**汇编语言与逆向技术实验报告**

**Lab3-bubble sort**

**学号：2110957 姓名：蒋薇 专业： 计算机科学与技术**

1. **实验内容**

编写汇编程序bubble.asm，功能是将Windows命令行输入的10个1万以内的十进制无符号整数，进行排序，然后输出在Windows命令行中。10个无符号整数之间用逗号","或者空格" "分割。

使用函数获得用户输入的十进制整数序列；

使用函数在Windows命令函中输出排好序的十进制整数序列。

使用ml和link程序将源代码编译、链接成可执行文件bubble\_sort.exe。

1. **实验步骤**

**实验设计报告**

|  |  |
| --- | --- |
| 实验题目 | 汇编实现整数数组的冒泡排序 |
| 实验目的 | 1熟悉汇编语言的整数数组；  2熟悉基址变址操作数、相对基址变址操作数；  3掌握排序算法的底层实现细节 |
| 实验内容 | 1实现十个整数的输入  2冒泡排序汇编代码  3实现十个整数的输出  4其他函数 |
| 实验步骤及结果 | 1数据段  DATAS SEGMENT  string\_1 DB 'Please input a numbers(0-65536):','$'  string\_2 DB 'ERROR: OVERFLOW! Please input again:','$'  string\_3 DB 'The array you have input is:',0ah,0dh,'$'  string\_4 DB 'After Sort the num is:',0ah,0dh,'$'  string\_5 DB ' ','$'  DATA DW 10 DUP(?)  DATAS ENDS  2堆栈段  STACKS SEGMENT  DW 256 dup(?)  STACKS ENDS  3代码段  CODES SEGMENT  ASSUME CS:CODES,DS:DATAS,SS:STACKS  输入函数子程序  ;---------输入函数（单数字输入）------------  Input PROC Near  push AX  push BX  push CX  push DX  ;---------输入提示--------------    MOV BX, 0  CLC  MOV DX, 0  ;----------输入数字--------------  Lp\_0:  MOV AH, 1  INT 21H  CMP AL, 20H ;回车  JE L\_CRLF    ;----- x belong to [0,9] ----------  SUB AL, 30H ; ASCII -> int  JL L\_ERROR  CMP AL, 9  JG L\_ERROR  ;------- string -> int -----------  MOV AH, 0 ;将 AL扩展成 AX  XCHG AX, BX ;保护 AX值  MOV CX, 10  MUL CX ; bx \*= 10  ADD AX , BX  JC L\_ERROR ; OVERFLOW处理  XCHG AX, BX  JMP Lp\_0  L\_ERROR:  MOV DX, 0  MOV BX, 0  CALL CRLF ; 换行  CALL ERROR ; 输出错误提示  JMP Lp\_0  L\_CRLF: ; 以换行作为一个数的结束标志  MOV DX, 0  MOV DATA[SI], BX ;  POP DX  POP CX  POP BX  POP AX  RET  Input ENDP  实现[冒泡排序](https://so.csdn.net/so/search?q=%E5%86%92%E6%B3%A1%E6%8E%92%E5%BA%8F&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/dinghanzhi5986/article/details/_blank)  ;---------Bubble\_sort--------------------  Bubble\_sort PROC NEAR  PUSH BX  PUSH DX  MOV SI,DI  LOOP1:  ADD SI,2  MOV BX,DATA[DI]  CMP BX,DATA[SI]  JA SWAP  JMP NEXT  SWAP:  MOV DX,DATA[SI]  MOV DATA[DI],DX  MOV DATA[SI],BX  NEXT:  CMP SI,18  JL LOOP1  POP DX  POP BX  RET  Bubble\_sort ENDP  ;----------Sort-----------  MOV CX, 9  MOV DI, 0  FOR1:  CALL Bubble\_sort  ADD DI, 2  LOOP FOR1  3.DOS输出到屏幕  CALL CRLF  MOV DX, OFFSET string\_4 ;'After Sort the num is:'  MOV AH, 9  INT 21H    MOV CX, 10  MOV DI, 0  FOR2:  CALL Print  CALL Space  ADD DI , 2  LOOP FOR2  CALL CRLF  Print PROC Near  PUSH AX  PUSH BX  PUSH CX  PUSH DX    MOV CX, 0  MOV BX, 10  MOV AX, DATA[DI]  LAST:  MOV DX, 0  DIV BX ; DIV商放AX，余数放入DX  PUSH DX  INC CX  CMP AX, 0  JNZ LAST  AGE:  POP DX  OR DX, 30H  MOV AH, 2  INT 21H  LOOP AGE  POP DX  POP CX  POP BX  POP AX  RET  Print ENDP  其他函数  ;----换行子函数（一个数输入完毕）-------  CRLF PROC Near  push AX  push DX  MOV DL, 0ah  MOV AH, 2  INT 21H  pop DX  pop AX  RET  CRLF ENDP  ;---------空格-----------  Space PROC Near  push AX  push DX  MOV DX, OFFSET string\_5 ;' '  MOV AH, 9  INT 21H  pop DX  pop AX  RET  Space ENDP  ;----------错误提示-------------  ERROR PROC Near  push BX  push DX  MOV DX, OFFSET string\_2 ; ERROR: OVERFLOW! Please input again:  MOV AH, 9  INT 21H  pop DX  pop BX  RET  ERROR ENDP |
| 方法说明 | string\_1：输入范围提示  string\_2:输入错误提示  string\_3:输出原数组提示  string\_4：输出排序后数组提示  string\_5:空格符  DATA:缓冲区数组  时间复杂度 O(n2)O(n^2)O(n2) 需要两层循环，为了提高代码的可读性，将内层的循环写成一个子程序每次调用，内层循环每次从头比到尾，遇到比它小的交换。因为是字操作数，所以循环的下标到18为结束条件。  输出DATA内的数字，每次输出一个数字然后在输出一个空格  Print函数:  利用DIV函数的特点——每次除10的商放在AX， 余数放入DX  并利用栈的 FILO（First in Last Out）的特点 |
| 实验收获 | 除了已给我们写好的IO函数,我们可以自己用代码实现 |

;-----数据段------------

DATAS SEGMENT

string\_1 DB 'Please input 10 numbers(0-65536):','$'

string\_2 DB 'ERROR: OVERFLOW! Please input again:','$'

string\_3 DB 'The array you have input is:',0ah,0dh,'$'

string\_4 DB 'After Sort the num is:',0ah,0dh,'$'

string\_5 DB ' ','$'

DATA DW 10 DUP(?)

DATAS ENDS

;-----堆栈段------------

STACKS SEGMENT

DW 256 dup(?)

STACKS ENDS

;-----代码段------------

CODES SEGMENT

ASSUME CS:CODES,DS:DATAS,SS:STACKS

;-----------程序开始------------

START:

MOV AX,DATAS

MOV DS,AX

MOV SI, 0 ;指针初始化

MOV CX, 10 ;循环次数

;---------Input----------

MOV DX, OFFSET string\_1 ;Please input 10 numbers(0-65536)

MOV AH, 9

INT 21H

Lp:

CALL Input

ADD SI, 2

Loop Lp

;--------结束输入，换行---------------

CALL CRLF

MOV DX, OFFSET string\_3 ;'The array you have input is:'

MOV AH, 9 ;首地址 DS:DX

INT 21H

;-------输出 ----------------

MOV CX, 10

MOV DI, 0

Again:

CALL Print

CALL Space

ADD DI , 2

Loop Again

;/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

;----------Sort-----------

MOV CX, 9

MOV DI, 0

FOR1:

CALL Sort

ADD DI, 2

LOOP FOR1

CALL CRLF

MOV DX, OFFSET string\_4 ;'After Sort the num is:'

MOV AH, 9

INT 21H

MOV CX, 10

MOV DI, 0

FOR2:

CALL Print

CALL Space

ADD DI , 2

LOOP FOR2

CALL CRLF

;-------求和输出---------------------

MOV DX, OFFSET massege;

MOV AH, 9

INT 21H

CALL Get\_sum

MOV DI, 0

CALL Print

EXIT:

MOV AH, 4CH

INT 21H

;/\*\*\*\*\*\*\*\*\*\*\*\*子程序调用\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

;---------输入函数（单数字输入）------------

Input PROC Near

push AX

push BX

push CX

push DX

MOV BX, 0

CLC

MOV DX, 0

;----------输入数字--------------

Lp\_0:

MOV AH, 1

INT 21H

CMP AL, 20H ;空格

JE L\_CRLF

;----- x belong to [0,9] ----------

SUB AL, 30H ; ASCII -> int

JL L\_ERROR

CMP AL, 9

JG L\_ERROR

;------- string -> int -----------

MOV AH, 0 ;将 AL扩展成 AX

XCHG AX, BX ;保护 AX值

MOV CX, 10

MUL CX ; bx \*= 10

ADD AX , BX

JC L\_ERROR ; OVERFLOW处理

XCHG AX, BX

JMP Lp\_0

L\_ERROR:

MOV DX, 0

MOV BX, 0

CALL CRLF ; 换行

CALL ERROR ; 输出错误提示

JMP Lp\_0

L\_CRLF: ; 以换行作为一个数的结束标志

MOV DX, 0

MOV DATA[SI], BX ;

POP DX

POP CX

POP BX

POP AX

RET

Input ENDP

;----换行子函数（一个数输入完毕）-------

CRLF PROC Near

push AX

push DX

MOV DL, 0ah

MOV AH, 2

INT 21H

pop DX

pop AX

RET

CRLF ENDP

;---------空格-----------

Space PROC Near

push AX

push DX

MOV DX, OFFSET string\_5 ;' '

MOV AH, 9

INT 21H

pop DX

pop AX

RET

Space ENDP

;----------错误提示-------------

ERROR PROC Near

push BX

push DX

MOV DX, OFFSET string\_2 ; ERROR: OVERFLOW! Please input again:

MOV AH, 9

INT 21H

pop DX

pop BX

RET

ERROR ENDP

;---------输出函数（单数字输出）-------------

Print PROC Near

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV CX, 0

MOV BX, 10

MOV AX, DATA[DI]

LAST:

MOV DX, 0

DIV BX ; DIV商放AX，余数放入DX

PUSH DX

INC CX

CMP AX, 0

JNZ LAST

AGE:

POP DX

OR DX, 30H

MOV AH, 2

INT 21H

LOOP AGE

POP DX

POP CX

POP BX

POP AX

RET

Print ENDP

;---------SORT---------------------

SORT PROC NEAR

PUSH BX

PUSH DX

MOV SI,DI

LOOP1:

ADD SI,2

MOV BX,DATA[DI]

CMP BX,DATA[SI]

JA CHANGE

JMP NEXT

CHANGE:

MOV DX,DATA[SI]

MOV DATA[DI],DX

MOV DATA[SI],BX

NEXT:

CMP SI,18

JL LOOP1

POP DX

POP BX

RET

SORT ENDP

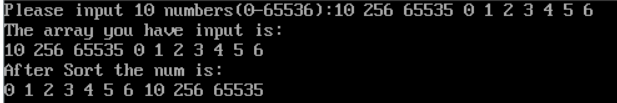
CODES ENDS

END START

实验过程中的详细步骤，关键代码与截图如上。

三：实验截图

实验成功的截图



四：实验心得

mov <mem>,<mem> ;error

mov eax,var1

xchg eax,var2

mov var1,eax

Masm32使用方括号表示内存寻址操作，通常直接内存寻址不使用中括号，涉及算术表达式时，使用中括号；

OFFSET操作符返回数据标号偏移地址

PTR操作符可以重载操作数声明的默认尺寸

用寄存器作为指针并控制该寄存器的值称为间接寻址

任何一个32位通用寄存器加上[]构成一个简介操作数

变址操作数把常量和寄存器相加得到有效地址

基址变址操作数把两个寄存器的值相加，得到一个偏移地址

相对基址变址操作数把偏移、基址、变址、可选比例因子组合产生偏移地址