**汇编语言与逆向技术实验报告**

**Lab3-整数数组的冒泡排序**

**学号：2113997 姓名：齐明杰 专业：信息安全**

1. **实验内容**

排序算法中，汇编语言的基址变址寻址方式和相对基址变址寻址方式起到了重要的作用。

基址变址（base-index）操作数把两个寄存器的值相加，得到一个偏移地址。两个寄存器分别称为基址寄存器（base）和变址寄存器（index）。格式为[base + index]，例如mov eax, [ebx + esi]。在例子中，ebx是基址寄存器，esi是变址寄存器。基址寄存器和变址寄存器可以使用任意的32位通用寄存器。

相对基址变址（based-indexed with displacement）操作数把偏移、基址、变址以及可选的比例因子组合起来，产生一个偏移地址。常见的两种格式为：[base + index + displacement]和displacement[base + index]，例子如下：

table dword 10h, 20h, 30h, 40h

row\_size = ($ - table)

dword 50h, 60h, 70h, 80h

dword 90h, 0a0h, 0b0h, 0c0h

mov ebx, row\_size

mov esi, 2

mov eax, table[ebx + esi \* 4]

table是一个二维数组，共3行4列。ebx是基址寄存器，相当于二维数组的行索引，esi是变址寄存器，相当于二维数组的列索引。

冒泡排序算法（Bubble Sort）的过程是从位置0和1开始比较每对数据的值，如果两个数据的顺序不对，就进行交换。如果一遍处理完之后，数组没有排好序，就开始下一次循环。在最多完成n-1次循环后，数组排序完成。

本次实验要求编写汇编程序bubble\_sort.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StdIn函数获得用户输入的十进制整数序列。StdIn函数的定义在\masm32\include\masm32.inc，库文件是\masm32\lib\masm32.lib。StdIn函数的定义“StdIn PROTO :DWORD,:DWORD”，有两个参数，第一个是内存存储空间的起始地址，第二个是内存存储空间的大小。

使用StdOut函数在Windows命令函中输出排好序的十进制整数序列。StdOut函数的定义在\masm32\include\masm32.inc，库文件是\masm32\lib\masm32.lib。StdOut函数的定义“StdOut PROTO :DWORD”，只有一个参数，是内存存储空间的起始地址。

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. **源代码和注释**

**.386 ;允许汇编80386处理器的非特权指令，禁用其后处理器引入的汇编指令**

**.model flat, stdcall ;.model用于初始化程序的内存模式**

**option casemap :none ;大小写敏感**

**include \masm32\include\windows.inc**

**include \masm32\include\masm32.inc**

**include \masm32\include\kernel32.inc**

**includelib \masm32\lib\kernel32.lib**

**includelib \masm32\lib\masm32.lib ;函数的常量和链接库声明**

**.stack 4096 ;栈空间声明**

**.data**

**str1 byte "Please input ten unsigned int number: ",0 ;提示输入数组(以空格分割)**

**str2 byte "The result of bubble sort is: ",0 ;提示输出数组(以空格分割)**

**inputstr byte 80 dup(0) ;存储输入的字符串**

**pArray dword 15 dup(0) ;存储字符串转化成的各个数字**

**outputstr byte 80 dup(0) ;存储输出的字符串**

**const10 dword 10 ;常数10**

**count dword 1 ;数组长度**

**count2 dword 0 ;用于保存ecx**

**.code**

**main proc ;main过程**

**invoke StdOut,addr str1 ;输出提示**

**invoke StdIn,addr inputstr,60 ;输入数组**

**call strtonum ;输入字符串转化为数组**

**call BubbleSort ;数组冒泡排序**

**call numtostr ;排序后的数组重新转化为可输出的字符串**

**invoke StdOut,addr str2 ;输出提示**

**invoke StdOut,addr outputstr ;输出排序后数组**

**invoke ExitProcess,0**

**main endp**

**strtonum proc uses esi eax ebx ecx ;输入字符串转化为数组**

**mov esi,0**

**mov ebx,0**

**mov ecx,0**

**L1:**

**mov bl,[inputstr+esi] ;以esi为变址寄存器读取输入字符串**

**cmp bl,32 ;判断是否读取到空格(ASCII码值为32)**

**jne L2**

**inc count**

**add ecx,4 ;是空格则count+1,同时ecx+4**

**inc esi**

**mov bl,[inputstr+esi] ;略过空格**

**L2:**

**sub bl,48 ;字符转化为数字**

**mov eax,[pArray+ecx] ;读取数字**

**mul const10 ;eax值乘10**

**add eax,ebx ;加上刚读取的数字**

**mov [pArray+ecx],eax ;存回**

**inc esi**

**cmp [inputstr+esi],0 ;读取到串尾0停止**

**jne L1**

**ret**

**strtonum endp**

**BubbleSort proc uses eax ecx esi ;数组冒泡排序**

**mov ecx,count**

**dec ecx ;令ecx等于数组长度-1**

**L3:**

**push ecx ;入栈,保存外层ecx**

**mov esi,0**

**L4:**

**mov eax,[pArray+esi]**

**cmp eax,[pArray+esi+4] ;比较数**

**jbe L5**

**xchg eax,[pArray+esi+4]**

**mov [pArray+esi],eax ;若前一个数比后一个大则交换**

**L5:**

**add esi,4**

**loop L4 ;内层循环**

**pop ecx ;出栈,恢复外层ecx**

**loop L3 ;外层循环**

**ret**

**BubbleSort endp**

**numtostr proc ;排序后的数组重新转化为可输出的字符串(难)**

**思路：遍历排序完的数字，每个数字对10取余数倒排得到字符串，倒排利用出栈入栈即可实现。**

**mov ecx,count**

**dec ecx**

**mov esi,0**

**L10:**

**add esi,4**

**loop L10 ;使esi=4\*(count-1),即从最后一个数字倒序遍历**

**mov edi,0**

**mov edx,0**

**mov ecx,count**

**L6:**

**mov count2,ecx ;保存ecx**

**mov eax,[pArray+esi] ;获取数字**

**sub esi,4 ;移动到前一个数字**

**L7:**

**div const10 ;除法指令,被除数为edx:eax,eax存商,edx存余数**

**mov ecx,edx**

**mov edx,0**

**add ecx,48 ;数字转化为字符**

**push ecx ;字符入栈**

**inc edi**

**cmp eax,0**

**jne L7**

**push 32 ;商为0则入栈一个空格**

**inc edi**

**mov ecx,count2 ;恢复ecx**

**loop L6**

**mov ecx,edi ;循环次数为输出字符长度**

**mov esi,0**

**L8:**

**pop eax ;循环出栈**

**mov [outputstr+esi],al ;出栈字符保存到输出字符串中**

**inc esi**

**loop L8**

**mov [outputstr+esi],0 ;补充串尾0**

**ret**

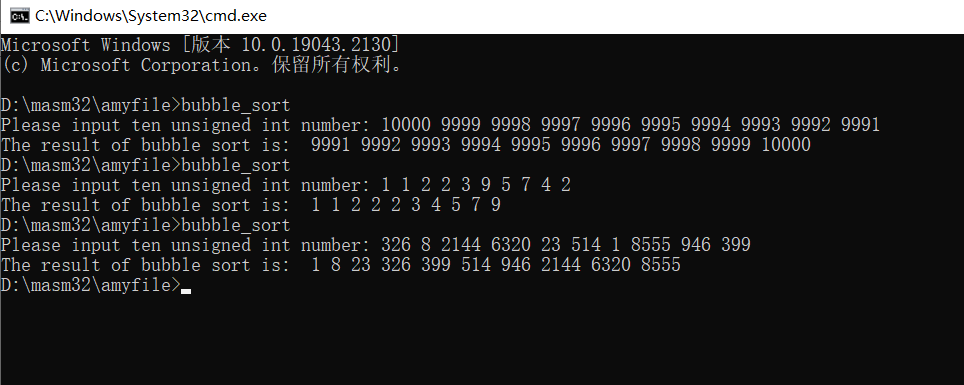
**numtostr endp**

**end main**

1. **实验测试和截图**

编译：使用指令ml -c -coff bubble\_sort.asm生成obj文件

链接：使用指令link -subsystem:console bubble\_sort.obj 生成exe文件



说明：

三组数据分别采用最大值法，最小值法，随机法测试，输入输出数字均以空格分割，均成功排序输出。

1. **知识点总结**

1、交换指令xchg。xchg指令交换两个操作数的内容，但不能直接交换两个内存的内容，可用于数组内的交换。

2、循环指令loop。Loop以ecx为循环计数器进行循环，可用于遍历数组

3、判断指令cmp，条件跳转指令，无条件跳转。

Cmp指令通过修改cpu的标志位达到比较的目的，通常和je,jne,ja,jb,jg,jl等条件跳转配合使用，以及无条件跳转jmp指令。

4、间接寻址—变址操作数、基址变址操作数。

形如[eax + array1]的操作数便称为变址操作数，最常用于遍历数据。

形如[ebx+esi]的操作数称为基址变址操作数，可用于访问二维数组

5、dup操作符用于声明大型数组，包括需要初始化的数组和不要初始化的数组。

6、寄存器esi和edi是常用的变址寄存器。他们类似于指针，对字符操作非常有用。

7、在处理dword类型的数组时，偏移量是以4为单位，而不是1，因为一个带符号双字节占用4个字节的内存空间

8、操作符lengthof用于计算数组的元素个数，操作符sizeof用于计算数组占用字节空间

**五、 实验心得**

通过这次实验，我学会了使用冒泡排序来对数组进行排序，学会了将数字数据转化为字符数据，然后使用stdout进行输出，也学会了各种跳转指令的应用。