```
                        ;Bee Cha
                        ;CSCI 112, Lab 7
                        ;Prompt user to enter 10 numbers
                        ;The program will convert the numbers to DWORD using atod
                        ;Then it will compare each DWORD element and sort them in ascending
order
                        ;Output the sorted array along with the original numbers

                        .586
                        .MODEL FLAT
                        INCLUDE io.h
                       C ; IO.H -- header file for I/O macros (listing suppressed)
                       C .NOLIST      ; turn off listing
                       C .LIST        ; begin listing
                       C
                        .STACK 4096

 00000000               .DATA
 00000000 45 6E 74 65 72  prompt1 BYTE "Enter ten scores (separated by a space each): ",0
          20 74 65 6E 20
          73 63 6F 72 65
          73 20 28 73 65
          70 61 72 61 74
          65 64 20 62 79
          20 61 20 73 70
          61 63 65 20 65
          61 63 68 29 3A
          20 00
 0000002F 6F 75 74 70 75  prompt2 BYTE "output is: ",0
          74 20 69 73 3A
          20 00
 0000003B 4F 72 69 67 69  label1 BYTE "Original Scores:", 0 ;16 characters
          6E 61 6C 20 53
          63 6F 72 65 73
          3A 00
 0000004C 53 6F 72 74 65  label2 BYTE "Sorted Scores:", 0 ;14 characters
          64 20 53 63 6F
          72 65 73 3A 00
 0000005B 52 65 73 75 6C  label3 BYTE "Results", 0
          74 73 00
 00000063 00              scoreString byte 40 dup(?),0; //for 10 scores - make enough room
 0000008C 00              sortString byte 140 dup(?),0;
 00000119 00              outstr BYTE 400 dup(?), 0
 000002AA                 scoreArray DWORD 10 dup (?)
 000002D2 00              temp byte 11 dup(?),0
 000002DE 00              tempx byte 11 dup(" "),0;     //or,..dup(20H); 20H=space
 000002EA 00              temp2 byte 11 dup(?),0
 000002F6 00              tempstr byte 11 dup(?),0
 00000302 00000000        count1 dword 0
 00000306 FFFFFFFF        count2 dword -1

 00000000               .CODE
 00000000               _MainProc PROC
                                input prompt1, scoreString, 40; //prompt user to enter 10
scores
 0000001E  8D 1D 00000063 R         lea ebx, scoreString
```

```
 00000024  FF 05 00000302 R  outerLoop: inc count1; //outer loop counter++

                                    ;-----To put space, 20h, into the string-----;
 0000002A  8D 35 000002DE R          lea esi, tempx; //flush temp string before using
 00000030  8D 3D 000002D2 R          lea edi, temp
 00000036  FC                        cld
 00000037  B9 0000000B                      mov ecx, 11
 0000003C  F3/ A4                    rep movsb          ;put 11 20h into memory of &temp

 0000003E  8D 15 000002D2 R          lea edx, temp;
 00000044  80 3B 20           innerLoop: cmp byte ptr[ebx], 20h; //if ending mark(space), done,
[ebx] is where the array is
 00000047  74 0D                     je done1;
 00000049  80 3B 00                  cmp byte ptr[ebx], 00h;     //else if null char, also done,
[ebx] is where the array is
 0000004C  74 08                     je done1;
 0000004E  8A 03                     mov AL, byte ptr[ebx]; //otherwise, get 1 byte from input
string
 00000050  88 02                     mov [edx], AL;         //and move it to temp

 00000052  43                        inc ebx;       //to next byte in input string
 00000053  42                        inc edx;       //to next byte in temp string
 00000054  EB EE                     jmp innerLoop; //inner loop (temp <- one score)

 00000056                     done1: ;output prompt2, temp; test display of temp
                                    atod temp; //eax <- temp

 00000065  8B 0D 00000302 R          mov ecx, count1;
 0000006B  49                        dec ecx;       //counter:1 -> array index:0
 0000006C  6B C9 04                  imul ecx, 4;   //array ele size = 4 bytes
 0000006F  89 81 000002AA R          mov scoreArray[ecx], eax; //store one score in array

 00000075  43                        inc ebx;       //skip the end mark(space) in the input
string
 00000076  83 3D 00000302 R          cmp count1, 10; //loop 10 times
           0A
 0000007D  7C A5                     jnge outerLoop

                                    ;dtoa temp2, scoreArray[0]; or, scorearray+0; //testing
display
                                    ;output  prompt2, temp2


                              ;-----First, you must convert each ASCII element into DWORD-----;

                                    ;get each element from the string and convert it to DWORD
with atod
                                    ;each element is separated by a space, 20h

 0000007F  8D 1D 00000063 R                lea ebx, scoreString
 00000085                       outLoop:
 00000085  FF 05 00000306 R                inc count2          ;count2++

                                    ;flush the contents in temp2 before using it again
 0000008B  8D 35 000002DE R                lea esi, tempx
 00000091  8D 3D 000002EA R                lea edi, temp2
 00000097  FC                              cld
 00000098  B9 0000000B                            mov ecx, 11
```

```
 0000009D  F3/ A4                          rep movsb

 0000009F  8D 15 000002EA R                lea    edx, temp2                 ;load the effective
address of temp2 into edx register
 000000A5                    inLoop:
 000000A5  80 3B 20                        cmp    byte ptr [ebx], 20h  ;compare if and when it
will hit a space character in the string array
 000000A8  74 0D                           je finish1
 000000AA  80 3B 00                        cmp    byte ptr [ebx], 00h ;compare if and when it
will hit a null character in the string array
 000000AD  74 08                           je finish1
 000000AF  8A 03                           mov    al, byte ptr [ebx]   ;otherwise get 1 byte from
the string
 000000B1  88 02                           mov [edx], al              ;move it to temp2
                                           ;then grab the next byte
 000000B3  43                              inc    ebx
 000000B4  42                              inc edx
 000000B5  EB EE                           jmp inLoop

 000000B7                    finish1:
                                           atod temp2                         ;convert ascii
value into DWORD type into eax register
 000000C6  8B 0D 00000306 R                mov ecx, count2
 000000CC  6B C9 04                        imul ecx, 4                        ;each DWORD size is
4 bytes
 000000CF  89 81 000002AA R                mov    scoreArray[ecx], eax ;store the contents into
scoreArray[0], scoreArray[4], and so on

 000000D5  43                              inc    ebx                              ;skip the
initial 20h, go to next character in the string
 000000D6  83 3D 00000306 R                cmp    count2, 10
           0A
 000000DD  7C A6                           jnge   outLoop                    ;loop 10 times for
10 elements


                           ;-----Now perform the sorting algorithm-----;


                                  ;Reset my counters
 000000DF  C7 05 00000302 R               mov    count1, -1    ;count for outerloop
           FFFFFFFF
 000000E9  C7 05 00000306 R               mov    count2, 0     ;count for innerloop
           00000000

                                  ;-----Selection Sort Algorithm-----;
                                  ;outer loop: for (i = 0; i < 10, i++)
                                         ;inner loop: for (j = i+1, j < 10, j++)
                                                ;if a[j] < a[i]
                                                       ;swap (a[i], a[j])

                                  ;scoreArray = [123, 45, 6, 777, 8, 20, 15, 35, 100, 50]

 000000F3  8D 1D 000002AA R               lea ebx, scoreArray        ;load starting address of
scoreArray

 000000F9                    sortOut:
 000000F9  FF 05 00000302 R               inc    count1
 000000FF  8B 0D 00000302 R               mov    ecx, count1
 00000105  6B C9 04                       imul ecx, 4
```

```
00000108  8B 81 000002AA R              mov eax, dword ptr scoreArray[ecx]
0000010E  8B 3D 00000302 R              mov edi, count1
00000114  89 3D 00000306 R              mov count2, edi

0000011A                    sortIn:
0000011A  FF 05 00000306 R              inc count2
00000120  8B 35 00000306 R              mov    esi, count2
00000126  6B F6 04                      imul esi, 4
00000129  8B 96 000002AA R              mov    edx, dword ptr scoreArray[esi]
0000012F  3B D0                         cmp edx, eax
                                        ;if edx is less than eax, exchange
00000131  7C 02                         jl swap
00000133  EB 0D                         jmp skip
00000135                    swap:
00000135  92                            xchg edx, eax
                                        ;after swapping, move it back into the array
00000136  89 81 000002AA R              mov scoreArray[ecx], eax
0000013C  89 96 000002AA R              mov scoreArray[esi], edx

00000142                    skip:
00000142  83 3D 00000306 R              cmp    count2, 8
          08
00000149  7E CF                         jng           sortIn

0000014B  83 3D 00000302 R              cmp count1, 7
          07
00000152  7E A5                         jng           sortOut


                            ;-----Append the sorted array to the a string and output
result-----;

                                   ;First, you must convert the sorted array into ASCII value
using dtoa

00000154  C7 05 00000302 R              mov           count1, -1
          FFFFFFFF
0000015E  C7 05 00000306 R              mov           count2, -1
          FFFFFFFF

00000168                    stringCPY:
                                             ;-----To put space, 20h, into the string-----;
00000168  8D 35 000002DE R         lea esi, tempx; //flush temp string before using
0000016E  8D 3D 000002F6 R         lea edi, tempstr
00000174  FC                       cld
00000175  B9 0000000B                    mov ecx, 11
0000017A  F3/ A4                   rep movsb           ;put 11 20h into memory of &tempstr


0000017C  FF 05 00000302 R              inc           count1
00000182  FF 05 00000306 R              inc           count2
00000188  8B 1D 00000302 R              mov           ebx, count1
0000018E  6B DB 04                      imul   ebx, 4
00000191  8B 83 000002AA R              mov           eax, scoreArray[ebx]
                                        dtoa   tempstr, eax

000001AF  8B 15 00000306 R              mov           edx, count2
000001B5  6B D2 0B                      imul   edx, 11
000001B8  8D BA 0000008C R              lea           edi, sortString[edx]
000001BE  8D 35 000002F6 R              lea           esi, tempstr
```

```
 000001C4  FC                            cld
 000001C5  B9 0000000B                          mov           ecx, 11
 000001CA  F3/ A4                        rep           movsb

 000001CC  83 3D 00000302 R              cmp           count1, 10
           0A
 000001D3  7C 93                         jl            stringCPY


                                         ;-----Append to String-----;

 000001D5  8D 35 0000003B R              lea           esi, label1
 000001DB  8D 3D 00000119 R              lea           edi, outstr
 000001E1  FC                            cld
 000001E2  B9 00000010                          mov           ecx, 16
 000001E7  F3/ A4                        rep           movsb

 000001E9  8D 35 00000063 R              lea           esi, scoreString
 000001EF  8D 3D 00000129 R              lea           edi, outstr+16
 000001F5  FC                            cld
 000001F6  B9 0000001E                          mov           ecx, 30
 000001FB  F3/ A4                        rep           movsb

 000001FD  C6 05 00000147 R              mov           outstr+46, 0dh
           0D

 00000204  8D 35 0000004C R              lea           esi, label2
 0000020A  8D 3D 00000148 R              lea           edi, outstr+47
 00000210  FC                            cld
 00000211  B9 0000000E                          mov           ecx, 14
 00000216  F3/ A4                        rep           movsb

 00000218  8D 35 0000008C R              lea           esi, sortString
 0000021E  8D 3D 00000156 R              lea           edi, outstr+61
 00000224  FC                            cld
 00000225  B9 00000078                          mov           ecx, 120
 0000022A  F3/ A4                        rep           movsb

                                         output label3, outstr

 00000245  B8 00000000                    mov     eax, 0
 0000024A  C3                        ret
 0000024B                    _MainProc ENDP
                             END
```

```
Microsoft (R) Macro Assembler Version 12.00.30501.0        03/19/15 10:13:06
Lab7-test.asm                                            Symbols 2 - 1



Macros:

                N a m e                 Type

atod . . . . . . . . . . . . . .  Proc
atow . . . . . . . . . . . . . .  Proc
dtoa . . . . . . . . . . . . . .  Proc
input  . . . . . . . . . . . . .  Proc
output . . . . . . . . . . . . .  Proc
```

```
wtoa . . . . . .  Proc


Segments and Groups:

                N a m e                 Size     Length   Align   Combine Class

FLAT . . . . . .  GROUP
STACK  . . . . . .  32 Bit  00001000 Para         Stack         'STACK'
_DATA  . . . . . .  32 Bit  0000030A Para         Public  'DATA'
_TEXT  . . . . . .  32 Bit  0000024B Para         Public  'CODE'


Procedures, parameters, and locals:

                N a m e                 Type     Value    Attr

_MainProc  . . . . . .  P Near  00000000 _TEXT      Length= 0000024B Public
  outerLoop  . . . . . .  L Near  00000024 _TEXT
  innerLoop  . . . . . .  L Near  00000044 _TEXT
  done1  . . . . . .  L Near  00000056 _TEXT
  outLoop  . . . . . .  L Near  00000085 _TEXT
  inLoop . . . . . .  L Near  000000A5 _TEXT
  finish1  . . . . . .  L Near  000000B7 _TEXT
  sortOut  . . . . . .  L Near  000000F9 _TEXT
  sortIn . . . . . .  L Near  0000011A _TEXT
  swap . . . . . .  L Near  00000135 _TEXT
  skip . . . . . .  L Near  00000142 _TEXT
  stringCPY  . . . . . .  L Near  00000168 _TEXT


Symbols:

                N a m e                 Type     Value    Attr

@CodeSize  . . . . . .  Number  00000000h
@DataSize  . . . . . .  Number  00000000h
@Interface . . . . . .  Number  00000000h
@Model . . . . . .  Number  00000007h
@code  . . . . . .  Text           _TEXT
@data  . . . . . .  Text           FLAT
@fardata?  . . . . . .  Text           FLAT
@fardata . . . . . .  Text           FLAT
@stack . . . . . .  Text           FLAT
_getInput  . . . . . .  L Near  00000000 FLAT       External
_showOutput  . . . . . .  L Near  00000000 FLAT       External
atodproc . . . . . .  L Near  00000000 FLAT       External
atowproc . . . . . .  L Near  00000000 FLAT       External
count1 . . . . . .  DWord   00000302 _DATA
count2 . . . . . .  DWord   00000306 _DATA
dtoaproc . . . . . .  L Near  00000000 FLAT       External
label1 . . . . . .  Byte    0000003B _DATA
label2 . . . . . .  Byte    0000004C _DATA
label3 . . . . . .  Byte    0000005B _DATA
outstr . . . . . .  Byte    00000119 _DATA
prompt1  . . . . . .  Byte    00000000 _DATA
prompt2  . . . . . .  Byte    0000002F _DATA
scoreArray . . . . . .  DWord   000002AA _DATA
scoreString  . . . . . .  Byte    00000063 _DATA
sortString . . . . . .  Byte    0000008C _DATA
```

```
temp2  . . . . . .  Byte    000002EA _DATA
tempstr  . . . . . .  Byte    000002F6 _DATA
tempx  . . . . . .  Byte    000002DE _DATA
temp . . . . . .  Byte    000002D2 _DATA
wtoaproc . . . . . .  L Near  00000000 FLAT      External

          0 Warnings
          0 Errors
```