

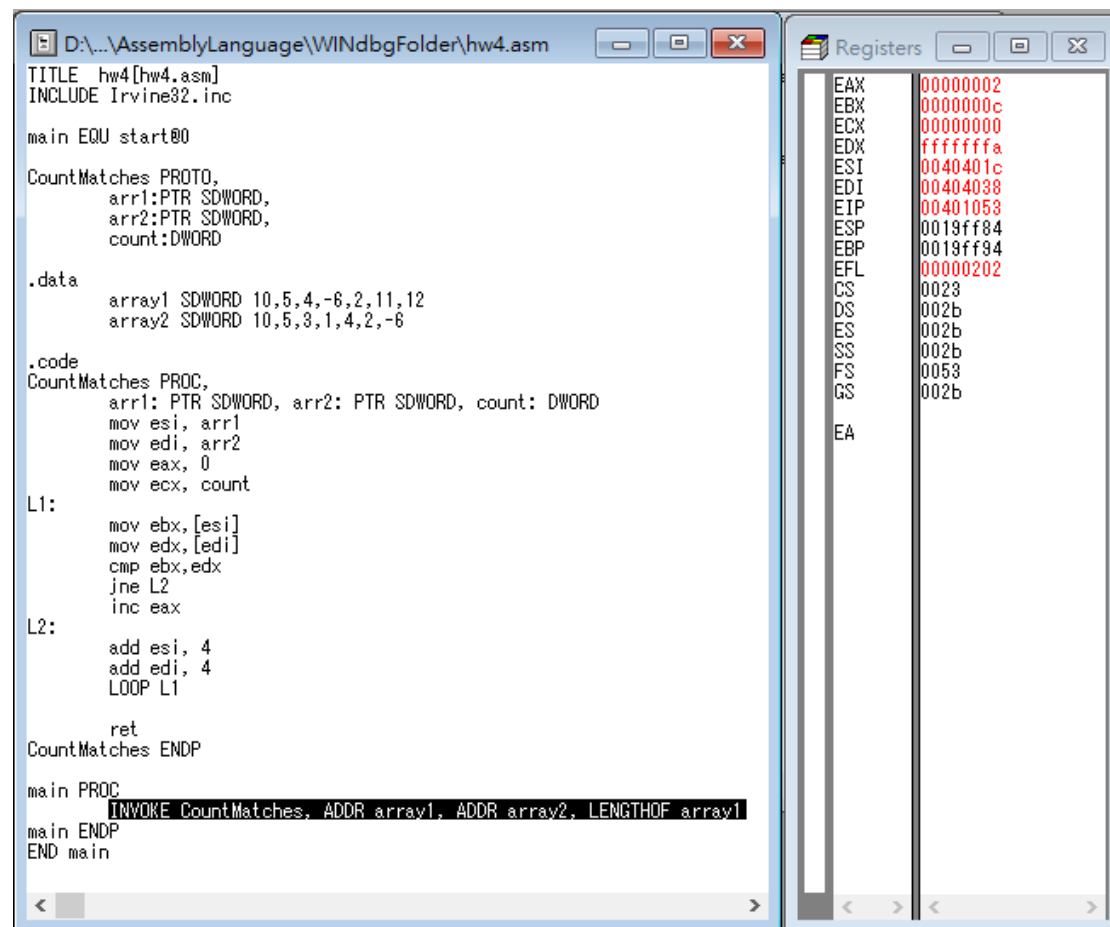
## HW#04Counting Matching Elements

資管 3A 102403015 程祥恩

### Code – Basic:

```
1  TITLE  hw4[hw4.asm]
2  INCLUDE Irvine32.inc
3
4  main EQU start@0
5
6  CountMatches PROTO,
7      arr1:PTR SDWORD,
8      arr2:PTR SDWORD,
9      count:DWORD
10
11  .data
12      array1 SDWORD 10,5,4,-6,2,11,12
13      array2 SDWORD 10,5,3,1,4,2,-6
14
15  .code
16  CountMatches PROC,
17      arr1: PTR SDWORD, arr2: PTR SDWORD, count: DWORD
18      mov esi, arr1
19      mov edi, arr2
20      mov eax, 0
21      mov ecx, count
22  L1:
23      mov ebx,[esi]
24      mov edx,[edi]
25      cmp ebx,edx
26      jne L2
27      inc eax
28  L2:
29      add esi, 4
30      add edi, 4
31      LOOP L1
32
33      ret
34  CountMatches ENDP
35
36  main PROC
37      INVOKE CountMatches, ADDR array1, ADDR array2, LENGTHOF array1
38  main ENDP
39  END main
```

## Result – Basic:



The screenshot displays an assembly editor window titled 'D:\...\AssemblyLanguage\WINdbgFolder\hw4.asm'. The source code defines a procedure 'CountMatches' that compares two arrays, 'array1' and 'array2', and returns the count of matching elements in 'eax'. The 'main' procedure calls 'CountMatches' with the addresses of the arrays and the length of 'array1'. The 'Registers' window on the right shows the current state of the CPU registers, with 'eax' set to 00000002.

```

D:\...\AssemblyLanguage\WINdbgFolder\hw4.asm
TITLE hw4[hw4.asm]
INCLUDE Irvine32.inc

main EQU start@0

CountMatches PROTO,
    arr1:PTR SDWORD,
    arr2:PTR SDWORD,
    count:DWORD

.data
    array1 SDWORD 10,5,4,-6,2,11,12
    array2 SDWORD 10,5,3,1,4,2,-6

.code
CountMatches PROC,
    arr1: PTR SDWORD, arr2: PTR SDWORD, count: DWORD
    mov esi, arr1
    mov edi, arr2
    mov eax, 0
    mov ecx, count

L1:
    mov ebx, [esi]
    mov edx, [edi]
    cmp ebx, edx
    jne L2
    inc eax

L2:
    add esi, 4
    add edi, 4
    LOOP L1

    ret
CountMatches ENDP

main PROC
    INVOKE CountMatches, ADDR array1, ADDR array2, LENGTHOF array1
main ENDP
END main

```

Register	Value
EAX	00000002
EBX	0000000c
ECX	00000000
EDX	fffffffa
ESI	0040401c
EDI	00404038
EIP	00401053
ESP	0019ff84
EBP	0019ff94
EFL	00000202
CS	0023
DS	002b
ES	002b
SS	002b
FS	0053
GS	002b
EA	

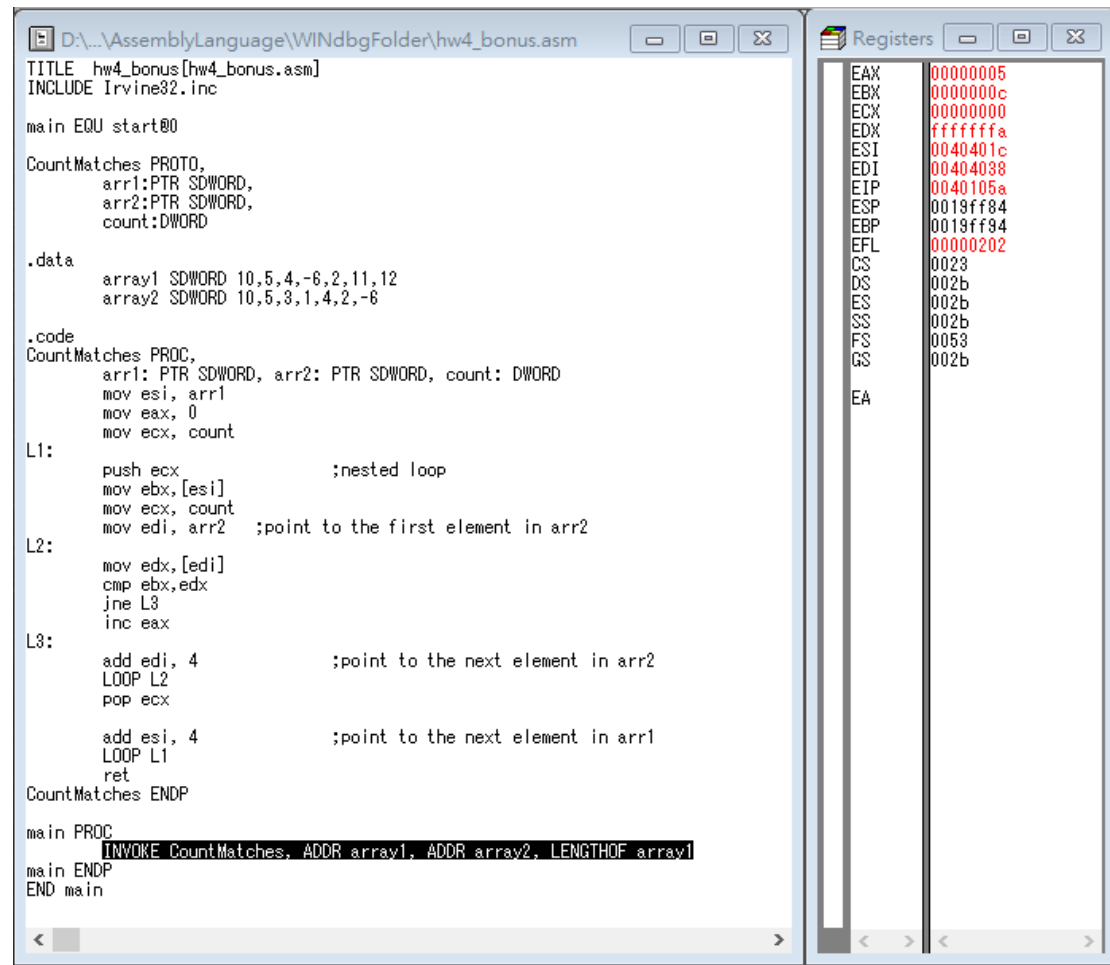
## Explanation – Basic:

首先要建立出 CountMatches PROC 的原型(PROTO)，原型包含定義程序名稱、傳入參數的型別與數目，如此一來才能用 INVOKE 呼叫 CountMatches PROC。在 CountMatches 中，先把 eax 設為零以便於存放結果，並建立一個迴圈，其重複次數等於陣列的元素數量，接著比對 arr1[i]和 arr2[i]的值，如果值相等，就會執行 inc eax 使 eax 加一，若否則直接跳到 L2，CountMatches PROC 結束後，eax 中即為「The number of matching array elements」。

## Code – Bonus:

```
1  TITLE  hw4_bonus[hw4_bonus.asm]
2  INCLUDE Irvine32.inc
3
4  main EQU start@0
5
6  CountMatches PROTO,
7      arr1:PTR SDWORD,
8      arr2:PTR SDWORD,
9      count:DWORD
10
11  .data
12      array1 SDWORD 10,5,4,-6,2,11,12
13      array2 SDWORD 10,5,3,1,4,2,-6
14
15  .code
16  CountMatches PROC,
17      arr1: PTR SDWORD, arr2: PTR SDWORD, count: DWORD
18      mov esi, arr1
19      mov eax, 0
20      mov ecx, count
21  L1:
22      push ecx          ;nested loop
23      mov ebx,[esi]
24      mov ecx, count
25      mov edi, arr2     ;point to the first element in arr2
26  L2:
27      mov edx,[edi]
28      cmp ebx,edx
29      jne L3
30      inc eax
31  L3:
32      add edi, 4        ;point to the next element in arr2
33      LOOP L2
34      pop ecx
35
36      add esi, 4        ;point to the next element in arr1
37      LOOP L1
38      ret
39  CountMatches ENDP
40
41  main PROC
42      INVOKE CountMatches, ADDR array1, ADDR array2, LENGTHOF array1
43  main ENDP
44  END main
```

## Result – Bonus:



The screenshot shows a debugger window with two panes. The left pane displays assembly code for a program named 'hw4\_bonus.asm'. The code defines a procedure 'CountMatches' that takes two pointers to single-word arrays and a count, and returns the number of matching elements. It uses nested loops (L1 and L2) to compare elements from 'arr1' and 'arr2'. The right pane shows the state of the CPU registers. EAX is 00000005, indicating the final count of matching elements.

```

D:\...\AssemblyLanguage\WINdbgFolder\hw4_bonus.asm
TITLE hw4_bonus[hw4_bonus.asm]
INCLUDE Irvine32.inc

main EQU start@0

CountMatches PROTO,
    arr1:PTR SDWORD,
    arr2:PTR SDWORD,
    count:DWORD

.data
    array1 SDWORD 10,5,4,-6,2,11,12
    array2 SDWORD 10,5,3,1,4,2,-6

.code
CountMatches PROC,
    arr1: PTR SDWORD, arr2: PTR SDWORD, count: DWORD
    mov esi, arr1
    mov eax, 0
    mov ecx, count
L1:
    push ecx                ;nested loop
    mov ebx,[esi]
    mov ecx, count
    mov edi, arr2           ;point to the first element in arr2
L2:
    mov edx,[edi]
    cmp ebx,edx
    jne L3
    inc eax
L3:
    add edi, 4              ;point to the next element in arr2
    LOOP L2
    pop ecx
    add esi, 4              ;point to the next element in arr1
    LOOP L1
    ret
CountMatches ENDP

main PROC
    INVOKE CountMatches, ADDR array1, ADDR array2, LENGTHOF array1
main ENDP
END main

```

Register	Value
EAX	00000005
EBX	0000000c
ECX	00000000
EDX	fffffffa
ESI	0040401c
EDI	00404038
EIP	0040105a
ESP	0019ff84
EBP	0019ff94
EFL	00000202
CS	0023
DS	002b
ES	002b
SS	002b
FS	0053
GS	002b
EA	

## Explanation – Bonus:

在 CountMatches 中，先把 eax 設為零以便於存放結果，並透過 stack 概念建立巢狀迴圈，此巢狀迴圈會將 arr1 和 arr2 所有元素兩兩比對，如果值相等，就會執行 inc eax 使 eax 加一，否則直接跳到 L3，CountMatches PROC 結束後，eax 中即為「The count of all the matching elements」。