Marco Martinez

CISP 310

Assignment 3b

**Hierarchy Chart**

3.0 main

3.1 BubbleSort(in pArray as Array of Integers,in/out pIndex as Array of Integers,in Count as Integer)

3.2 LinearSearch(in pArray as array of integers,in pIndex as array of integers,in Count as integer,in key as integer)

3.3 DisplayArray(in pArray as array of integers,in pIndex as array of integers,in Count as integer)

3.4 DisplaySearchResult(in key as integer,in location as integer)

**Pseudo Code**

*Main Module*

Begin

DEFINE BEGIN\_DEFINE as String " SEARCH AND SORT TEST ",0

DEFINE BORDER\_DEFINE as String "===============================================================================================",0

DECLARE array1 as Array of Integers {40,-10,400,20,-300,12,10,0}

DECLARE index1 as Array of Integers {0,1,2,3,4,5,6,7}

DECLARE length1 as Integer LENGTHOF array1

DECLARE array2 as Array of Integers {0,-10,-20,-30,50,100,200,300,-100,-80,1000,2000,-5000,60,70,550,-550,-300,-900,1010,2300}

DECLARE index2 as Array of Integers {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20}

DECLARE length2 as Integer LENGTHOF array2

DECLARE array3 as Array of Integers {50,40,30,20,10}

DECLARE index3 as Array of Integers {0,1,2,3,4}

DECLARE length3 as LENGTHOF array3

DECLARE key1 as Integer -10

DECLARE key2 as Integer 1000

DECLARE key3 as Integer 0

DECLARE location1 as Integer

DECLARE location2 as Integer

DECLARE location3 as Integer

DECLARE msg1 as String "Unsorted array: "

DECLARE msg2 as String "Sorted array: "

DECLARE msg3 as String "Search found the value "

DECLARE msg4 as String " at position "

Set ecx as 0

Do

Save reg

Call crlf

Set edx as OFFSET BEGIN\_DEFINE

Set ecx as ecx + 1

Set eax as ecx

Call writeString

Call writeInt

Call crlf

Load reg

Save edx

Set edx as OFFSET BORDER\_DEFINE

Call writeString

Call crlf

Load edx

Save reg

Set edx as OFFSET msg1

Call writeString

Call DisplayArray(in ADDR array1,in ADDR index1,in length1)

Call BubbleSort(in ADDR array1,in/out ADDR index1,in length1)

Set edx as OFFSET msg2

Call writeString

Call DisplayArray(in ADDR array1,in ADDR index1,in length1)

Load reg

Save reg

Call LinearSearch(in ADDR array1,in ADDR index1,in length1,in key1)

Set edx as OFFSET msg3

Call writeString

Set edx as OFFSET msg4

Set location1 as eax

Call DisplaySearchResult(key1,location1)

Call crlf

Load reg

Save reg

Set edx as OFFSET msg1

Call writeString

Call DisplayArray(in ADDR array2,in ADDR index2,in length2)

Call BubbleSort(in ADDR array2,in/out ADDR index2,in length2

Set edx as OFFSET msg2

Call writeString

Call DisplayArray(in ADDR array2,in ADDR index2,in length2)

Load reg

Save reg

Call LinearSearch(in ADDR array2,in ADDR index2,in length2,in key2)

Set edx as OFFSET msg3

Call writeString

Set edx as OFFSET msg4

Set location2 as eax

Call DisplaySearchResult(in key2,in location2)

Call crlf

Load reg

Save reg

Set edx as OFFSET msg2

Call writeString

Call DisplayArray(in ADDR array3,in ADDR index3,in length3)

Call BubbleSort(in ADDR array3,in/out ADDR index3,in length3)

Set edx as OFFSET msg2

Call writeString

Call DisplayArray(in ADDR array3,in ADDR index3,in length3)

Load reg

Save reg

Call LinearSearch(in ADDR array3,in ADDR index3,in length3,in key3)

Set edx as OFFSET msg3

Call writeString

Set edx as OFFSET msg4

Set location3 as eax

Call DisplaySearchResult(in key3,in location3)

Call crlf

Call crlf

Load reg

Set key1 as key1 + 10

Set key2 as key2 + 10

Set key3 as key3 + 10

Set ecx as ecx + 1

While (ecx < 3)

*End main*

BubbleSort(in pArray as Array of Integers,in/out pIndex as Array of Integers,in Count as Integer)

*Begin*

DECLARE boolean swap

Set swap as false

Set ecx as Count

Set ecx as ecx - 1

L1:

Save ecx

Set edi as pIndex

Set ebx as [edi]

Set ebx as ebx \* 4

L2:

Set esi as pArray

Set esi as esi + ebx

Set eax as [esi]

Set esi as pArray

Set ebx as [edi+4]

Set ebx as ebx \* 4

Set esi as pArray

Set esi as esi + ebx

If ([esi]>eax)

Set ebx as [edi+4]

Exchange ebx and [edi]

Set [edi+4] as ebx

Set swap as true

End If

Set edi as edi + 4

Set ebx as [edi]

Set ebx as ebx \* 4

While (ecx < Count-1)

If(swap)

*Return*

End If

Load ecx

While (ecx < Count)

*Return*

LinearSearch(in pArray as array of integers,in pIndex as array of integers,in Count as integer,in key as integer)

*Begin*

Set eax as 0

Set edx as 0

Set ecx as Count

Set edi as pIndex

Set ebx as [edi]

Set ebx as ebx \* 2

Do

Set esi as pArray

Set esi as esi + ebx

Set eax as [esi]

If (eax == key)

Set eax as edx

Return

End If

Set edx as edx + 1

Set edi as edi + 4

Set ebx as [edi]

Set ebx as ebx \* 4

Set ecx as ecx - 1

While (ecx>0)

Set eax as -1

*Return*

DisplayArray(in pArray as array of integers,in pIndex as array of integers,in Count as integer)

*Begin*

Set ecx as Count

Set edi as pIndex

Set ebx as [edi]

Set ebx as ebx \* 4

Do

Set esi as pArray

Set esi as esi + ebx

Set eax as [esi]

Call writeInt

Set edi as edi + 4

Set ebx as [edi]

Set ebx as ebx \* 4

Set ecx as ecx - 1

While (ecx>0)

Call crlf

*Return*

DisplaySearchResult(in key as integer,in location as integer)

*Begin*

Set eax as key

Call writeInt

Call writeString

Set eax as location

Call writeInt

Call crlf

Set edx as 0

*Return*

**Assembly Source Code**

;; Author: Marco Martinez

;; Filename: IndexedSortAndSearch.asm

;; Version: 1.0

;; Description: Add a variable to the BubbleSort procedure in Section 9.5.1 that is set to 1 whenever a pair of

;; values is exchanged within the inner loop. Use this variable to exit the sort before its normal

;; completion if you discover that no exchanges took place during a complete pass through the

;; array. (This variable is commonly known as an exchange flag.)

;; Date: 12/8

;;

;; Program Change Log

;; ==================

;; Name Date Description

;; Marco 12/8 Create baseline for IndexedSortAndSearch.asm

;;

INCLUDE Irvine32.inc

.data

BEGIN\_DEFINE BYTE " SEARCH AND SORT TEST ",0

BORDER\_DEFINE BYTE "===============================================================================================",0

array1 DWORD 40,-10,400,20,-300,12,10,0

index1 DWORD 0,1,2,3,4,5,6,7

length1 DWORD LENGTHOF array1

array2 DWORD 0,-10,-20,-30,50,100,200,300,-100,-80,1000,2000,-5000,60,70,550,-550,-300,-900,1010,2300

index2 DWORD 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20

length2 DWORD LENGTHOF array2

array3 DWORD 50,40,30,20,10

index3 DWORD 0,1,2,3,4

length3 DWORD LENGTHOF array3

key1 DWORD -10

key2 DWORD 1000

key3 DWORD 0

location1 DWORD ?

location2 DWORD ?

location3 DWORD ?

msg1 BYTE "Unsorted array: ",0

msg2 BYTE "Sorted array: ",0

msg3 BYTE "Search found the value ",0

msg4 BYTE " at position ",0

.code

BubbleSort PROTO,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD

LinearSearch PROTO,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD,

key:DWORD

DisplayArray PROTO,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD

DisplaySearchResult PROTO,

key:DWORD,

location:DWORD

main PROC

mov ecx,0

Begin:

pushad

call crlf

mov edx,OFFSET BEGIN\_DEFINE

add ecx,1

mov eax,ecx

call writeString

call writeInt

call crlf

popad

push edx

mov edx,OFFSET BORDER\_DEFINE

call writeString

call crlf

pop edx

pushad

mov edx,OFFSET msg1

call writeString

INVOKE DisplayArray,ADDR array1,ADDR index1,length1

INVOKE BubbleSort,ADDR array1,ADDR index1,length1

mov edx,OFFSET msg2

call writeString

INVOKE DisplayArray,ADDR array1,ADDR index1,length1

popad

pushad

INVOKE LinearSearch,ADDR array1,ADDR index1,length1,key1

mov edx,OFFSET msg3

call writeString

mov edx,OFFSET msg4

mov location1,eax

INVOKE DisplaySearchResult,key1,location1

call crlf

popad

pushad

mov edx,OFFSET msg1

call writeString

INVOKE DisplayArray,ADDR array2,ADDR index2,length2

INVOKE BubbleSort,ADDR array2,ADDR index2,length2

mov edx,OFFSET msg2

call writeString

INVOKE DisplayArray,ADDR array2,ADDR index2,length2

popad

pushad

INVOKE LinearSearch,ADDR array2,ADDR index2,length2,key2

mov edx,OFFSET msg3

call writeString

mov edx,OFFSET msg4

mov location2,eax

INVOKE DisplaySearchResult,key2,location2

call crlf

popad

pushad

mov edx,OFFSET msg2

call writeString

INVOKE DisplayArray,ADDR array3,ADDR index3,length3

INVOKE BubbleSort,ADDR array3,ADDR index3,length3

mov edx,OFFSET msg2

call writeString

INVOKE DisplayArray,ADDR array3,ADDR index3,length3

popad

pushad

INVOKE LinearSearch,ADDR array3,ADDR index3,length3,key3

mov edx,OFFSET msg3

call writeString

mov edx,OFFSET msg4

mov location3,eax

INVOKE DisplaySearchResult,key3,location3

call crlf

call crlf

popad

add key1,10

add key2,10

add key3,10

inc ecx

cmp ecx,3

jl Begin

exit

main ENDP

BubbleSort PROC USES eax ebx ecx edx esi edi,

pArray:PTR DWORD, ; pointer to array

pIndex:PTR DWORD,

Count:DWORD ; array size

LOCAL swap:BYTE

mov swap,0

mov ecx,Count

dec ecx ; decrement count by 1

L1:

push ecx ; save outer loop count

mov edi,pIndex

mov ebx,[edi]

shl ebx,2

L2:

mov esi,pArray

add esi,ebx

mov eax,[esi]

mov esi,pArray

mov ebx,[edi+4]

shl ebx,2

mov esi,pArray

add esi,ebx

cmp [esi],eax; compare a pair of values

jg L3

mov ebx,[edi+4]

xchg ebx,[edi]

mov [edi+4],ebx

inc swap

L3:

add edi,4

mov ebx,[edi]

shl ebx,2

loop L2 ; inner loop

cmp swap,0

je l4

mov swap,0

pop ecx ; retrieve outer loop count

loop L1 ; else repeat outer loop

L4:

ret

BubbleSort ENDP

LinearSearch PROC USES ebx ecx edx esi edi,

pArray:PTR DWORD, ; pointer to array

pIndex:PTR DWORD,

Count:DWORD, ; array size

key:DWORD

mov eax,0

mov edx,0

mov ecx,Count

mov edi,pIndex

mov ebx,[edi]

shl ebx,2

L1:

mov esi,pArray

add esi,ebx

mov eax,[esi]

cmp eax,key

je Found

inc edx

add edi,4

mov ebx,[edi]

shl ebx,2

dec ecx

cmp ecx,0

jg L1

mov eax,-1

jmp Return

Found:

mov eax,edx

Return:

ret

LinearSearch ENDP

DisplayArray PROC USES eax ebx ecx edx esi edi,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD

mov ecx,Count

mov edi,pIndex

mov ebx,[edi]

shl ebx,2

L1:

mov esi,pArray

add esi,ebx

mov eax,[esi]

call writeInt

add edi,4

mov ebx,[edi]

shl ebx,2

dec ecx

cmp ecx,0

jg L1

call crlf

ret

DisplayArray ENDP

DisplaySearchResult PROC USES eax ebx ecx esi edi,

key:DWORD,

location:DWORD,

mov eax,key

call writeInt

call writeString

mov eax,location

call writeInt

call crlf

mov edx,0

ret

DisplaySearchResult ENDP

end main

**Listing File**

Microsoft (R) Macro Assembler Version 14.15.26732.1 12/15/18 18:52:11

..\..\..\..\Documents\School Work\P310\thirdProject\3b\IndexedSortAndSearch.asm Page 1 - 1

;; Author: Marco Martinez

;; Filename: IndexedSortAndSearch.asm

;; Version: 1.0

;; Description: Add a variable to the BubbleSort procedure in Section 9.5.1 that is set to 1 whenever a pair of

;; values is exchanged within the inner loop. Use this variable to exit the sort before its normal

;; completion if you discover that no exchanges took place during a complete pass through the

;; array. (This variable is commonly known as an exchange flag.)

;; Date: 12/8

;;

;; Program Change Log

;; ==================

;; Name Date Description

;; Marco 12/8 Create baseline for IndexedSortAndSearch.asm

;;

INCLUDE Irvine32.inc

C ; Include file for Irvine32.lib (Irvine32.inc)

C

C ;OPTION CASEMAP:NONE ; optional: make identifiers case-sensitive

C

C INCLUDE SmallWin.inc ; MS-Windows prototypes, structures, and constants

C .NOLIST

C .LIST

C

C INCLUDE VirtualKeys.inc

C ; VirtualKeys.inc

C .NOLIST

C .LIST

C

C

C .NOLIST

C .LIST

C

00000000 .data

00000000 09 09 09 09 09 BEGIN\_DEFINE BYTE " SEARCH AND SORT TEST ",0

53 45 41 52 43

48 20 41 4E 44

20 53 4F 52 54

20 54 45 53 54

20 00

0000001B 3D 3D 3D 3D 3D BORDER\_DEFINE BYTE "===============================================================================================",0

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

3D 3D 3D 3D 3D

00

0000007B 00000028 array1 DWORD 40,-10,400,20,-300,12,10,0

FFFFFFF6

00000190

00000014

FFFFFED4

0000000C

0000000A

00000000

0000009B 00000000 index1 DWORD 0,1,2,3,4,5,6,7

00000001

00000002

00000003

00000004

00000005

00000006

00000007

000000BB 00000008 length1 DWORD LENGTHOF array1

000000BF 00000000 array2 DWORD 0,-10,-20,-30,50,100,200,300,-100,-80,1000,2000,-5000,60,70,550,-550,-300,-900,1010,2300

FFFFFFF6

FFFFFFEC

FFFFFFE2

00000032

00000064

000000C8

0000012C

FFFFFF9C

FFFFFFB0

000003E8

000007D0

FFFFEC78

0000003C

00000046

00000226

FFFFFDDA

FFFFFED4

FFFFFC7C

000003F2

000008FC

00000113 00000000 index2 DWORD 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20

00000001

00000002

00000003

00000004

00000005

00000006

00000007

00000008

00000009

0000000A

0000000B

0000000C

0000000D

0000000E

0000000F

00000010

00000011

00000012

00000013

00000014

00000167 00000015 length2 DWORD LENGTHOF array2

0000016B 00000032 array3 DWORD 50,40,30,20,10

00000028

0000001E

00000014

0000000A

0000017F 00000000 index3 DWORD 0,1,2,3,4

00000001

00000002

00000003

00000004

00000193 00000005 length3 DWORD LENGTHOF array3

00000197 FFFFFFF6 key1 DWORD -10

0000019B 000003E8 key2 DWORD 1000

0000019F 00000000 key3 DWORD 0

000001A3 00000000 location1 DWORD ?

000001A7 00000000 location2 DWORD ?

000001AB 00000000 location3 DWORD ?

000001AF 55 6E 73 6F 72 msg1 BYTE "Unsorted array: ",0

74 65 64 20 61

72 72 61 79 3A

20 00

000001C0 53 6F 72 74 65 msg2 BYTE "Sorted array: ",0

64 20 61 72 72

61 79 3A 20 00

000001CF 53 65 61 72 63 msg3 BYTE "Search found the value ",0

68 20 66 6F 75

6E 64 20 74 68

65 20 76 61 6C

75 65 20 00

000001E7 20 61 74 20 70 msg4 BYTE " at position ",0

6F 73 69 74 69

6F 6E 20 00

00000000 .code

BubbleSort PROTO,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD

LinearSearch PROTO,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD,

key:DWORD

DisplayArray PROTO,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD

DisplaySearchResult PROTO,

key:DWORD,

location:DWORD

00000000 main PROC

00000000 B9 00000000 mov ecx,0

00000005 Begin:

00000005 60 pushad

00000006 E8 00000000 E call crlf

0000000B BA 00000000 R mov edx,OFFSET BEGIN\_DEFINE

00000010 83 C1 01 add ecx,1

00000013 8B C1 mov eax,ecx

00000015 E8 00000000 E call writeString

0000001A E8 00000000 E call writeInt

0000001F E8 00000000 E call crlf

00000024 61 popad

00000025 52 push edx

00000026 BA 0000001B R mov edx,OFFSET BORDER\_DEFINE

0000002B E8 00000000 E call writeString

00000030 E8 00000000 E call crlf

00000035 5A pop edx

00000036 60 pushad

00000037 BA 000001AF R mov edx,OFFSET msg1

0000003C E8 00000000 E call writeString

INVOKE DisplayArray,ADDR array1,ADDR index1,length1

00000041 FF 35 000000BB R \* push length1

00000047 68 0000009B R \* push OFFSET index1

0000004C 68 0000007B R \* push OFFSET array1

00000051 E8 0000028B \* call DisplayArray

INVOKE BubbleSort,ADDR array1,ADDR index1,length1

00000056 FF 35 000000BB R \* push length1

0000005C 68 0000009B R \* push OFFSET index1

00000061 68 0000007B R \* push OFFSET array1

00000066 E8 000001CA \* call BubbleSort

0000006B BA 000001C0 R mov edx,OFFSET msg2

00000070 E8 00000000 E call writeString

INVOKE DisplayArray,ADDR array1,ADDR index1,length1

00000075 FF 35 000000BB R \* push length1

0000007B 68 0000009B R \* push OFFSET index1

00000080 68 0000007B R \* push OFFSET array1

00000085 E8 00000257 \* call DisplayArray

0000008A 61 popad

0000008B 60 pushad

INVOKE LinearSearch,ADDR array1,ADDR index1,length1,key1

0000008C FF 35 00000197 R \* push key1

00000092 FF 35 000000BB R \* push length1

00000098 68 0000009B R \* push OFFSET index1

0000009D 68 0000007B R \* push OFFSET array1

000000A2 E8 000001F0 \* call LinearSearch

000000A7 BA 000001CF R mov edx,OFFSET msg3

000000AC E8 00000000 E call writeString

000000B1 BA 000001E7 R mov edx,OFFSET msg4

000000B6 A3 000001A3 R mov location1,eax

INVOKE DisplaySearchResult,key1,location1

000000BB FF 35 000001A3 R \* push location1

000000C1 FF 35 00000197 R \* push key1

000000C7 E8 00000252 \* call DisplaySearchResult

000000CC E8 00000000 E call crlf

000000D1 61 popad

000000D2 60 pushad

000000D3 BA 000001AF R mov edx,OFFSET msg1

000000D8 E8 00000000 E call writeString

INVOKE DisplayArray,ADDR array2,ADDR index2,length2

000000DD FF 35 00000167 R \* push length2

000000E3 68 00000113 R \* push OFFSET index2

000000E8 68 000000BF R \* push OFFSET array2

000000ED E8 000001EF \* call DisplayArray

INVOKE BubbleSort,ADDR array2,ADDR index2,length2

000000F2 FF 35 00000167 R \* push length2

000000F8 68 00000113 R \* push OFFSET index2

000000FD 68 000000BF R \* push OFFSET array2

00000102 E8 0000012E \* call BubbleSort

00000107 BA 000001C0 R mov edx,OFFSET msg2

0000010C E8 00000000 E call writeString

INVOKE DisplayArray,ADDR array2,ADDR index2,length2

00000111 FF 35 00000167 R \* push length2

00000117 68 00000113 R \* push OFFSET index2

0000011C 68 000000BF R \* push OFFSET array2

00000121 E8 000001BB \* call DisplayArray

00000126 61 popad

00000127 60 pushad

INVOKE LinearSearch,ADDR array2,ADDR index2,length2,key2

00000128 FF 35 0000019B R \* push key2

0000012E FF 35 00000167 R \* push length2

00000134 68 00000113 R \* push OFFSET index2

00000139 68 000000BF R \* push OFFSET array2

0000013E E8 00000154 \* call LinearSearch

00000143 BA 000001CF R mov edx,OFFSET msg3

00000148 E8 00000000 E call writeString

0000014D BA 000001E7 R mov edx,OFFSET msg4

00000152 A3 000001A7 R mov location2,eax

INVOKE DisplaySearchResult,key2,location2

00000157 FF 35 000001A7 R \* push location2

0000015D FF 35 0000019B R \* push key2

00000163 E8 000001B6 \* call DisplaySearchResult

00000168 E8 00000000 E call crlf

0000016D 61 popad

0000016E 60 pushad

0000016F BA 000001C0 R mov edx,OFFSET msg2

00000174 E8 00000000 E call writeString

INVOKE DisplayArray,ADDR array3,ADDR index3,length3

00000179 FF 35 00000193 R \* push length3

0000017F 68 0000017F R \* push OFFSET index3

00000184 68 0000016B R \* push OFFSET array3

00000189 E8 00000153 \* call DisplayArray

INVOKE BubbleSort,ADDR array3,ADDR index3,length3

0000018E FF 35 00000193 R \* push length3

00000194 68 0000017F R \* push OFFSET index3

00000199 68 0000016B R \* push OFFSET array3

0000019E E8 00000092 \* call BubbleSort

000001A3 BA 000001C0 R mov edx,OFFSET msg2

000001A8 E8 00000000 E call writeString

INVOKE DisplayArray,ADDR array3,ADDR index3,length3

000001AD FF 35 00000193 R \* push length3

000001B3 68 0000017F R \* push OFFSET index3

000001B8 68 0000016B R \* push OFFSET array3

000001BD E8 0000011F \* call DisplayArray

000001C2 61 popad

000001C3 60 pushad

INVOKE LinearSearch,ADDR array3,ADDR index3,length3,key3

000001C4 FF 35 0000019F R \* push key3

000001CA FF 35 00000193 R \* push length3

000001D0 68 0000017F R \* push OFFSET index3

000001D5 68 0000016B R \* push OFFSET array3

000001DA E8 000000B8 \* call LinearSearch

000001DF BA 000001CF R mov edx,OFFSET msg3

000001E4 E8 00000000 E call writeString

000001E9 BA 000001E7 R mov edx,OFFSET msg4

000001EE A3 000001AB R mov location3,eax

INVOKE DisplaySearchResult,key3,location3

000001F3 FF 35 000001AB R \* push location3

000001F9 FF 35 0000019F R \* push key3

000001FF E8 0000011A \* call DisplaySearchResult

00000204 E8 00000000 E call crlf

00000209 E8 00000000 E call crlf

0000020E 61 popad

0000020F 83 05 00000197 R add key1,10

0A

00000216 83 05 0000019B R add key2,10

0A

0000021D 83 05 0000019F R add key3,10

0A

00000224 41 inc ecx

00000225 83 F9 03 cmp ecx,3

00000228 0F 8C FFFFFDD7 jl Begin

exit

0000022E 6A 00 \* push +000000000h

00000230 E8 00000000 E \* call ExitProcess

00000235 main ENDP

00000235 BubbleSort PROC USES eax ebx ecx edx esi edi,

pArray:PTR DWORD, ; pointer to array

pIndex:PTR DWORD,

Count:DWORD ; array size

LOCAL swap:BYTE

00000235 55 \* push ebp

00000236 8B EC \* mov ebp, esp

00000238 83 C4 FC \* add esp, 0FFFFFFFCh

0000023B 50 \* push eax

0000023C 53 \* push ebx

0000023D 51 \* push ecx

0000023E 52 \* push edx

0000023F 56 \* push esi

00000240 57 \* push edi

00000241 C6 45 FF 00 mov swap,0

00000245 8B 4D 10 mov ecx,Count

00000248 49 dec ecx ; decrement count by 1

00000249 L1:

00000249 51 push ecx ; save outer loop count

0000024A 8B 7D 0C mov edi,pIndex

0000024D 8B 1F mov ebx,[edi]

0000024F C1 E3 02 shl ebx,2

00000252 L2:

00000252 8B 75 08 mov esi,pArray

00000255 03 F3 add esi,ebx

00000257 8B 06 mov eax,[esi]

00000259 8B 75 08 mov esi,pArray

0000025C 8B 5F 04 mov ebx,[edi+4]

0000025F C1 E3 02 shl ebx,2

00000262 8B 75 08 mov esi,pArray

00000265 03 F3 add esi,ebx

00000267 39 06 cmp [esi],eax; compare a pair of values

00000269 7F 0B jg L3

0000026B 8B 5F 04 mov ebx,[edi+4]

0000026E 87 1F xchg ebx,[edi]

00000270 89 5F 04 mov [edi+4],ebx

00000273 FE 45 FF inc swap

00000276 L3:

00000276 83 C7 04 add edi,4

00000279 8B 1F mov ebx,[edi]

0000027B C1 E3 02 shl ebx,2

0000027E E2 D2 loop L2 ; inner loop

00000280 80 7D FF 00 cmp swap,0

00000284 74 07 je l4

00000286 C6 45 FF 00 mov swap,0

0000028A 59 pop ecx ; retrieve outer loop count

0000028B E2 BC loop L1 ; else repeat outer loop

0000028D L4:

ret

0000028D 5F \* pop edi

0000028E 5E \* pop esi

0000028F 5A \* pop edx

00000290 59 \* pop ecx

00000291 5B \* pop ebx

00000292 58 \* pop eax

00000293 C9 \* leave

00000294 C2 000C \* ret 0000Ch

00000297 BubbleSort ENDP

00000297 LinearSearch PROC USES ebx ecx edx esi edi,

pArray:PTR DWORD, ; pointer to array

pIndex:PTR DWORD,

Count:DWORD, ; array size

key:DWORD

00000297 55 \* push ebp

00000298 8B EC \* mov ebp, esp

0000029A 53 \* push ebx

0000029B 51 \* push ecx

0000029C 52 \* push edx

0000029D 56 \* push esi

0000029E 57 \* push edi

0000029F B8 00000000 mov eax,0

000002A4 BA 00000000 mov edx,0

000002A9 8B 4D 10 mov ecx,Count

000002AC 8B 7D 0C mov edi,pIndex

000002AF 8B 1F mov ebx,[edi]

000002B1 C1 E3 02 shl ebx,2

000002B4 L1:

000002B4 8B 75 08 mov esi,pArray

000002B7 03 F3 add esi,ebx

000002B9 8B 06 mov eax,[esi]

000002BB 3B 45 14 cmp eax,key

000002BE 74 16 je Found

000002C0 42 inc edx

000002C1 83 C7 04 add edi,4

000002C4 8B 1F mov ebx,[edi]

000002C6 C1 E3 02 shl ebx,2

000002C9 49 dec ecx

000002CA 83 F9 00 cmp ecx,0

000002CD 7F E5 jg L1

000002CF B8 FFFFFFFF mov eax,-1

000002D4 EB 02 jmp Return

000002D6 Found:

000002D6 8B C2 mov eax,edx

000002D8 Return:

ret

000002D8 5F \* pop edi

000002D9 5E \* pop esi

000002DA 5A \* pop edx

000002DB 59 \* pop ecx

000002DC 5B \* pop ebx

000002DD C9 \* leave

000002DE C2 0010 \* ret 00010h

000002E1 LinearSearch ENDP

000002E1 DisplayArray PROC USES eax ebx ecx edx esi edi,

pArray:PTR DWORD,

pIndex:PTR DWORD,

Count:DWORD

000002E1 55 \* push ebp

000002E2 8B EC \* mov ebp, esp

000002E4 50 \* push eax

000002E5 53 \* push ebx

000002E6 51 \* push ecx

000002E7 52 \* push edx

000002E8 56 \* push esi

000002E9 57 \* push edi

000002EA 8B 4D 10 mov ecx,Count

000002ED 8B 7D 0C mov edi,pIndex

000002F0 8B 1F mov ebx,[edi]

000002F2 C1 E3 02 shl ebx,2

000002F5 L1:

000002F5 8B 75 08 mov esi,pArray

000002F8 03 F3 add esi,ebx

000002FA 8B 06 mov eax,[esi]

000002FC E8 00000000 E call writeInt

00000301 83 C7 04 add edi,4

00000304 8B 1F mov ebx,[edi]

00000306 C1 E3 02 shl ebx,2

00000309 49 dec ecx

0000030A 83 F9 00 cmp ecx,0

0000030D 7F E6 jg L1

0000030F E8 00000000 E call crlf

ret

00000314 5F \* pop edi

00000315 5E \* pop esi

00000316 5A \* pop edx

00000317 59 \* pop ecx

00000318 5B \* pop ebx

00000319 58 \* pop eax

0000031A C9 \* leave

0000031B C2 000C \* ret 0000Ch

0000031E DisplayArray ENDP

0000031E DisplaySearchResult PROC USES eax ebx ecx esi edi,

key:DWORD,

location:DWORD,

0000031E 55 \* push ebp

0000031F 8B EC \* mov ebp, esp

00000321 50 \* push eax

00000322 53 \* push ebx

00000323 51 \* push ecx

00000324 56 \* push esi

00000325 57 \* push edi

00000326 8B 45 08 mov eax,key

00000329 E8 00000000 E call writeInt

0000032E E8 00000000 E call writeString

00000333 8B 45 0C mov eax,location

00000336 E8 00000000 E call writeInt

0000033B E8 00000000 E call crlf

00000340 BA 00000000 mov edx,0

ret

00000345 5F \* pop edi

00000346 5E \* pop esi

00000347 59 \* pop ecx

00000348 5B \* pop ebx

00000349 58 \* pop eax

0000034A C9 \* leave

0000034B C2 0008 \* ret 00008h

0000034E DisplaySearchResult ENDP

end main

Microsoft (R) Macro Assembler Version 14.15.26732.1 12/15/18 18:52:11

..\..\..\..\Documents\School Work\P310\thirdProject\3b\IndexedSortAndSearch.asm Symbols 2 - 1

Structures and Unions:

N a m e Size

Offset Type

CONSOLE\_CURSOR\_INFO . . . . . . 00000008

dwSize . . . . . . . . . . . . 00000000 DWord

bVisible . . . . . . . . . . . 00000004 DWord

CONSOLE\_SCREEN\_BUFFER\_INFO . . . 00000016

dwSize . . . . . . . . . . . . 00000000 DWord

dwCursorPosition . . . . . . . 00000004 DWord

wAttributes . . . . . . . . . 00000008 Word

srWindow . . . . . . . . . . . 0000000A QWord

dwMaximumWindowSize . . . . . 00000012 DWord

COORD . . . . . . . . . . . . . 00000004

X . . . . . . . . . . . . . . 00000000 Word

Y . . . . . . . . . . . . . . 00000002 Word

FILETIME . . . . . . . . . . . . 00000008

loDateTime . . . . . . . . . . 00000000 DWord

hiDateTime . . . . . . . . . . 00000004 DWord

FOCUS\_EVENT\_RECORD . . . . . . . 00000004

bSetFocus . . . . . . . . . . 00000000 DWord

FPU\_ENVIRON . . . . . . . . . . 0000001C

controlWord . . . . . . . . . 00000000 Word

statusWord . . . . . . . . . . 00000004 Word

tagWord . . . . . . . . . . . 00000008 Word

instrPointerOffset . . . . . . 0000000C DWord

instrPointerSelector . . . . . 00000010 DWord

operandPointerOffset . . . . . 00000014 DWord

operandPointerSelector . . . . 00000018 Word

INPUT\_RECORD . . . . . . . . . . 00000014

EventType . . . . . . . . . . 00000000 Word

Event . . . . . . . . . . . . 00000004 XmmWord

bKeyDown . . . . . . . . . . . 00000000 DWord

wRepeatCount . . . . . . . . . 00000004 Word

wVirtualKeyCode . . . . . . . 00000006 Word

wVirtualScanCode . . . . . . . 00000008 Word

uChar . . . . . . . . . . . . 0000000A Word

UnicodeChar . . . . . . . . . 00000000 Word

AsciiChar . . . . . . . . . . 00000000 Byte

dwControlKeyState . . . . . . 0000000C DWord

dwMousePosition . . . . . . . 00000000 DWord

dwButtonState . . . . . . . . 00000004 DWord

dwMouseControlKeyState . . . . 00000008 DWord

dwEventFlags . . . . . . . . . 0000000C DWord

dwSize . . . . . . . . . . . . 00000000 DWord

dwCommandId . . . . . . . . . 00000000 DWord

bSetFocus . . . . . . . . . . 00000000 DWord

KEY\_EVENT\_RECORD . . . . . . . . 00000010

bKeyDown . . . . . . . . . . . 00000000 DWord

wRepeatCount . . . . . . . . . 00000004 Word

wVirtualKeyCode . . . . . . . 00000006 Word

wVirtualScanCode . . . . . . . 00000008 Word

uChar . . . . . . . . . . . . 0000000A Word

UnicodeChar . . . . . . . . . 00000000 Word

AsciiChar . . . . . . . . . . 00000000 Byte

dwControlKeyState . . . . . . 0000000C DWord

MENU\_EVENT\_RECORD . . . . . . . 00000004

dwCommandId . . . . . . . . . 00000000 DWord

MOUSE\_EVENT\_RECORD . . . . . . . 00000010

dwMousePosition . . . . . . . 00000000 DWord

dwButtonState . . . . . . . . 00000004 DWord

dwMouseControlKeyState . . . . 00000008 DWord

dwEventFlags . . . . . . . . . 0000000C DWord

SMALL\_RECT . . . . . . . . . . . 00000008

Left . . . . . . . . . . . . . 00000000 Word

Top . . . . . . . . . . . . . 00000002 Word

Right . . . . . . . . . . . . 00000004 Word

Bottom . . . . . . . . . . . . 00000006 Word

SYSTEMTIME . . . . . . . . . . . 00000010

wYear . . . . . . . . . . . . 00000000 Word

wMonth . . . . . . . . . . . . 00000002 Word

wDayOfWeek . . . . . . . . . . 00000004 Word

wDay . . . . . . . . . . . . . 00000006 Word

wHour . . . . . . . . . . . . 00000008 Word

wMinute . . . . . . . . . . . 0000000A Word

wSecond . . . . . . . . . . . 0000000C Word

wMilliseconds . . . . . . . . 0000000E Word

WINDOW\_BUFFER\_SIZE\_RECORD . . . 00000004

dwSize . . . . . . . . . . . . 00000000 DWord

Segments and Groups:

N a m e Size Length Align Combine Class

FLAT . . . . . . . . . . . . . . GROUP

STACK . . . . . . . . . . . . . 32 Bit 00001000 Para Stack 'STACK'

\_DATA . . . . . . . . . . . . . 32 Bit 000001F5 Para Public 'DATA'

\_TEXT . . . . . . . . . . . . . 32 Bit 0000034E Para Public 'CODE'

Procedures, parameters, and locals:

N a m e Type Value Attr

BubbleSort . . . . . . . . . . . P Near 00000235 \_TEXT Length= 00000062 Public STDCALL

pArray . . . . . . . . . . . . DWord bp + 00000008

pIndex . . . . . . . . . . . . DWord bp + 0000000C

Count . . . . . . . . . . . . DWord bp + 00000010

swap . . . . . . . . . . . . . Byte bp - 00000001

L1 . . . . . . . . . . . . . . L Near 00000249 \_TEXT

L2 . . . . . . . . . . . . . . L Near 00000252 \_TEXT

L3 . . . . . . . . . . . . . . L Near 00000276 \_TEXT

L4 . . . . . . . . . . . . . . L Near 0000028D \_TEXT

CloseFile . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

CloseHandle . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Clrscr . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

CreateFileA . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

CreateOutputFile . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Crlf . . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Delay . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

DisplayArray . . . . . . . . . . P Near 000002E1 \_TEXT Length= 0000003D Public STDCALL

pArray . . . . . . . . . . . . DWord bp + 00000008

pIndex . . . . . . . . . . . . DWord bp + 0000000C

Count . . . . . . . . . . . . DWord bp + 00000010

L1 . . . . . . . . . . . . . . L Near 000002F5 \_TEXT

DisplaySearchResult . . . . . . P Near 0000031E \_TEXT Length= 00000030 Public STDCALL

key . . . . . . . . . . . . . DWord bp + 00000008

location . . . . . . . . . . . DWord bp + 0000000C

DumpMem . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

DumpRegs . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ExitProcess . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

FileTimeToDosDateTime . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

FileTimeToSystemTime . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

FlushConsoleInputBuffer . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

FormatMessageA . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetCommandLineA . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetCommandTail . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetConsoleCP . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetConsoleCursorInfo . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetConsoleMode . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetConsoleScreenBufferInfo . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetDateTime . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetFileTime . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetKeyState . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetLastError . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetLocalTime . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetMaxXY . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetMseconds . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetNumberOfConsoleInputEvents . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetProcessHeap . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetStdHandle . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetSystemTime . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetTextColor . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

GetTickCount . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Gotoxy . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

HeapAlloc . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

HeapCreate . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

HeapDestroy . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

HeapFree . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

HeapSize . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

IsDigit . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

LinearSearch . . . . . . . . . . P Near 00000297 \_TEXT Length= 0000004A Public STDCALL

pArray . . . . . . . . . . . . DWord bp + 00000008

pIndex . . . . . . . . . . . . DWord bp + 0000000C

Count . . . . . . . . . . . . DWord bp + 00000010

key . . . . . . . . . . . . . DWord bp + 00000014

L1 . . . . . . . . . . . . . . L Near 000002B4 \_TEXT

Found . . . . . . . . . . . . L Near 000002D6 \_TEXT

Return . . . . . . . . . . . . L Near 000002D8 \_TEXT

LocalFree . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

MessageBoxA . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

MsgBoxAsk . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

MsgBox . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

OpenInputFile . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ParseDecimal32 . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ParseInteger32 . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

PeekConsoleInputA . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Random32 . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

RandomRange . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Randomize . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadChar . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadConsoleA . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadConsoleInputA . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadDec . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadFile . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadFloat . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadFromFile . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadHex . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadInt . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadKeyFlush . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadKey . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ReadString . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleCursorInfo . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleCursorPosition . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleMode . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleScreenBufferSize . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleTextAttribute . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleTitleA . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetConsoleWindowInfo . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetFilePointer . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetLocalTime . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SetTextColor . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

ShowFPUStack . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Sleep . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

StrLength . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Str\_compare . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Str\_copy . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Str\_length . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Str\_trim . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

Str\_ucase . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

SystemTimeToFileTime . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WaitMsg . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteBinB . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteBin . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteChar . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteConsoleA . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteConsoleOutputAttribute . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteConsoleOutputCharacterA . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteDec . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteFile . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteFloat . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteHexB . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteHex . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteInt . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteStackFrameName . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteStackFrame . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteString . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteToFile . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

WriteWindowsMsg . . . . . . . . P Near 00000000 FLAT Length= 00000000 External STDCALL

main . . . . . . . . . . . . . . P Near 00000000 \_TEXT Length= 00000235 Public STDCALL

Begin . . . . . . . . . . . . L Near 00000005 \_TEXT

printf . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External C

scanf . . . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External C

wsprintfA . . . . . . . . . . . P Near 00000000 FLAT Length= 00000000 External C

Symbols:

N a m e Type Value Attr

@CodeSize . . . . . . . . . . . Number 00000000h

@DataSize . . . . . . . . . . . Number 00000000h

@Interface . . . . . . . . . . . Number 00000003h

@Model . . . . . . . . . . . . . Number 00000007h

@code . . . . . . . . . . . . . Text \_TEXT

@data . . . . . . . . . . . . . Text FLAT

@fardata? . . . . . . . . . . . Text FLAT

@fardata . . . . . . . . . . . . Text FLAT

@stack . . . . . . . . . . . . . Text FLAT

ALT\_MASK . . . . . . . . . . . . Number 00000003h

BEGIN\_DEFINE . . . . . . . . . . Byte 00000000 \_DATA

BORDER\_DEFINE . . . . . . . . . Byte 0000001B \_DATA

CAPSLOCK\_ON . . . . . . . . . . Number 00000080h

CREATE\_ALWAYS . . . . . . . . . Number 00000002h

CREATE\_NEW . . . . . . . . . . . Number 00000001h

CTRL\_MASK . . . . . . . . . . . Number 0000000Ch

CreateFile . . . . . . . . . . . Text CreateFileA

DO\_NOT\_SHARE . . . . . . . . . . Number 00000000h

ENABLE\_ECHO\_INPUT . . . . . . . Number 00000004h

ENABLE\_LINE\_INPUT . . . . . . . Number 00000002h

ENABLE\_MOUSE\_INPUT . . . . . . . Number 00000010h

ENABLE\_PROCESSED\_INPUT . . . . . Number 00000001h

ENABLE\_PROCESSED\_OUTPUT . . . . Number 00000001h

ENABLE\_WINDOW\_INPUT . . . . . . Number 00000008h

ENABLE\_WRAP\_AT\_EOL\_OUTPUT . . . Number 00000002h

ENHANCED\_KEY . . . . . . . . . . Number 00000100h

FALSE . . . . . . . . . . . . . Number 00000000h

FILE\_APPEND\_DATA . . . . . . . . Number 00000004h

FILE\_ATTRIBUTE\_ARCHIVE . . . . . Number 00000020h

FILE\_ATTRIBUTE\_COMPRESSED . . . Number 00000800h

FILE\_ATTRIBUTE\_DEVICE . . . . . Number 00000040h

FILE\_ATTRIBUTE\_DIRECTORY . . . . Number 00000010h

FILE\_ATTRIBUTE\_ENCRYPTED . . . . Number 00004000h

FILE\_ATTRIBUTE\_HIDDEN . . . . . Number 00000002h

FILE\_ATTRIBUTE\_NORMAL . . . . . Number 00000080h

FILE\_ATTRIBUTE\_NOT\_CONTENT\_INDEXED . Number 00002000h

FILE\_ATTRIBUTE\_OFFLINE . . . . . Number 00001000h

FILE\_ATTRIBUTE\_READONLY . . . . Number 00000001h

FILE\_ATTRIBUTE\_REPARSE\_POINT . . Number 00000400h

FILE\_ATTRIBUTE\_SPARSE\_FILE . . . Number 00000200h

FILE\_ATTRIBUTE\_SYSTEM . . . . . Number 00000004h

FILE\_ATTRIBUTE\_TEMPORARY . . . . Number 00000100h

FILE\_BEGIN . . . . . . . . . . . Number 00000000h

FILE\_CURRENT . . . . . . . . . . Number 00000001h

FILE\_DELETE\_CHILD . . . . . . . Number 00000040h

FILE\_END . . . . . . . . . . . . Number 00000002h

FILE\_READ\_DATA . . . . . . . . . Number 00000001h

FILE\_SHARE\_DELETE . . . . . . . Number 00000004h

FILE\_SHARE\_READ . . . . . . . . Number 00000001h

FILE\_SHARE\_WRITE . . . . . . . . Number 00000002h

FILE\_WRITE\_DATA . . . . . . . . Number 00000002h

FOCUS\_EVENT . . . . . . . . . . Number 00000010h

FORMAT\_MESSAGE\_ALLOCATE\_BUFFER . Number 00000100h

FORMAT\_MESSAGE\_FROM\_SYSTEM . . . Number 00001000h

FormatMessage . . . . . . . . . Text FormatMessageA

GENERIC\_ALL . . . . . . . . . . Number 10000000h

GENERIC\_EXECUTE . . . . . . . . Number 20000000h

GENERIC\_READ . . . . . . . . . . Number -80000000h

GENERIC\_WRITE . . . . . . . . . Number 40000000h

GetCommandLine . . . . . . . . . Text GetCommandLineA

HANDLE . . . . . . . . . . . . . Text DWORD

HEAP\_GENERATE\_EXCEPTIONS . . . . Number 00000004h

HEAP\_GROWABLE . . . . . . . . . Number 00000002h

HEAP\_NO\_SERIALIZE . . . . . . . Number 00000001h

HEAP\_REALLOC\_IN\_PLACE\_ONLY . . . Number 00000010h

HEAP\_ZERO\_MEMORY . . . . . . . . Number 00000008h

IDABORT . . . . . . . . . . . . Number 00000003h

IDCANCEL . . . . . . . . . . . . Number 00000002h

IDCLOSE . . . . . . . . . . . . Number 00000008h

IDCONTINUE . . . . . . . . . . . Number 0000000Bh

IDHELP . . . . . . . . . . . . . Number 00000009h

IDIGNORE . . . . . . . . . . . . Number 00000005h

IDNO . . . . . . . . . . . . . . Number 00000007h

IDOK . . . . . . . . . . . . . . Number 00000001h

IDRETRY . . . . . . . . . . . . Number 00000004h

IDTIMEOUT . . . . . . . . . . . Number 00007D00h

IDTRYAGAIN . . . . . . . . . . . Number 0000000Ah

IDYES . . . . . . . . . . . . . Number 00000006h

INVALID\_HANDLE\_VALUE . . . . . . Number -00000001h

KBDOWN\_FLAG . . . . . . . . . . Number 00000001h

KEY\_EVENT . . . . . . . . . . . Number 00000001h

KEY\_MASKS . . . . . . . . . . . Number 0000001Fh

LEFT\_ALT\_PRESSED . . . . . . . . Number 00000002h

LEFT\_CTRL\_PRESSED . . . . . . . Number 00000008h

MB\_ABORTRETRYIGNORE . . . . . . Number 00000002h

MB\_APPLMODAL . . . . . . . . . . Number 00000000h

MB\_CANCELTRYCONTINUE . . . . . . Number 00000006h

MB\_DEFBUTTON1 . . . . . . . . . Number 00000000h

MB\_DEFBUTTON2 . . . . . . . . . Number 00000100h

MB\_DEFBUTTON3 . . . . . . . . . Number 00000200h

MB\_DEFBUTTON4 . . . . . . . . . Number 00000300h

MB\_HELP . . . . . . . . . . . . Number 00004000h

MB\_ICONASTERISK . . . . . . . . Number 00000040h

MB\_ICONERROR . . . . . . . . . . Number 00000010h

MB\_ICONEXCLAMATION . . . . . . . Number 00000030h

MB\_ICONHAND . . . . . . . . . . Number 00000010h

MB\_ICONINFORMATION . . . . . . . Number 00000040h

MB\_ICONQUESTION . . . . . . . . Number 00000020h

MB\_ICONSTOP . . . . . . . . . . Number 00000010h

MB\_ICONWARNING . . . . . . . . . Number 00000030h

MB\_OKCANCEL . . . . . . . . . . Number 00000001h

MB\_OK . . . . . . . . . . . . . Number 00000000h

MB\_RETRYCANCEL . . . . . . . . . Number 00000005h

MB\_SYSTEMMODAL . . . . . . . . . Number 00001000h

MB\_TASKMODAL . . . . . . . . . . Number 00002000h

MB\_USERICON . . . . . . . . . . Number 00000080h

MB\_YESNOCANCEL . . . . . . . . . Number 00000003h

MB\_YESNO . . . . . . . . . . . . Number 00000004h

MENU\_EVENT . . . . . . . . . . . Number 00000008h

MOUSE\_EVENT . . . . . . . . . . Number 00000002h

MessageBox . . . . . . . . . . . Text MessageBoxA

NULL . . . . . . . . . . . . . . Number 00000000h

NUMLOCK\_ON . . . . . . . . . . . Number 00000020h

OPEN\_ALWAYS . . . . . . . . . . Number 00000004h

OPEN\_EXISTING . . . . . . . . . Number 00000003h

PeekConsoleInput . . . . . . . . Text PeekConsoleInputA

RIGHT\_ALT\_PRESSED . . . . . . . Number 00000001h

RIGHT\_CTRL\_PRESSED . . . . . . . Number 00000004h

ReadConsoleInput . . . . . . . . Text ReadConsoleInputA

ReadConsole . . . . . . . . . . Text ReadConsoleA

SCROLLLOCK\_ON . . . . . . . . . Number 00000040h

SHIFT\_MASK . . . . . . . . . . . Number 00000010h

SHIFT\_PRESSED . . . . . . . . . Number 00000010h

STD\_ERROR\_HANDLE . . . . . . . . Number -0000000Ch

STD\_INPUT\_HANDLE . . . . . . . . Number -0000000Ah

STD\_OUTPUT\_HANDLE . . . . . . . Number -0000000Bh

SetConsoleTitle . . . . . . . . Text SetConsoleTitleA

TAB . . . . . . . . . . . . . . Number 00000009h

TRUE . . . . . . . . . . . . . . Number 00000001h

TRUNCATE\_EXISTING . . . . . . . Number 00000005h

VK\_11 . . . . . . . . . . . . . Number 000000BDh

VK\_12 . . . . . . . . . . . . . Number 000000BBh

VK\_ADD . . . . . . . . . . . . . Number 0000006Bh

VK\_BACK . . . . . . . . . . . . Number 00000008h

VK\_CANCEL . . . . . . . . . . . Number 00000003h

VK\_CAPITAL . . . . . . . . . . . Number 00000014h

VK\_CLEAR . . . . . . . . . . . . Number 0000000Ch

VK\_CONTROL . . . . . . . . . . . Number 00000011h

VK\_DECIMAL . . . . . . . . . . . Number 0000006Eh

VK\_DELETE . . . . . . . . . . . Number 0000002Eh

VK\_DIVIDE . . . . . . . . . . . Number 0000006Fh

VK\_DOWN . . . . . . . . . . . . Number 00000028h

VK\_END . . . . . . . . . . . . . Number 00000023h

VK\_ESCAPE . . . . . . . . . . . Number 0000001Bh

VK\_EXECUTE . . . . . . . . . . . Number 0000002Bh

VK\_F10 . . . . . . . . . . . . . Number 00000079h

VK\_F11 . . . . . . . . . . . . . Number 0000007Ah

VK\_F12 . . . . . . . . . . . . . Number 0000007Bh

VK\_F13 . . . . . . . . . . . . . Number 0000007Ch

VK\_F14 . . . . . . . . . . . . . Number 0000007Dh

VK\_F15 . . . . . . . . . . . . . Number 0000007Eh

VK\_F16 . . . . . . . . . . . . . Number 0000007Fh

VK\_F17 . . . . . . . . . . . . . Number 00000080h

VK\_F18 . . . . . . . . . . . . . Number 00000081h

VK\_F19 . . . . . . . . . . . . . Number 00000082h

VK\_F1 . . . . . . . . . . . . . Number 00000070h

VK\_F20 . . . . . . . . . . . . . Number 00000083h

VK\_F21 . . . . . . . . . . . . . Number 00000084h

VK\_F22 . . . . . . . . . . . . . Number 00000085h

VK\_F23 . . . . . . . . . . . . . Number 00000086h

VK\_F24 . . . . . . . . . . . . . Number 00000087h

VK\_F2 . . . . . . . . . . . . . Number 00000071h

VK\_F3 . . . . . . . . . . . . . Number 00000072h

VK\_F4 . . . . . . . . . . . . . Number 00000073h

VK\_F5 . . . . . . . . . . . . . Number 00000074h

VK\_F6 . . . . . . . . . . . . . Number 00000075h

VK\_F7 . . . . . . . . . . . . . Number 00000076h

VK\_F8 . . . . . . . . . . . . . Number 00000077h

VK\_F9 . . . . . . . . . . . . . Number 00000078h

VK\_HELP . . . . . . . . . . . . Number 0000002Fh

VK\_HOME . . . . . . . . . . . . Number 00000024h

VK\_INSERT . . . . . . . . . . . Number 0000002Dh

VK\_LBUTTON . . . . . . . . . . . Number 00000001h

VK\_LCONTROL . . . . . . . . . . Number 000000A2h

VK\_LEFT . . . . . . . . . . . . Number 00000025h

VK\_LMENU . . . . . . . . . . . . Number 000000A4h

VK\_LSHIFT . . . . . . . . . . . Number 000000A0h

VK\_MENU . . . . . . . . . . . . Number 00000012h

VK\_MULTIPLY . . . . . . . . . . Number 0000006Ah

VK\_NEXT . . . . . . . . . . . . Number 00000022h

VK\_NUMLOCK . . . . . . . . . . . Number 00000090h

VK\_NUMPAD0 . . . . . . . . . . . Number 00000060h

VK\_NUMPAD1 . . . . . . . . . . . Number 00000061h

VK\_NUMPAD2 . . . . . . . . . . . Number 00000062h

VK\_NUMPAD3 . . . . . . . . . . . Number 00000063h

VK\_NUMPAD4 . . . . . . . . . . . Number 00000064h

VK\_NUMPAD5 . . . . . . . . . . . Number 00000065h

VK\_NUMPAD6 . . . . . . . . . . . Number 00000066h

VK\_NUMPAD7 . . . . . . . . . . . Number 00000067h

VK\_NUMPAD8 . . . . . . . . . . . Number 00000068h

VK\_NUMPAD9 . . . . . . . . . . . Number 00000069h

VK\_PAUSE . . . . . . . . . . . . Number 00000013h

VK\_PRINT . . . . . . . . . . . . Number 0000002Ah

VK\_PRIOR . . . . . . . . . . . . Number 00000021h

VK\_RBUTTON . . . . . . . . . . . Number 00000002h

VK\_RCONTROL . . . . . . . . . . Number 000000A3h

VK\_RETURN . . . . . . . . . . . Number 0000000Dh

VK\_RIGHT . . . . . . . . . . . . Number 00000027h

VK\_RMENU . . . . . . . . . . . . Number 000000A5h

VK\_RSHIFT . . . . . . . . . . . Number 000000A1h

VK\_SCROLL . . . . . . . . . . . Number 00000091h

VK\_SEPARATER . . . . . . . . . . Number 0000006Ch

VK\_SHIFT . . . . . . . . . . . . Number 00000010h

VK\_SNAPSHOT . . . . . . . . . . Number 0000002Ch

VK\_SPACE . . . . . . . . . . . . Number 00000020h

VK\_SUBTRACT . . . . . . . . . . Number 0000006Dh

VK\_TAB . . . . . . . . . . . . . Number 00000009h

VK\_UP . . . . . . . . . . . . . Number 00000026h

WINDOW\_BUFFER\_SIZE\_EVENT . . . . Number 00000004h

WriteConsoleOutputCharacter . . Text WriteConsoleOutputCharacterA

WriteConsole . . . . . . . . . . Text WriteConsoleA

array1 . . . . . . . . . . . . . DWord 0000007B \_DATA

array2 . . . . . . . . . . . . . DWord 000000BF \_DATA

array3 . . . . . . . . . . . . . DWord 0000016B \_DATA

black . . . . . . . . . . . . . Number 00000000h

blue . . . . . . . . . . . . . . Number 00000001h

brown . . . . . . . . . . . . . Number 00000006h

cyan . . . . . . . . . . . . . . Number 00000003h

exit . . . . . . . . . . . . . . Text INVOKE ExitProcess,0

gray . . . . . . . . . . . . . . Number 00000008h

green . . . . . . . . . . . . . Number 00000002h

index1 . . . . . . . . . . . . . DWord 0000009B \_DATA

index2 . . . . . . . . . . . . . DWord 00000113 \_DATA

index3 . . . . . . . . . . . . . DWord 0000017F \_DATA

key1 . . . . . . . . . . . . . . DWord 00000197 \_DATA

key2 . . . . . . . . . . . . . . DWord 0000019B \_DATA

key3 . . . . . . . . . . . . . . DWord 0000019F \_DATA

length1 . . . . . . . . . . . . DWord 000000BB \_DATA

length2 . . . . . . . . . . . . DWord 00000167 \_DATA

length3 . . . . . . . . . . . . DWord 00000193 \_DATA

lightBlue . . . . . . . . . . . Number 00000009h

lightCyan . . . . . . . . . . . Number 0000000Bh

lightGray . . . . . . . . . . . Number 00000007h

lightGreen . . . . . . . . . . . Number 0000000Ah

lightMagenta . . . . . . . . . . Number 0000000Dh

lightRed . . . . . . . . . . . . Number 0000000Ch

location1 . . . . . . . . . . . DWord 000001A3 \_DATA

location2 . . . . . . . . . . . DWord 000001A7 \_DATA

location3 . . . . . . . . . . . DWord 000001AB \_DATA

magenta . . . . . . . . . . . . Number 00000005h

msg1 . . . . . . . . . . . . . . Byte 000001AF \_DATA

msg2 . . . . . . . . . . . . . . Byte 000001C0 \_DATA

msg3 . . . . . . . . . . . . . . Byte 000001CF \_DATA

msg4 . . . . . . . . . . . . . . Byte 000001E7 \_DATA

red . . . . . . . . . . . . . . Number 00000004h

white . . . . . . . . . . . . . Number 0000000Fh

wsprintf . . . . . . . . . . . . Text wsprintfA

yellow . . . . . . . . . . . . . Number 0000000Eh

0 Warnings

0 Errors

**Console Screenshot**

