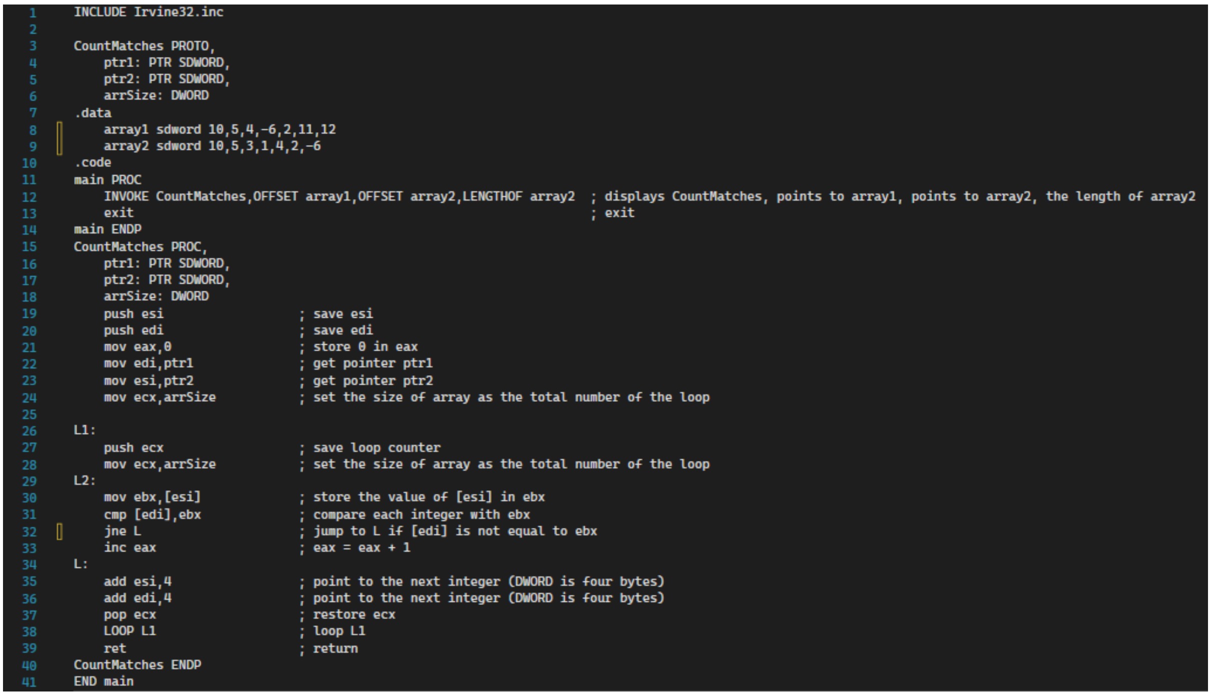
Title: Assembly Language HW3 Basic Function
Name / student ID: 曾千芸109504501

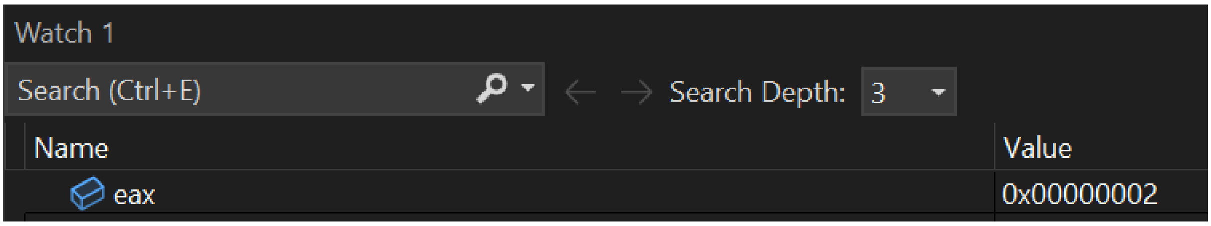
- **Screenshots of result and code + explanations**



```
INCLUDE Irvine32.inc

CountMatches PROTO,
    ptr1: PTR SDWORD,
    ptr2: PTR SDWORD,
    arrSize: DWORD
.data
    array1 sdword 10,5,4,-6,2,11,12
    array2 sdword 10,5,3,1,4,2,-6
.code
main PROC
    INVOKE CountMatches,OFFSET array1,OFFSET array2,LENGTHOF array2  ; displays CountMatches, points to array1, points to array2, the length of array2
    exit                                                             ; exit
main ENDP
CountMatches PROC,
    ptr1: PTR SDWORD,
    ptr2: PTR SDWORD,
    arrSize: DWORD
    push esi                  ; save esi
    push edi                  ; save edi
    mov eax,0                 ; store 0 in eax
    mov edi,ptr1              ; get pointer ptr1
    mov esi,ptr2              ; get pointer ptr2
    mov ecx,arrSize           ; set the size of array as the total number of the loop

L1:
    push ecx                  ; save loop counter
    mov ecx,arrSize           ; set the size of array as the total number of the loop
L2:
    mov ebx,[esi]             ; store the value of [esi] in ebx
    cmp [edi],ebx             ; compare each integer with ebx
    jne L                     ; jump to L if [edi] is not equal to ebx
    inc eax                   ; eax = eax + 1
L:
    add esi,4                 ; point to the next integer (DWORD is four bytes)
    add edi,4                 ; point to the next integer (DWORD is four bytes)
    pop ecx                   ; restore ecx
    LOOP L1                   ; loop L1
    ret                       ; return
CountMatches ENDP
END main
```



Watch 1

Search (Ctrl+E)   Search Depth: 3

Name	Value
eax	0x00000002

- **main idea**

1. store array1, array2 and the size of the array to CountMatches function
2. apply one-dimensional loop to compare whether each position of the array are the same, then add one to eax

- **detailed explanation**

1. edi is where array1 point to
2. esi is where array2 point to
3. ecx is the total number of the loop which means how many times the program is going to compare
4. set eax to zero at the beginning of the program
5. after the comparison, if both value are the same, then add one to eax, else jump to L and go back to L1

- **Thoughts about the HW3 Basic Function:**

Initially, I got stuck on the problem and I couldn't make any progress on it. Therefore, I started to study the textbook to figure out what is the possible way to solve out the question. I then found out that loop L2 was the most challenging part of this homework because I needed to be clear of the logic of comparison. Eventually, I did finish homework3 on time.