**Lab 15**   
**Stack-Based Parameter Passing in Assembly**   
A list of numbers (integers) representing the distance travelled. Calculate the total distance that adding all the distance values using a stack-based procedure call. Push these values onto the stack and pass to a procedure called SumAll, which retrieves them from the stack using the base pointer (EBP), adds them, and returns the result in the EAX register. After the procedure completes, the main program should display the total distance.

**Solution 1 - without arrays:**   
INCLUDE Irvine32.inc   
.data   
; Simulated sensor readings at different time intervals (in units) distance1 DWORD 100   
distance2 DWORD 150   
distance3 DWORD 120   
distance4 DWORD 130   
msgTotal BYTE "Total distance covered by robot: ", 0   
.code   
main PROC   
 ; Display message   
 mov edx, OFFSET msgTotal   
 call WriteString   
 ; Push distance readings onto the stack (in reverse order) push distance4   
 push distance3   
 push distance2   
 push distance1   
 ; Call SumAll with 4 arguments   
 push 4 ; Number of values being summed   
 call SumAll ; Returns result in EAX   
 call WriteDec ; Display total distance (EAX)   
 call Crlf   
 exit   
main ENDP

; ----------------------------   
; SumAll Procedure   
; Receives:   
; [ebp+8] = count (number of values)   
; [ebp+12] ... = values to sum   
; Returns:   
; EAX = total sum   
; ----------------------------   
SumAll PROC   
 push ebp   
 mov ebp, esp   
 mov ecx, [ebp+8] ; number of values to add   
 mov eax, 0 ; initialize sum to 0   
 mov esi, 12 ; starting offset for first value   
SumLoop:   
 add eax, [ebp+esi] ; add value to EAX   
 add esi, 4 ; move to next value   
 loop SumLoop ; repeat for ECX values   
 pop ebp   
 ret 4 + 4\*4 ; clean up: 1 count + 4 values (5 args × 4 bytes) SumAll ENDP   
END main

**Solution 2 – using arrays:**   
INCLUDE Irvine32.inc   
.data   
; Sensor distance values (can be modified)   
distances DWORD 5, 10, 15, 20, 25   
count DWORD LENGTHOF distances ; Number of elements msg1 BYTE "Total distance covered by robot: ", 0   
.code   
main PROC

; Display message   
 mov edx, OFFSET msg1   
 call WriteString   
 ; Push array values onto the stack in reverse order   
 mov ecx, count   
 mov esi, OFFSET distances   
 add esi, (LENGTHOF distances - 1) \* 4 ; Point to last element   
push\_loop:   
 push DWORD PTR [esi]   
 sub esi, 4   
 loop push\_loop   
 ; Call SumAll and pass a number of elements   
 push LENGTHOF distances   
 call SumAll   
 ; Clean up parameters from stack (5 elements + 1 count = 6 DWORDs) add esp, (LENGTHOF distances + 1) \* 4   
 ; Result is in EAX, display it   
 call WriteInt   
 call Crlf   
 exit   
main ENDP   
; ---------------------------------------------   
; SumAll Procedure   
; Receives: number of items in stack (above return address)   
; Returns: sum in EAX   
; ---------------------------------------------   
SumAll PROC   
 push ebp   
 mov ebp, esp   
 ; Get number of items (first argument)   
 mov ecx, [ebp + 8]   
 ; Offset to first data item (starting from [ebp + 12])

mov esi, 12   
 ; Initialize sum   
 mov eax, 0   
sum\_loop:   
 add eax, [ebp + esi]   
 add esi, 4   
 loop sum\_loop   
 pop ebp   
 ret   
SumAll ENDP   
END main

A structure with three instances, each having an ID and a value. Push the values onto the stack. Call a procedure that determines and displays the values of all the structure’s instances as follows:   
Sample1 ID: 4, Value: 4   
Sample2 ID: 7, Value: 7   
Sample3 ID: 3, Value: 3

**SOLUTION:**   
INCLUDE Irvine32.inc   
; --------------------------   
; Define Sensor Structure   
; --------------------------   
Sample STRUCT   
 id DWORD ?

value DWORD ?

Sample ENDS   
.data   
; Define 3 Samples with IDs and Value   
Sample1 Sensor <1, 5>   
Sample2 Sensor <2, 3>   
Sample3 Sensor <3, 7>   
msg1 BYTE "ID: ", 0

msg2 BYTE ", Values: ", 0   
.code   
main PROC   
 ; Push each Sample’s Value and ID onto the stack   
 push Sample1.value   
 push Sample1.id   
 push Sample2.value   
 push Sample2.id   
 push Sample3.value   
 push Sample3.id   
 call DisplayTotal   
 exit   
main ENDP   
; ----------------------------------------------------   
; Procedure: DisplayTotal   
; Pops and displays Sample data (ID + value) for 3 Samples   
; ----------------------------------------------------   
DisplayTotal PROC   
 ; We'll process 3 samples (each with ID and value = 2 DWORDs) mov ecx, 3   
ProcessNext:   
 ; Pop sample ID into EBX   
 pop ebx   
 ; Pop sample value into EAX   
 pop eax   
 ; Display sample ID   
 mov edx, OFFSET msg1   
 call WriteString   
 mov eax, ebx   
 call WriteDec   
 ; Display total   
 mov edx, OFFSET msg2

call WriteString   
 call WriteDec   
 call Crlf   
 loop ProcessNext   
 ret   
DisplayTotal ENDP   
END main