8%8E%B1%E6%98%82" \t "/Users/mine/Documents\\x/_blank)[纳多](https://baike.baidu.com/item/%E7%BA%B3%E5%A4%9A" \t "/Users/mine/Documents\\x/_blank)·斐波那契（Leonardoda Fibonacci）以兔子繁殖为例子而引入，故又称为“[兔子数列](https://baike.baidu.com/item/%E5%85%94%E5%AD%90%E6%95%B0%E5%88%97/6849441" \t "/Users/mine/Documents\\x/_blank)”，指的是这样一个数列：0、1、1、2、3、5、8、13、21、34、……在数学上，斐波那契数列以如下被以递推的方法定义：F(1)=1，F(2)=1, F(n)=F(n - 1)+F(n - 2)（n ≥ 3，n ∈ N\*）

# 思路分析

Fibonacci数列在数学上是用递归定义的，因此用递归的设计方法实现起来非常容易。用类C语言实现求第n位Fibonacci数可如下所示：

int Fibo(int n){

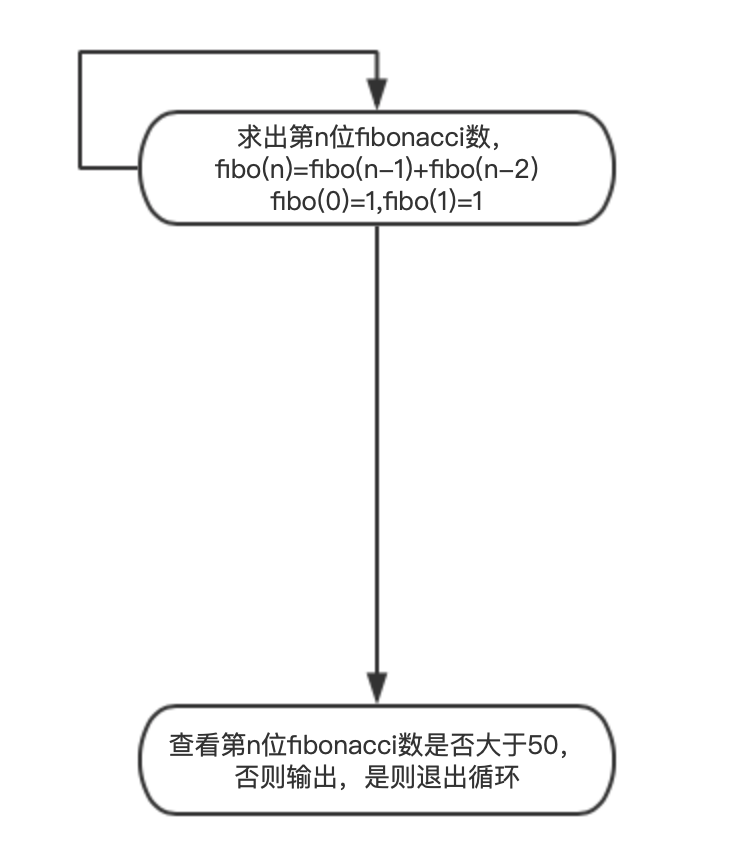
If(n==0 or n==1)then: return 1;

Else: return Fibo(n-1)+Fibo(n-2);

}

若用汇编实现这样的思路，则需要仔细注意参数的返回，在子程序（函数）中要注意保存在子程序中被改变的寄存器（用来返回结果的寄存器除外）。否则会因寄存器的改变而无法保证递归出栈过程的正确性。

# 实现步骤

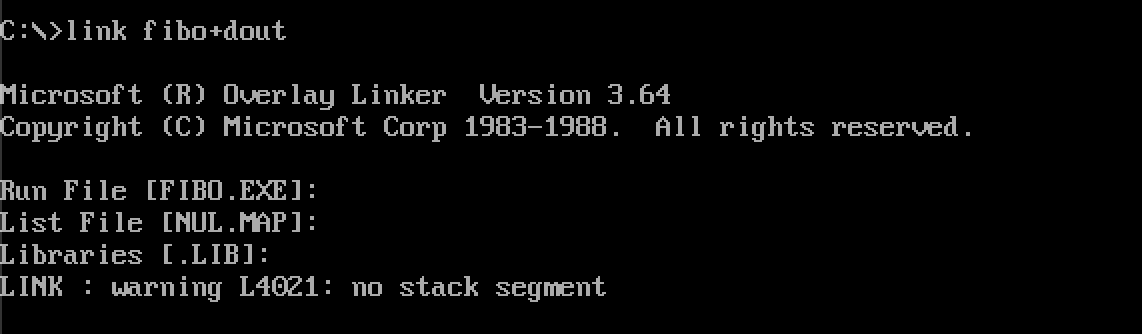


# 实验结果

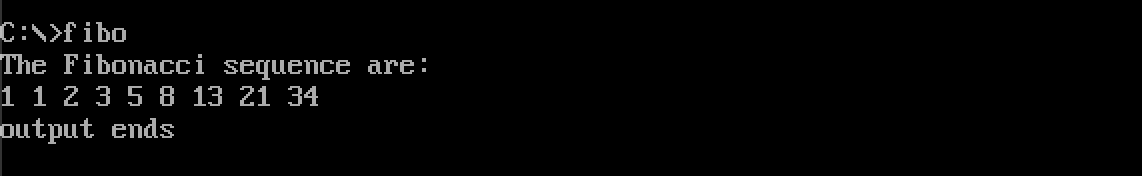
汇编：



链接：



运行：



# 感想体验

在这道题的编写过程中，遇到了一个非常难以理解的错误，就是在逻辑正确的情况下，输出结果不正确。在我单步执行调试后，发现是因为子程序，即函数fibo(n)的返回过程中，用来暂时保存结果的寄存器cx发生了改变，而改变的位置在递归的更深一层。于是反应过来，在子程序中需要保存被改变的寄存器（除用来返回结果的以外）。出现这种bug的原因主要是因为汇编中子程序的编写经验不足，同时并未十分系统的学习汇编中递归函数的编写。

由于之前的错误，在发现错误原因之前，我曾尝试将结果保存到栈中，再在下一次调用函数时取出结果。但是程序跳转到了一个无法理解的位置，想起来系统在ret时会从栈中取出返回地址，于是认为用栈来保存返回结果是个不靠谱的想法，除非用另外一个栈，而不是和系统共用一个。

这次用汇编来写递归函数，让我对高级语言中递归函数的理解加深了一层，也对操作系统在不同程序间转移时，将临时结果保存到栈中的必要性，有了更深的理解。

# **源代码**

Fibo.asm

extrn decout : far

data segment

output\_msg db 'The Fibonacci sequence are: ',0dh,0ah,'$'

end\_msg db 0dh,0ah,'output ends',0dh,0ah,'$'

data ends

code segment

main proc far

assume cs:code,ds:data

start:

push ds

sub ax,ax

push ax

mov ax,data

mov ds,ax

lea dx,output\_msg*;输出提示信息*

mov ah,9

int 21h

mov ax,1

circulation:

push ax

call fibo*;计算第ax位Fibonacci数，存入bx*

cmp bx,50

ja return\_main

call decout*;输出bx*

mov dl,' '*;输出空格*

mov ah,2

int 21h

pop ax

inc ax

jmp circulation

return\_main:

lea dx,end\_msg

mov ah,9

int 21h

mov ah,4ch

int 21h

ret

main endp

*;fibo递归实现fibonacci数列*

*;输入参数：ax表示第几位*

*;输出参数：第ax位Fibonacci数字存入bx*

fibo proc near

push ax

push cx

push dx

cmp ax,1*;若是前两位Fibonacci数，就是1*

je fibo12

cmp ax,2

je fibo12

dec ax*;调用fibo（i-1）*

call fibo

mov cx,bx*;结果存入cx*

dec ax*;调用fibo(i-2),之前已经减了一次1*

call fibo

mov dx,bx*;结果存入dx*

add cx,dx*;fibo(i)=fibo(i-1)+fibo(i-2)*

mov bx,cx

jmp return

fibo12:

mov bx,1

return:

pop dx

pop cx

pop ax

ret

fibo endp

code ends

end start

实验三

1. **实验题目**

建立学生成绩文件，包括学号、成绩、名次，从键盘输入学号，实现成绩、名次等信息的查询：

学号 姓名 成绩 名次

04131 张三 902 1

04132 李四 806 2

# 背景知识

磁盘文件的读写有两种方法，一种称为文件控制块;另一种方法称为文件标记，前者在读写文件时首先要设定文件控制块，知名文件所在的当前磁盘的驱动器、文件名，同时还要制定所读写的文件所处的当前块号、当前记录号、记录长度等参量，此外还需要设置磁盘传输区。在读磁盘数据文件时，应先将磁盘上有关数据读入磁盘传输区，然后再传送至目的的内存区。在写磁盘数据文件时，要写入磁盘文件的数据也必须先送入磁盘传输区，然后再执行写操作。此种方法的缺点是不支持树形目录结构。目前常用的方法是利用文件标记读写文件，无论用什么方法读写文件，都要解决以下几个问题∶

A） 用户程序要告诉操作系统将要存取那个文件。

B） 无论是从磁盘独处的数据，还是要写入磁盘的数据都必须存放在一个制定的内

存缓冲区中，这个内存缓冲区叫数据传输区（DTA）。

C）在读一个磁盘文件之前，要先打开文件，然后才能将文件内容读入内存。而在读写一个文件之前，要先建立一个新文件名，再将内存中的内容写入磁盘。

D）在存取文件之后，特别是在写入文件之后，务必将此文件关闭。利用文件标记读写文件的主要特点是∶

通过建立文件、打开文件，将磁盘路径名、文件名转换为文件标记或件号，在以后读写文件的操作中均要与文件标记打交道。在一个系统中，可同时打开多个文件，并配置相应的文件标记。在利用文件标记读写磁盘文件是要掌握以下几个要点∶

1） 使用建立文件、打开文件系统功能之前，都必须将DS∶DX指向驱动器名、路径名、文件名和以数值零为结尾的ASCIZ字符串的首地址。

2）对于一个新文件要用3CH系统功能调用，建立文件。（如果不是新文件，此步骤可略）

3）已存在的文件则利用3DH系统功能调用，打开此文件。

4） 如果文件建立、打开成功，则CF=0，并且在AX寄存器中返回文件标记。

5）在建立、打开文件成功之后，即可对文件进行读、写操作。如果要对文件进行读操作，则在打开文件之后，用3FH系统功能调用将文件读入数据缓冲区，调用前的入口参数是∶文件标记存入 BX，读入文件的字节数放入CX，数据缓冲区的起始地址存入 DS∶DX。调用完毕，AX返回实际读入的字节数。

6） 读入数据缓冲区的文件，可利用40H系统功能调用将文件写入指定的磁盘。调用入口参数是∶文件标记存入 BX，要写入的字节数放入CX，DS∶DX指向要写入数据缓冲区的首地址。调用结束后 AX返回实际写入的字节数。如果 AX<CX，则表示磁盘空间已满，但系统不提示错误信息。

7）在读、写文件完成后，要用3EH系统功能调用关闭，释放原占有的文件号，并将缓冲区的数据最终写入磁盘。特别是对于写磁盘文件操作，在写入之后，必须要关闭文件，否则该数据文件将会丢失。此外，特别要注意的是，在一个系统中不能同时建立两个以上的新文件。

8）在文件建立或打开后，文件指针的初始位置是指向文件首偏移为' 0'处，如果不想从文件首开始读、写文件，则可用系统功能调用42H移动文件指针的方法实现。调用前，将文件标记存入 BX，所需移动偏移量的字节数（CX为高位部分）存放于CXx∶DX;调用后在DX∶AX中返回读写指针移动后的位置。移动的方式有三种∶

AL=0 指针从文件头移动到由CX∶DX所指定的偏移量处;AL=1 指针从当前位置移动到由CX∶DX所指定的偏移量处;AL=2 指针从文件尾移动到由CX∶DX所指定的偏移量处;

# 思路分析

由于需要对文件进行操作，而我并未系统学习过汇编语言对于文件的操作，所以主要参考网络上的知识进行操作。这次主要用的是int21h中断中的几个调用，包括：3ch建立文件，3dh打开文件（al=0读，al=1写，al=2读写），3eh关闭文件。3fh读取文件，40h写入文件，42h挪动文件指针（al=0文件头，al=1当前位置，al=2文件尾）。

建立学生成绩文件：包括学号sno，姓名sname，成绩grade，排名rank。文件中的每一行代表一条学生信息，单项信息之间用空格分割。例如：

11 ming 100 1

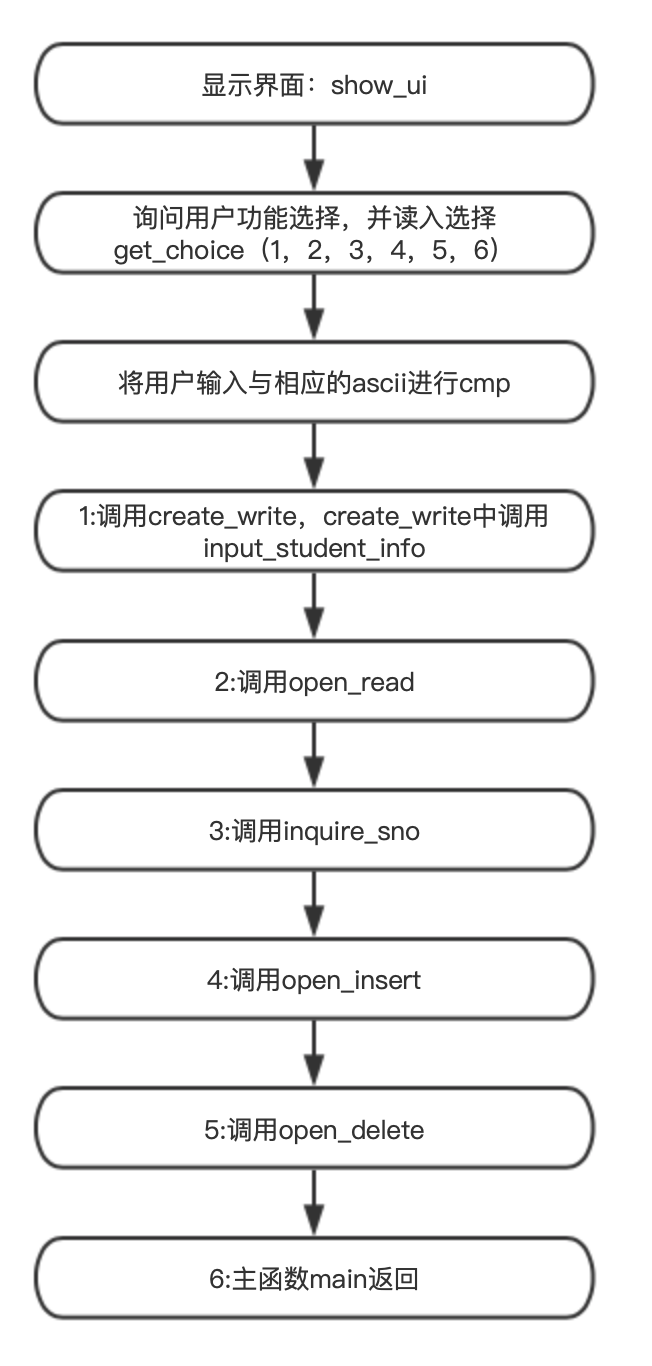
33 hong 88 3

有格式的信息方便对于信息的阅读，分割、处理等操作。

为了让系统更加完善，除了题目要求的根据学号查询学生的各项信息之外，我还加入了5个功能，包括：

1. 建立文件，写入信息：根据用户输入的文件位置和文件名称信息新建文件，并循环读入用户输入的学生信息，将其写入文件，直到用户输入回车为止。
2. 打开文件，读出信息：根据用户输入的文件名打开文件，读入其中的信息到缓冲区，再调用中断将缓冲区的信息打印到屏幕上
3. 查询信息：根据用户输入的文件名打开文件，读入其中的信息到缓冲区，再根据用户输入的学号，循环对缓冲区中的每一行学号信息尝试匹配，匹配上之后则打印该行的信息，并退出循环；对所有行信息尝试匹配皆失败后，提示未找到该学号并结束。
4. 插入信息：根据用户输入的文件名打开文件，打印其中的信息，再循环读入用户想要插入的学生信息，并将其写入到文件的末尾。
5. 删除信息：由于并未找到删除文件中信息的调用，因此采用迂回的方式删除信息。步骤为：先根据用户输入的文件名打开文件并打印其中信息，询问用户想要删除的学生学号，根据学号对文件中的每一行信息中的学号尝试匹配。匹配上后，记录该行上一行的行尾位置，以及该行下一行的行首位置。根据这两个位置，将该行前面的所有内容读入到一个缓冲区，并将该行之后所有的信息读入到另一个缓冲区。最后新建一个和原文件同名的文件进行覆盖，将前面的两个缓冲区内容写入进去，则完成了对于某一行信息的删除。

# 实现步骤

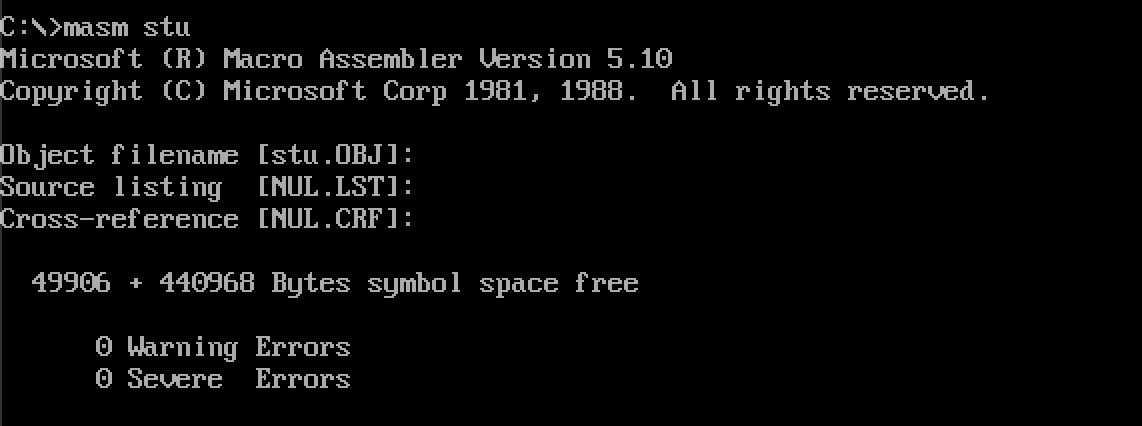


子程序（函数）信息列表如下：

|  |  |  |
| --- | --- | --- |
| 函数名 | 功能 | 输出 |
| show\_ui | 显示功能菜单 | 无 |
| get\_choice | 读入用户选择，以字符ascii的形式， | al |
| create\_write | 新建一个文件，询问文件名，询问数据进行写入 | 文件句柄（写入到缓冲区） |
| open\_read | 询问文件名，打开，从中读出信息。打不开则报错 | 文件句柄（写入到缓冲区） |
| inquire\_sno | 根据学号用打开或创建的文件显示该学号学生的全部信息 | 打印 |
| open\_insert | 打开一个文件，显示其中的内容，向文件最后插入用户输入的信息 | 写入文件 |
| open\_delete | 打开一个文件，显示信息，询问得到学号，删除该学号所在行信息（将其前面和后面的信息挪到同名的新建文件中） | 新建的其他信息文件 |

# 五、实验结果

汇编：

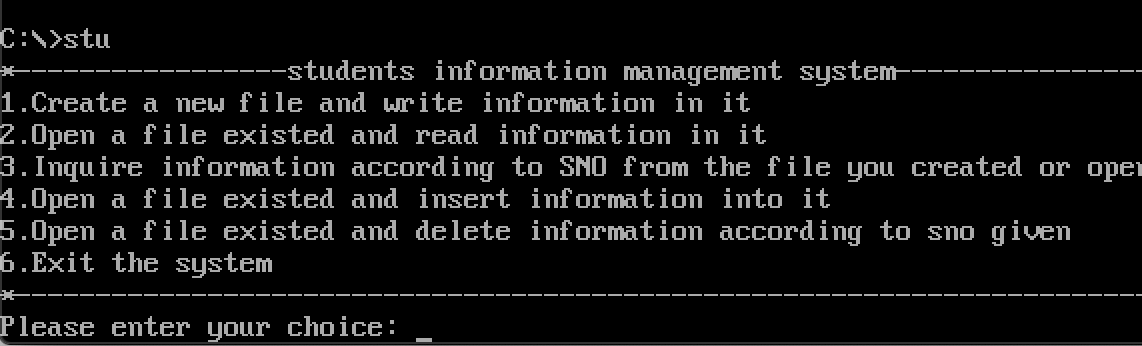


链接：

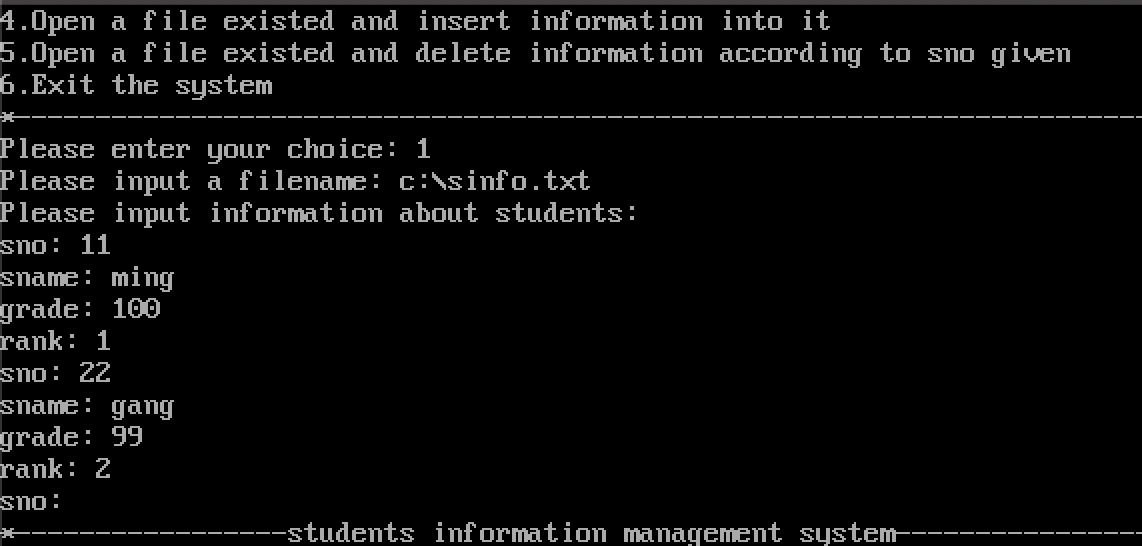


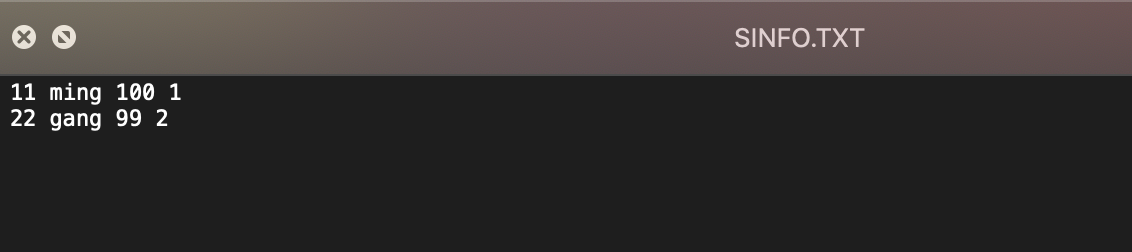
运行：

主菜单：

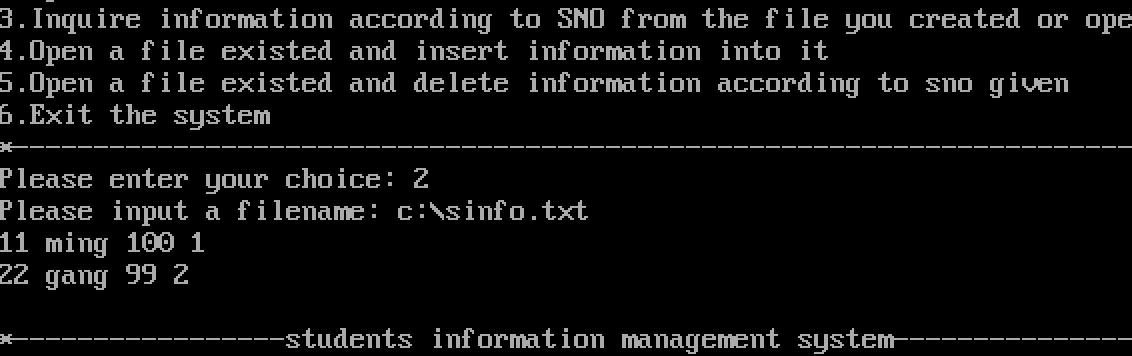


功能1: 新建文件，写入学生信息

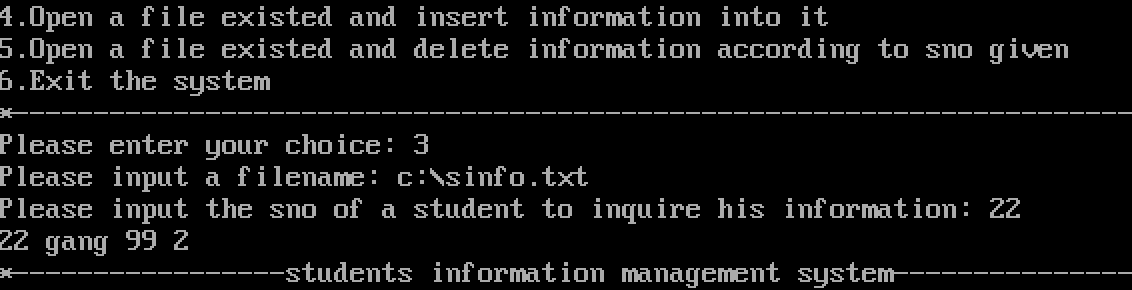




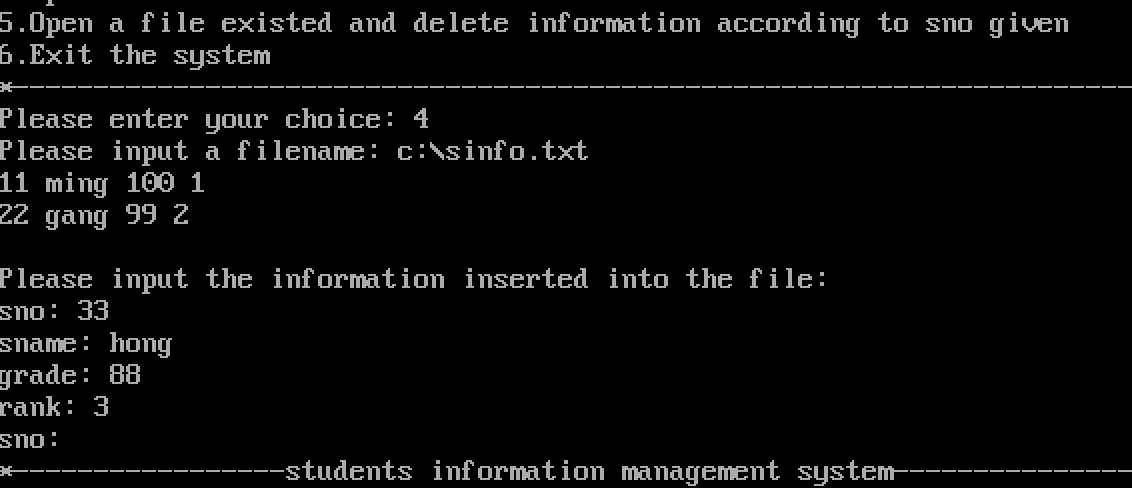
功能2: 打开文件，查看其中的信息

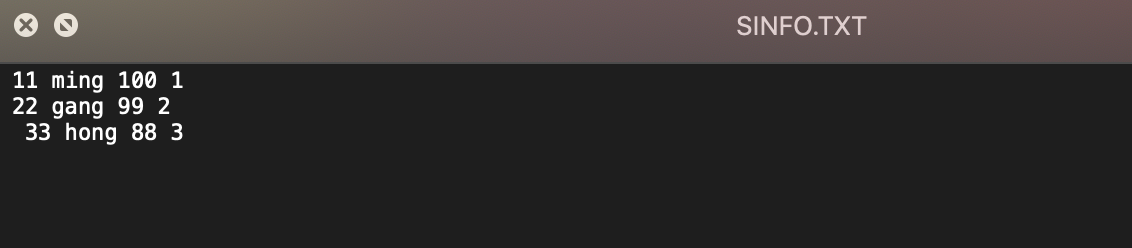


功能3: 根据用户给出的学号查询某学生信息

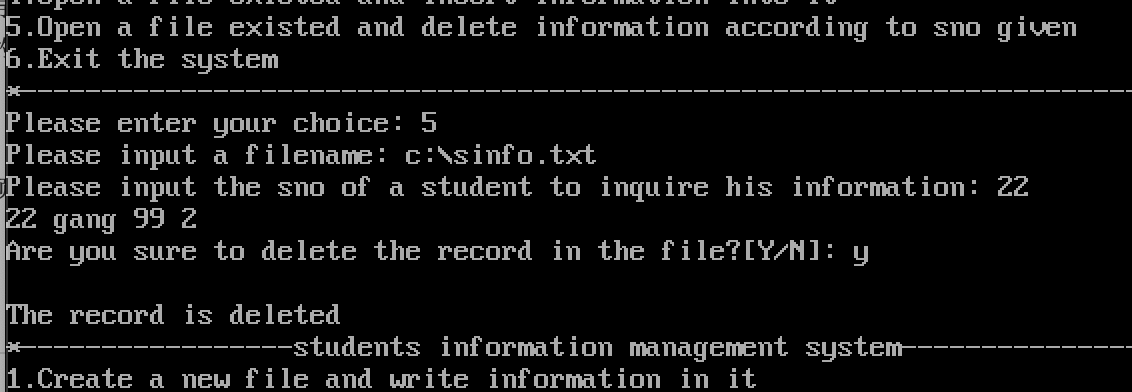


功能4: 往文件中插入信息

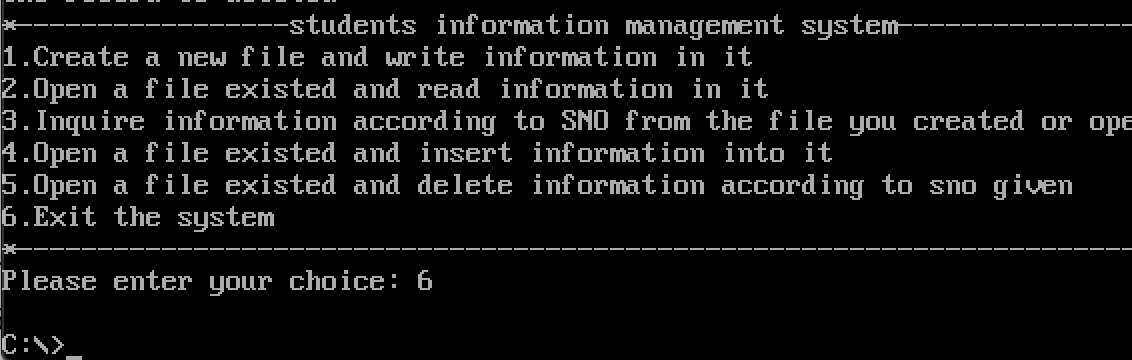




功能5: 根据用户给出的学号，删除某学生信息



功能6: 退出系统



# 感想体验

这道题由于实现的功能在用汇编实现时非常的复杂，所以可以说是一个忆苦思甜的过程，编写过程中让我非常想念高级语言方便的文件操作，例如Java极其方便的随机读写文件功能。并且由于内存和外存交互操作十分频繁，导致对于内存的管理非常消耗精力，有时不得不单步调试几百步才能发现内存中出现的错误。同时由于DOS环境的限制，只可能出现命令行界面，而可视化的图形用户界面没有实现的可能性。

记录几个刻骨铭心的bug：

1. 在刚开始写了个test测试对于文件读写时，一直无法成功打开文件。四处询问之后了解到，DOSBox可以算是一个虚拟的系统，模拟了80年代的微软DOS系统，其要求将其虚拟的C盘（或者其他盘符）挂载到当前文件系统的某个文件夹下，在其中的路径都是以这个挂载的C盘为基础进行操作。因此，使用原来文件系统的绝对路径是无法成功访问的，因为在DOSBox的虚拟C盘中不存在这样的一个路径，同理使用相对路径也是无法成功的。
2. 在实现到第三个功能时，同样需要打开文件，因此我将第二个功能的代码复制了过来。可是在第二个功能的子程序中正常运行的代码竟然失败了，这十分令人费解。几经周折之后，我发现原来打开文件需要对al进行赋值，即al=0时只读，al=1时只写，al=2时读写。出现这样令人懊恼的bug，主要是因为我对于文件的操作没有一个系统的学习，若不是调试时运气好，真无法发现这样的bug。不过调试几个小时的bug也给了我一些深刻的经验教训：1、在排除所有不可能之后，剩下的答案无论多么无法置信，都是唯一正确的答案。2、小黄鸭调试法非常的好用，能够帮人迅速的梳理思路，发现那个令人无法置信的错误原因。3、一个不大懂写代码的室友比小黄鸭更好用。在你和他解释这段代码是干什么用的的时候，会更迅速的梳理思路。
3. 在实现第五个功能时，无法成功将文件中的信息读入到缓冲区。经过一番周折，发现错误：文件指针在上一次读取之后到达了文件末尾，再次读入需要手动将文件指针调到文件头或者任何想要读取的位置。另外一个错误，用42h号中断调用时，mov dx写成了lea dx，结果找了半天也找不到错误原因。因为逻辑正确的情况下，错误的原因往往非常隐蔽。当然写成lea dx的原因主要是写了太多的9号输出调用。

# **七、源代码**

data segment

ui\_format1 db '\*-----------------students information management system----------------------\*',0dh,0ah,'$'*;界面*

ui\_create\_file db '1.Create a new file and write information in it',0dh,0ah,'$'

ui\_open\_file db '2.Open a file existed and read information in it',0dh,0ah,'$'

ui\_inquire db '3.Inquire information according to SNO from the file you created or opened',0dh,0ah,'$'

ui\_sort db '4.Open a file existed and insert information into it',0dh,0ah,'$'

ui\_delete db '5.Open a file existed and delete information according to sno given',0dh,0ah,'$'

ui\_exit db '6.Exit the system',0dh,0ah,'$'

ui\_format2 db '\*-----------------------------------------------------------------------------\*',0dh,0ah,'$'

ui\_get\_choice db 'Please enter your choice: ','$'

ctrl\_enter db 0dh,0ah,'$'*;回车*

space db ' '*;空格*

error\_choice db 'The choice you input is illegal!',0dh,0ah,'$'*;选项错误警告信息*

error\_file db 0dh,0ah,'file error!',0dh,0ah,'$'*;文件错误警告信息*

write\_file db 'c:\sinfo.txt',0*;新建和写入文件*

filename\_get\_msg db 'Please input a filename: ','$'

filename\_write\_max\_length db 50

filename\_write\_act\_length db ?

filename\_write db 50 dup(0)

file\_write\_handle dw ?

student\_info\_get\_msg db 'Please input information about students: ',0dh,0ah,'$'*;学生信息*

student\_info\_sno\_msg db 'sno: ','$'

student\_info\_sname\_msg db 'sname: ','$'

student\_info\_grade\_msg db 'grade: ','$'

student\_info\_rank\_msg db 'rank: ','$'

student\_info\_buffer\_max\_length db 20

student\_info\_buffer\_act\_length db ?

student\_info\_buffer db 20 dup(0)

file\_read\_handle dw ?*;打开和读入文件*

file\_read\_buffer db 1024 dup(0)

inquire\_get\_sno\_msg db 'Please input the sno of a student to inquire his information: $'*;查询*

inquire\_not\_find\_sno\_msg db 'The sno provided can not be found',0dh,0ah,'$'

inquire\_get\_sno\_max\_length db 10*;存放查询学生的学号*

inquire\_get\_sno\_act\_length db ?

inquire\_get\_sno db 10 dup(0)

open\_insert\_information\_msg db 'Please input the information inserted into the file: ',0dh,0ah,'$'

open\_delete\_information\_msg db 'Are you sure to delete the record in the file?[Y/N]: $'

open\_delete\_success\_msg db 'The record is deleted',0dh,0ah,'$'

open\_delete\_row\_head\_pos dw 0

open\_delete\_next\_row\_head\_pos dw 0

open\_delete\_row\_array db 1024 dup(0)

open\_delete\_new\_row\_array db 1024 dup(0)

data ends

code segment

main proc far

assume cs:code,ds:data

start:

push ds

sub ax,ax

push ax

mov ax,data

mov ds,ax

menu:

call show\_ui*;显示功能列表*

call get\_choice*;读入用户选择到al*

cmp al,49*;1的asc码为49*

je choice\_1

cmp al,50

je choice\_2

cmp al,51

je choice\_3

cmp al,52

je choice\_4

cmp al,53

je choice\_5

cmp al,54

je choice\_6

jmp error*;不在6之内，即为非法选项*

choice\_1:

call create\_write

jmp menu

choice\_2:

call open\_read

jmp menu

choice\_3:

call inquire\_sno

jmp menu

choice\_4:

call open\_insert

jmp menu

choice\_5:

call open\_delete

jmp menu

choice\_6:

jmp return

error:

lea dx,error\_choice

mov ah,9

int 21h

jmp menu

return:

mov ah,4ch

int 21h

ret

main endp

show\_ui proc near*;显示功能列表*

push dx

push ax

lea dx,ui\_format1

mov ah,9

int 21h

lea dx,ui\_create\_file

mov ah,9

int 21h

lea dx,ui\_open\_file

mov ah,9

int 21h

lea dx,ui\_inquire

mov ah,9

int 21h

lea dx,ui\_sort

mov ah,9

int 21h

lea dx,ui\_delete

mov ah,9

int 21h

lea dx,ui\_exit

mov ah,9

int 21h

lea dx,ui\_format2

mov ah,9

int 21h

lea dx,ui\_get\_choice

mov ah,9

int 21h

pop ax

pop dx

ret

show\_ui endp

*;功能：读入用户选择，以字符的形式，ascii*

*;输入参数：无*

*;输出参数：al*

get\_choice proc near

push dx

mov ah,1

int 21h

lea dx,ctrl\_enter

mov ah,9

int 21h

pop dx

ret

get\_choice endp

*;功能：新建一个文件，询问文件名，询问数据进行写入*

*;输入参数：无*

*;输出参数：文件句柄（写入到缓冲区）*

create\_write proc near

lea dx,filename\_get\_msg*;输入文件名提示*

mov ah,9

int 21h

lea dx,filename\_write\_max\_length*;读入文件名*

mov ah,10

int 21h

mov bl,filename\_write\_act\_length

mov bh,0

mov filename\_write[bx],0*;将回车置0*

mov ah,3ch*;建立文件*

mov cx,0

lea dx,filename\_write

int 21h

jc error\_create\_write

mov file\_write\_handle,ax*;将新建的文件句柄保存，注：\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

lea dx,ctrl\_enter*;输入一个回车*

mov ah,9

int 21h

lea dx,student\_info\_get\_msg*;输入学生信息提示*

mov ah,9

int 21h

input\_circulation:

lea dx,student\_info\_sno\_msg*;输入学号提示*

mov ah,9

int 21h

call input\_student\_info

cmp student\_info\_buffer[0],0dh*;若输入了个回车，循环结束*

je return\_create\_write

lea dx,student\_info\_sname\_msg*;输入名字提示*

mov ah,9

int 21h

call input\_student\_info*;从键盘输入名字，存入文件*

lea dx,student\_info\_grade\_msg*;输入成绩提示*

mov ah,9

int 21h

call input\_student\_info

lea dx,student\_info\_rank\_msg*;输入名次提示*

mov ah,9

int 21h

call input\_student\_info

mov ah,40h*;一行写入完，再写入个回车换行*

mov bx,file\_write\_handle

mov cx,2*;回车换行两字符*

lea dx,ctrl\_enter

int 21h

jc error\_create\_write

jmp input\_circulation

error\_create\_write:*;输出错误信息*

lea dx,error\_file

mov ah,9

int 21h

return\_create\_write:

mov ah,3eh*;关闭新建的文件，将缓冲区的数据保存到文件*

mov bx,file\_write\_handle

int 21h

ret

create\_write endp

input\_student\_info proc near*;从键盘输入信息，再写到文件中*

lea dx,student\_info\_buffer\_max\_length*;从键盘读入信息*

mov ah,10

int 21h

mov ah,40h*;将输入的信息写入文件*

mov bx,file\_write\_handle*;使用了之前建立的文件句柄，注：\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

mov cl,student\_info\_buffer\_act\_length

mov ch,0

lea dx,student\_info\_buffer

int 21h

jc error\_input\_student\_info

mov ah,40h*;在输入的信息后加上一个空格作为分隔符*

mov bx,file\_write\_handle

mov cx,1*;空格一字符*

lea dx,space

int 21h

jc error\_input\_student\_info

lea dx,ctrl\_enter*;在屏幕上打印一个回车*

mov ah,9

int 21h

jmp return\_input\_student\_info

error\_input\_student\_info:

lea dx,error\_file*;输出错误信息*

mov ah,9

int 21h

return\_input\_student\_info:

ret

input\_student\_info endp

*;功能：询问文件名，打开，从中读出信息。打不开则报错*

*;输入参数：无*

*;输出参数：文件句柄（写入到缓冲区）*

open\_read proc near

lea dx,filename\_get\_msg*;输入文件名提示*

mov ah,9

int 21h

lea dx,filename\_write\_max\_length*;读入文件名*

mov ah,10

int 21h

mov bl,filename\_write\_act\_length

mov bh,0

mov filename\_write[bx],0*;将回车置0*

mov ah,3dh*;打开文件*

mov al,2

lea dx,filename\_write

int 21h

jc error\_open\_read

mov file\_read\_handle,ax

mov ah,3fh*;读入文件*

mov bx,file\_read\_handle

mov cx,1024

lea dx,file\_read\_buffer

int 21h

jc error\_open\_read

mov bx,ax

mov file\_read\_buffer[bx],'$'*;末尾加上$*

lea dx,ctrl\_enter*;打印一个回车*

mov ah,9

int 21h

lea dx,file\_read\_buffer*;在屏幕上打印读入的信息*

mov ah,9

int 21h

lea dx,ctrl\_enter*;打印一个回车*

mov ah,9

int 21h

jmp return\_open\_read

error\_open\_read:*;输出错误*

lea dx,error\_file

mov ah,9

int 21h

return\_open\_read:

mov ah,3eh*;关闭打开的文件句柄*

mov bx,file\_read\_handle

int 21h

ret

open\_read endp

*;功能：根据学号用打开或创建的文件显示该学号学生的全部信息*

*;输入参数：无*

*;输出参数：打印*

inquire\_sno proc near

lea dx,filename\_get\_msg*;输入文件名提示*

mov ah,9

int 21h

lea dx,filename\_write\_max\_length*;读入文件名*

mov ah,10

int 21h

mov bl,filename\_write\_act\_length

mov bh,0

mov filename\_write[bx],0*;将回车置0*

mov ah,3dh*;打开文件*

mov al,0

lea dx,filename\_write

int 21h

*; jc error\_inquire\_sno;jc跳不了那么远*

jnc next\_temp1

jmp error\_inquire\_sno

next\_temp1:

mov file\_read\_handle,ax

mov ah,3fh*;读入文件*

mov bx,file\_read\_handle

mov cx,1024

lea dx,file\_read\_buffer

int 21h

*; jc error\_inquire\_sno;jc跳不了那么远*

jnc next\_temp2

jmp error\_inquire\_sno

next\_temp2:

mov bx,ax

mov file\_read\_buffer[bx],'$'*;末尾加上$*

lea dx,ctrl\_enter*;打印一个回车*

mov ah,9

int 21h

lea dx,inquire\_get\_sno\_msg*;询问学号提示*

mov ah,9

int 21h

lea dx,inquire\_get\_sno\_max\_length*;从键盘读入学号*

mov ah,10

int 21h

mov bl,inquire\_get\_sno\_act\_length

mov bh,0

mov inquire\_get\_sno[bx],0*;将回车置0*

lea dx,ctrl\_enter*;打印回车*

mov ah,9

int 21h

mov bx,0*;记录文件缓冲区中位置*

*; mov si,inquire\_get\_sno\_act\_length*

mov al,inquire\_get\_sno\_act\_length

mov ah,0

mov si,ax

row\_circulation:*;对于每一行记录*

mov di,0*;记录学号位置*

mov cx,0*;记录空格数目*

cmp file\_read\_buffer[bx+si],' '*;查看学号位数是否对得上*

jne not\_equal

sno\_circulation:*;对于每行记录的学号来说*

push bx

mov bx,di

mov al,inquire\_get\_sno[bx]

pop bx

cmp file\_read\_buffer[bx+di],'$'

je not\_find\_sno

cmp file\_read\_buffer[bx+di],al*;查看学号的每一位是否对得上*

jne not\_equal

inc di

mov al,inquire\_get\_sno\_act\_length

mov ah,0

cmp di,ax*;比较结束，学号每一位都相等*

je equal

jmp sno\_circulation*;没有比较结束，就接着比较*

not\_equal:*;该行的学号与输入学号不同，去查下一行（通过找到四个空格的方式）*

cmp file\_read\_buffer[bx],' '

je find\_space

inc bx

jmp not\_equal

find\_space:*;找空格，找到四个，去下一行*

inc cx

cmp cx,4

je next\_row

inc bx

jmp not\_equal

next\_row:*;去下一行*

add bx,3

jmp row\_circulation

equal:*;该行的学号与输入学号相同，输出该行所有信息（找到四个空格为止）*

mov dl,file\_read\_buffer[bx]

mov ah,2

int 21h

cmp file\_read\_buffer[bx],' '

je find\_space\_equal

inc bx

jmp equal

find\_space\_equal:*;找到四个空格，停止输出，退出查找*

inc cx

cmp cx,4

je end\_row\_circulation

inc bx

jmp equal

end\_row\_circulation:*;输出回车换行，退出查找*

lea dx,ctrl\_enter

mov ah,9

int 21h

jmp return\_inquire\_sno

not\_find\_sno:*;没找到输入的学号，提示并退出*

lea dx,inquire\_not\_find\_sno\_msg

mov ah,9

int 21h

jmp return\_inquire\_sno

error\_inquire\_sno:*;文件错误信息*

lea dx,error\_file

mov ah,9

int 21h

return\_inquire\_sno:

mov ah,3eh*;关闭打开的文件*

mov bx,file\_read\_handle

int 21h

ret

inquire\_sno endp

*;功能：打开一个文件，显示其中的内容，向文件最后插入用户输入的信息*

*;输入参数：无*

*;输出参数：写入文件*

open\_insert proc near

lea dx,filename\_get\_msg*;输入文件名提示*

mov ah,9

int 21h

lea dx,filename\_write\_max\_length*;读入文件名*

mov ah,10

int 21h

mov bl,filename\_write\_act\_length

mov bh,0

mov filename\_write[bx],0*;将回车置0*

mov ah,3dh*;打开文件*

mov al,2

lea dx,filename\_write

int 21h

*; jc error\_open\_insert*

jnc next\_temp4

jmp error\_open\_insert

next\_temp4:

mov file\_write\_handle,ax

mov ah,3fh*;读入文件*

mov bx,file\_write\_handle

mov cx,1024

lea dx,file\_read\_buffer

int 21h

jc error\_open\_insert

mov bx,ax

mov file\_read\_buffer[bx],'$'*;末尾加上$*

lea dx,ctrl\_enter*;打印一个回车*

mov ah,9

int 21h

lea dx,file\_read\_buffer*;在屏幕上打印读入的信息*

mov ah,9

int 21h

lea dx,ctrl\_enter*;打印一个回车*

mov ah,9

int 21h

lea dx,open\_insert\_information\_msg*;输入信息提示*

mov ah,9

int 21h

input\_circulation\_open\_insert:

lea dx,student\_info\_sno\_msg*;输入学号提示*

mov ah,9

int 21h

call input\_student\_info

cmp student\_info\_buffer[0],0dh*;若输入了个回车，循环结束*

*; je return\_create\_write;超出跳转范围*

jne next\_temp3

jmp return\_open\_insert

next\_temp3:

lea dx,student\_info\_sname\_msg*;输入名字提示*

mov ah,9

int 21h

call input\_student\_info*;从键盘输入名字，存入文件*

lea dx,student\_info\_grade\_msg*;输入成绩提示*

mov ah,9

int 21h

call input\_student\_info

lea dx,student\_info\_rank\_msg*;输入名次提示*

mov ah,9

int 21h

call input\_student\_info

mov ah,40h*;一行写入完，再写入个回车换行*

mov bx,file\_write\_handle

mov cx,2*;回车换行两字符*

lea dx,ctrl\_enter

int 21h

jc error\_open\_insert

jmp input\_circulation\_open\_insert

error\_open\_insert:

lea dx,error\_file

mov ah,9

int 21h

return\_open\_insert:

mov ah,3eh*;关闭打开的文件，将缓冲区的数据保存到文件*

mov bx,file\_write\_handle

int 21h

ret

open\_insert endp

*;功能：打开一个文件，显示信息，询问得到学号，删除该学号所在行信息（将其前面和后面的信息挪到同名的新建文件中）*

*;输入参数：无*

*;输出参数：新建的其他信息文件*

open\_delete proc near

call inquire\_sno

lea dx,open\_delete\_information\_msg*;是否删除提示*

mov ah,9

int 21h

mov ah,1*;从键盘输入Y/N，除了Y,y都默认为N*

int 21h

push ax

lea dx,ctrl\_enter*;回车*

mov ah,9

int 21h

pop ax

cmp al,'Y'*;从键盘输入Y/N，除了Y,y都默认为N*

je sure\_to\_delete

cmp al,'y'

je sure\_to\_delete

jmp return\_open\_delete

sure\_to\_delete:

mov ah,3dh*;打开文件*

mov al,0

lea dx,filename\_write

int 21h

jnc next\_temp1\_open\_delete

jmp error\_open\_delete

next\_temp1\_open\_delete:

mov file\_read\_handle,ax

mov ah,3fh*;读入文件*

mov bx,file\_read\_handle

mov cx,1024

lea dx,file\_read\_buffer

int 21h

jnc next\_temp2\_open\_delete

jmp error\_open\_delete

next\_temp2\_open\_delete:

mov bx,ax

mov file\_read\_buffer[bx],'$'*;末尾加上$*

lea dx,ctrl\_enter*;打印一个回车*

mov ah,9

int 21h

mov bx,0*;记录文件缓冲区中位置*

*; mov si,inquire\_get\_sno\_act\_length*

mov al,inquire\_get\_sno\_act\_length

mov ah,0

mov si,ax

open\_delete\_row\_circulation:*;对于每一行记录*

mov di,0*;记录学号位置*

mov cx,0*;记录空格数目*

cmp file\_read\_buffer[bx+si],' '*;查看学号位数是否对得上*

jne open\_delete\_not\_equal

open\_delete\_sno\_circulation:*;对于每行记录的学号来说*

push bx

mov bx,di

mov al,inquire\_get\_sno[bx]

pop bx

cmp file\_read\_buffer[bx+di],'$'

*; je open\_delete\_not\_find\_sno*

jne next\_temp5

jmp open\_delete\_not\_find\_sno

next\_temp5:

cmp file\_read\_buffer[bx+di],al*;查看学号的每一位是否对得上*

jne open\_delete\_not\_equal

inc di

mov al,inquire\_get\_sno\_act\_length

mov ah,0

cmp di,ax*;比较结束，学号每一位都相等*

je open\_delete\_equal

jmp open\_delete\_sno\_circulation*;没有比较结束，就接着比较*

open\_delete\_not\_equal:*;该行的学号与输入学号不同，去查下一行（通过找到四个空格的方式）*

cmp file\_read\_buffer[bx],' '

je open\_delete\_find\_space

inc bx

jmp open\_delete\_not\_equal

open\_delete\_find\_space:*;找空格，找到四个，去下一行*

inc cx

cmp cx,4

je open\_delete\_next\_row

inc bx

jmp open\_delete\_not\_equal

open\_delete\_next\_row:*;去下一行*

add bx,3

jmp open\_delete\_row\_circulation

open\_delete\_equal:

*;该行的学号与输入学号相同，记下该行的上一行行尾位置，求出下一行行首位置*

*;将该行前后的部分分别读入内存，以这些内容建立同名文件*

push bx

dec bx

mov open\_delete\_row\_head\_pos,bx*;记录该行上一行行尾位置*

pop bx

open\_delete\_equal\_circulation:*;找到四个空格，找下一行行首位置*

cmp file\_read\_buffer[bx],' '

je open\_delete\_find\_space\_equal

inc bx

jmp open\_delete\_equal\_circulation

open\_delete\_find\_space\_equal:*;找到四个空格，找下一行行首位置*

inc cx

cmp cx,4

je open\_delete\_end\_row\_circulation

inc bx

jmp open\_delete\_equal\_circulation

open\_delete\_end\_row\_circulation:*;下一行行首位置*

add bx,3

mov open\_delete\_next\_row\_head\_pos,bx

mov ah,42h

mov al,0

mov bx,file\_read\_handle

mov cx,0*;将文件指针挪到文件头*

mov dx,0

int 21h

*; jc error\_open\_delete*

jnc next\_temp7

jmp error\_open\_delete

next\_temp7:

mov ah,3fh

mov bx,file\_read\_handle

mov cx,open\_delete\_row\_head\_pos*;读取被删除行前面的内容*

lea dx,open\_delete\_row\_array

int 21h

*; jc error\_open\_delete*

jnc next\_temp6

jmp error\_open\_delete

next\_temp6:

mov ah,42h

mov al,0

mov bx,file\_read\_handle

mov cx,0

mov dx,open\_delete\_next\_row\_head\_pos*;挪动指针到被删除行的下一行行首*

int 21h

jc error\_open\_delete

mov ah,3fh

mov bx,file\_read\_handle

mov cx,1024

lea dx,open\_delete\_new\_row\_array*;读取被删除行后面的内容*

int 21h

jc error\_open\_delete

mov ah,3eh

mov bx,file\_read\_handle*;关闭之前的文件*

int 21h

jc error\_open\_delete

mov ah,3ch*;建立同名文件，覆盖原文件*

mov cx,0

lea dx,filename\_write

int 21h

jc error\_open\_delete

mov file\_write\_handle,ax

mov ah,40h

mov bx,file\_write\_handle

mov cx,open\_delete\_row\_head\_pos

lea dx,open\_delete\_row\_array*;写入删除行前面的内容*

int 21h

jc error\_open\_delete

mov ah,40h

mov bx,file\_write\_handle

mov cx,open\_delete\_next\_row\_head\_pos

lea dx,open\_delete\_new\_row\_array*;写入删除行后的内容*

int 21h

jc error\_open\_delete

mov ah,3eh*;关闭写好的文件，将缓冲区中的数据保存到文件*

mov bx,file\_write\_handle

int 21h

jc error\_open\_delete

lea dx,open\_delete\_success\_msg*;提示删除成功*

mov ah,9

int 21h

jmp return\_open\_delete

open\_delete\_not\_find\_sno:*;没找到输入的学号，提示并退出*

lea dx,inquire\_not\_find\_sno\_msg

mov ah,9

int 21h

jmp return\_open\_delete

error\_open\_delete:

lea dx,error\_file

mov ah,9

int 21h

return\_open\_delete:

ret

open\_delete endp

code ends

end start

计算机组成原理虚拟平台的模型机实验

