

微机原理实验二

PB13206106

罗勇冠

一.实验目的

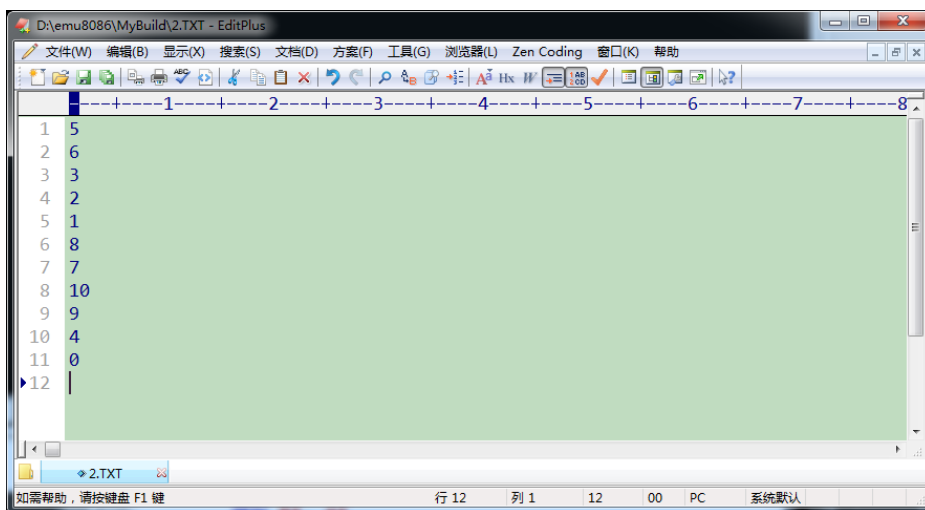
通过本次实验学会汇编语言的文件操作以及排序算法的汇编实现方式。

二.实验内容

题 2：编程实现排序算法，对文件 2.txt 中的无符号整数进行排序，排序结果输出到屏幕。数据的个数不超过 1024。

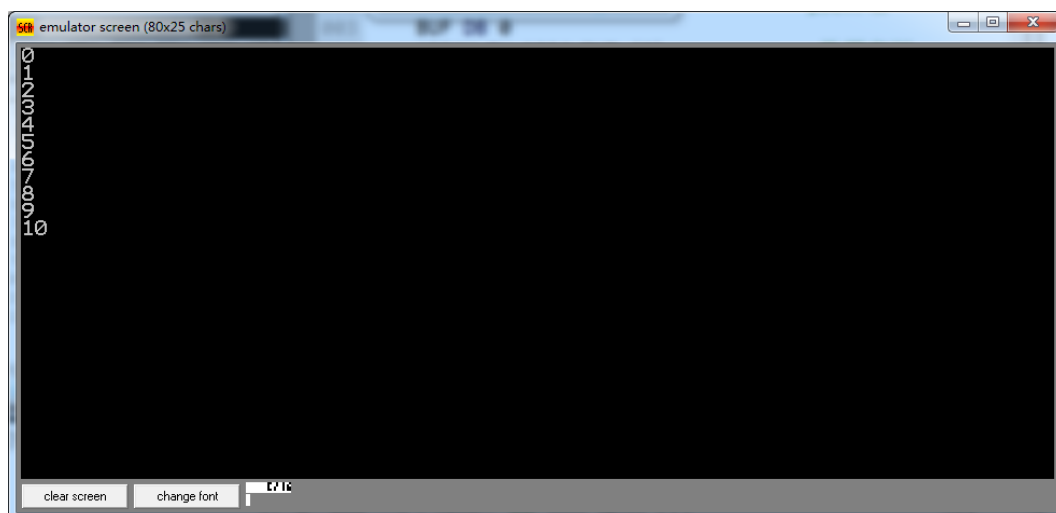
三.实验结果

测试数据为 11 个，2.txt 文件如下：



```
1 5
2 6
3 3
4 2
5 1
6 8
7 7
8 10
9 9
10 4
11 0
12 |
```

程序读取文件中的数字并进行冒泡排序后输出，运行结果如下：



```
0
1
2
3
4
5
6
7
8
9
10
```

四.实验分析

在数据段中，定义数据如下：

```
FILE DB '2.TXT', 0           ;文件名
BUF DB 0
ARRAY DW 1024 DUP (0)        ;数据空间
RANGE DW 22
ERRORINFO DB 0AH, 'ERROR!', '$' ;定义错误信息
HANDLE DW ?
```

FILE 为文件名，BUF 为字符缓存，ARRAY 为预留的 1024 个数据空间，RANGE 为读取的数据量的两倍，ERRORINFO 为错误信息，HANDLE 为文件句柄。

第一步，打开文件并读取数据，存入数组 ARRAY 中：

```
MOV AX, DATA
MOV DS, AX

MOV DX, OFFSET FILE
MOV AX, 3D00H
INT 21H           ;打开文件

MOV HANDLE, AX    ;转存文件代号
MOV BX, AX
MOV SI, 0          ;读入数下标
READFILE:
MOV CX, 1
LEA DX, BUF
MOV AH, 3FH
INT 21H           ;读取一个字节
CMP BUF, 0DH
JE READFILE       ;读到回车直接往下读
CMP BUF, 0AH
JE READNEW        ;读到换行符开始读下一个数
MOV AX, ARRAY[SI]
MOV CX, 10
MUL CX
MOV ARRAY[SI], AX ;数基本值乘 10
MOV CL, BUF
SUB CL, 30H
XOR CH, CH        ;将读入字节转成数字
ADD ARRAY[SI], CX ;加入到基本值
JMP READFILE      ;继续读下一字节
READNEW:
ADD SI, 2         ;数下标加一
CMP SI, RANGE
JB READFILE       ;若没读完则继续读
```

以上代码的主要思想是先打开文件，入口参数为AH=3D00H，DS:DX=表明文件的字符串，AL=打开方式标志位，出口参数为CF=0——打开成功，AX=文件句柄，否则AX=错误号（02H、03H、04H、05H或0CH）。然后读取文件，入口参数：AH=3FH，BX=文件句柄，CX=将要读出的字节数，DS:DX=存放字符的缓冲区地址，出口参数：CF=0——读取成功，AX=读取的字节数，否则，AX=错误号(05H或06H)。先一个字节一个字节读取，读完一个ASCII码后若该数据还没读完，则将单个ASCII码减去030H转换成数字再乘以10，与下一个数字相加，若读到换行回车符，表示整个数读取结束，将该数存入ARRAY数组中，并且移动下标。当读取的字节等于预读取的数据量后停止读数。

第二步，对ARRAY中的数据进行排序，结果按升序存储：

```
MOV DI, 0
BUBBLE:
MOV SI, 0
MOV AX, RANGE
SUB AX, DI
SUB AX, 2
SORT:
MOV BX, SI
ADD BX, 2
MOV CX, ARRAY[BX]
CMP ARRAY[SI], CX
JB RECY                                ;若相邻两数前者小于后者，跳转
MOV DX, ARRAY[SI]                      ;否则，交换两数位置
MOV CX, ARRAY[BX]
MOV ARRAY[SI], CX
MOV ARRAY[BX], DX

RECY:
ADD SI, 2
CMP SI, AX
JB SORT
ADD DI, 2
CMP DI, RANGE
JB BUBBLE
```

先置循环下标DI为0，BUBBLE标号为冒泡过程，SORT为排序过程，SORT的主要思路是对相邻两数进行递升排序，若不为递升顺序则交换两数位置，排完一遍序后下标自增进行下一遍冒泡过程，直至全部排好序。

第三步，将排好序的数组输出至控制台，并关闭文件：

```
MOV SI, 0                                ;初始取数的数组下标
OUTPUT:
MOV DI, 0                                ;记位
MOV BX, 100                              ;初始取值
PRINT:
MOV AX, ARRAY[SI]
XOR DX, DX                              ;读入数据到DX-AX
```

CMP DI, 0	;若第一次取位
JE LIMP	
MOV AX, CX	;若非第一次取位则把数记为之前取位得到的余数
LIMP:	
PUSH AX	
PUSH DX	;把数入栈
CMP DI, 0	
JE GETSIN	
MOV AX, BX	
XOR DX, DX	;把取位值读入到 DX-AX
MOV CX, 10	
DIV CX	
MOV BX, AX	;每次取位除 10
GETSIN:	
POP DX	
POP AX	;把数读出栈
DIV BX	;取位的位即商放在 AX，余数放在 DX
MOV CX, DX	;寄存余数
ADD AL, 30H	
MOV DL, AL	
CMP DL, 30H	
JE NUL	
CONT:	
MOV AH, 2	
INT 21H	;打印每一位
ADD DI, 1	;打印一次记位加一
CMP DI, 2	;取到最后一位
JE PRGE	
JMP PRINT	
PRGE:	
ADD CL, 30H	
MOV DL, CL	
MOV AH, 2	
INT 21H	;打印个位
ADD SI, 2	;一个数打印完成转到下一个
MOV DL, 0AH	
MOV AH, 2	
INT 21H	;换行
MOV DL, 0DH	
MOV AH, 2	
INT 21H	;回车
CMP SI, RANGE	
JE CLOSE	;若已读完则退出

JMP OUTPUT

CLOSE:

MOV BX, HANDLE

MOV AH, 3EH

INT 21H ;关闭文件

JNC ENDF

ERROR:

MOV DX, OFFSET ERRORINFO

MOV AH, 9

INT 21H

ENDF:

MOV AH, 4CH

INT 21H

NUL:

ADD DI, 1

CMP DI, 2

JE PRGE

JMP PRINT

主要思想是将数组 **ARRAY** 中的数用除法指令分解成单个单个数，然后再输出，一个数输出结束以后输出换行回车符。关闭文件的入口参数：**AH=3EH**，**BX**=文件句柄，出口参数：**CF=0**——关闭成功，否则，**AX**=错误号(06H)。

五.意见建议

在实验验收过程中，一开始输出结果为满 3 位输出，不满 3 位的前面补 0，助教要求将前面的 0 去掉以后，修正了代码，若除法指令的商为 0，则表明高位是 0，此后循环下标增加不输出任何 ASCII 码，直至遇到不为 0 的数再输出。

另外，在输出数字过程中，用栈输出数字较为方便，下次应注意栈输出的利用。

六.源代码

DATA SEGMENT

FILE DB '2.TXT', 0 ;文件名

BUF DB 0

ARRAY DW 1024 DUP (0) ;数据空间

RANGE DW 22

ERRORINFO DB 0AH, 'ERROR!', '\$' ;定义错误信息

HANDLE DW ?

DATA ENDS

STACK SEGMENT

```
DW 64 DUP (?)
STACK ENDS
```

CODE SEGMENT

```
ASSUME CS:CODE, DS:DATA, SS:STACK
```

```
START:
```

```
MOV AX, DATA
MOV DS, AX
```

```
MOV DX, OFFSET FILE
```

```
MOV AX, 3D00H
```

```
INT 21H ;打开文件
```

```
MOV HANDLE, AX ;转存文件代号
```

```
MOV BX, AX
```

```
MOV SI, 0 ;读入数下标
```

```
READFILE:
```

```
MOV CX, 1
```

```
LEA DX, BUF
```

```
MOV AH, 3FH
```

```
INT 21H ;读取一个字节
```

```
CMP BUF, 0DH
```

```
JE READFILE ;读到回车直接往下读
```

```
CMP BUF, 0AH
```

```
JE READNEW ;读到换行符开始读下一个数
```

```
MOV AX, ARRAY[SI]
```

```
MOV CX, 10
```

```
MUL CX
```

```
MOV ARRAY[SI], AX ;数基本值乘 10
```

```
MOV CL, BUF
```

```
SUB CL, 30H
```

```
XOR CH, CH ;将读入字节转成数字
```

```
ADD ARRAY[SI], CX ;加入到基本值
```

```
JMP READFILE ;继续读下一字节
```

```
READNEW:
```

```
ADD SI, 2 ;数下标加一
```

```
CMP SI, RANGE
```

```
JB READFILE ;若没读完则继续读
```

```
MOV DI, 0
```

```
BUBBLE:
```

```
MOV SI, 0
```

```
MOV AX, RANGE
```

```
SUB AX, DI
```

SUB AX, 2	
SORT:	
MOV BX, SI	
ADD BX, 2	
MOV CX, ARRAY[BX]	
CMP ARRAY[SI], CX	
JB RECY	;若相邻两数前者小于后者，跳转
MOV DX, ARRAY[SI]	;否则，交换两数位置
MOV CX, ARRAY[BX]	
MOV ARRAY[SI], CX	
MOV ARRAY[BX], DX	
RECY:	
ADD SI, 2	
CMP SI, AX	
JB SORT	
ADD DI, 2	
CMP DI, RANGE	
JB BUBBLE	
MOV SI, 0	;初始取数的数组下标
OUTPUT:	
MOV DI, 0	;记位
MOV BX, 100	;初始取值
PRINT:	
MOV AX, ARRAY[SI]	
XOR DX, DX	;读入数据到 DX-AX
CMP DI, 0	;若第一次取值
JE LIMP	
MOV AX, CX	;若非第一次取值则把数记为之前取值得到的余数
LIMP:	
PUSH AX	
PUSH DX	;把数入栈
CMP DI, 0	
JE GETSIN	
MOV AX, BX	
XOR DX, DX	;把取值读入到 DX-AX
MOV CX, 10	
DIV CX	
MOV BX, AX	;每次取值除 10
GETSIN:	
POP DX	
POP AX	;把数读出栈
DIV BX	;取位的位即商放在 AX，余数放在 DX

```

MOV CX, DX                ;寄存余数
ADD AL, 30H
MOV DL, AL
CMP DL,30H
JE NUL

CONT:
MOV AH, 2
INT 21H                   ;打印每一位
ADD DI, 1                 ;打印一次记位加一
CMP DI, 2                 ;取到最后一位
JE PRGE
JMP PRINT
PRGE:
ADD CL, 30H
MOV DL, CL
MOV AH, 2
INT 21H                   ;打印个位
ADD SI, 2                 ;一个数打印完成转到下一个
MOV DL, 0AH
MOV AH, 2
INT 21H                   ;换行
MOV DL, 0DH
MOV AH,2
INT 21H                   ;回车
CMP SI, RANGE
JE CLOSE                  ;若已读完则退出
JMP OUTPUT

CLOSE:
MOV BX, HANDLE
MOV AH, 3EH
INT 21H                   ;关闭文件
JNC ENDF

ERROR:
MOV DX, OFFSET ERRORINFO
MOV AH, 9
INT 21H

ENDF:
MOV AH, 4CH
INT 21H

```



```
NUL:  
ADD DI, 1  
CMP DI, 2  
JE PRGE  
JMP PRINT
```

```
CODE ENDS  
END START
```