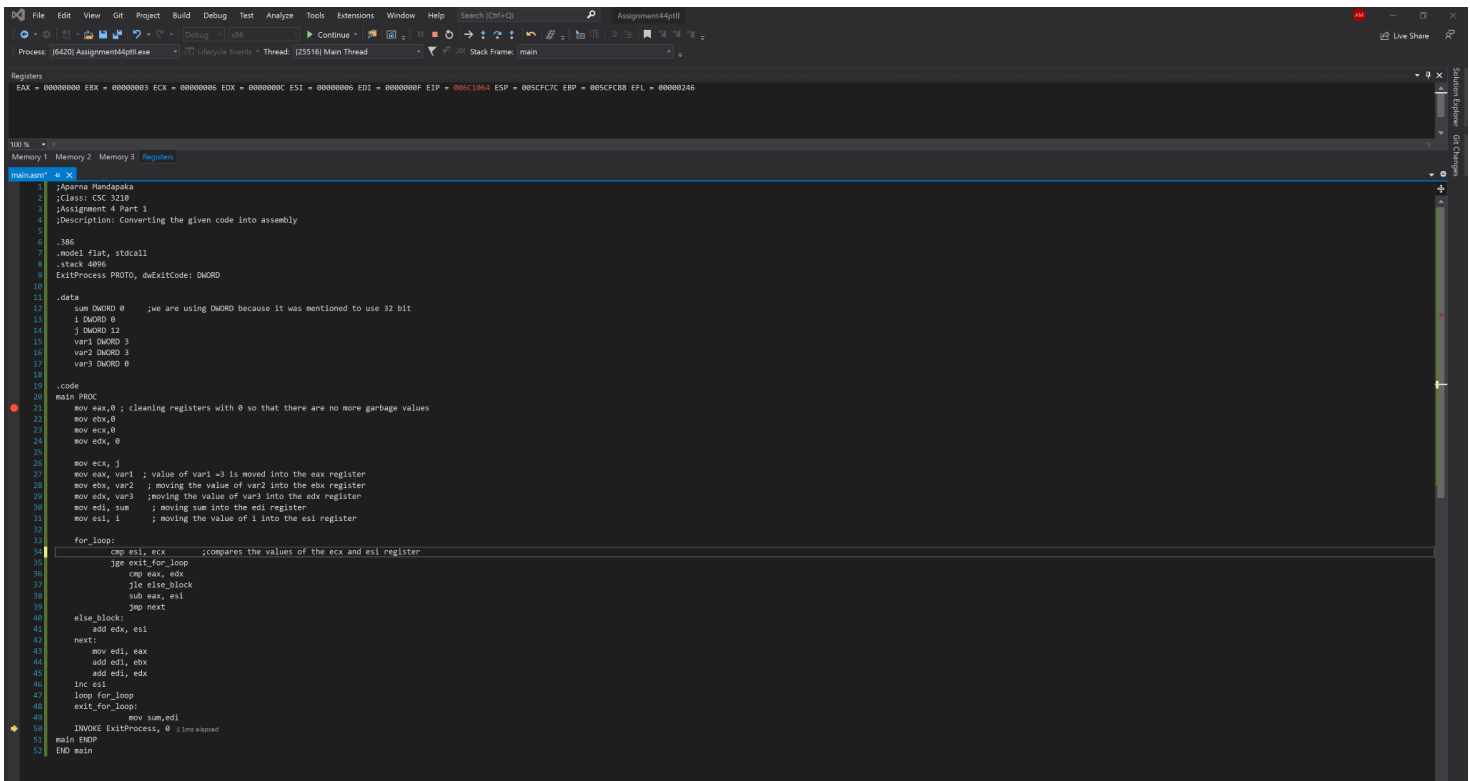
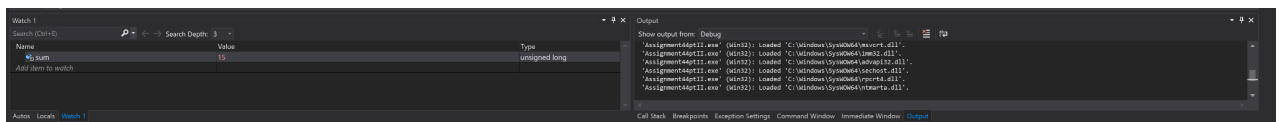


Question 1



```
1 ;Aparna Handapaka
2 ;Class: CSC 3210
3 ;Assignment 4 Part 1
4 ;Description: Converting the given code into assembly
5
6 .386
7 .model flat, stdcall
8 .stack 4096
9 ExitProcess PROTO, dwExitCode: DWORD
10
11 .data
12 sum DWORD 0 ;we are using DWORD because it was mentioned to use 32 bit
13 i DWORD 0
14 j DWORD 12
15 var1 DWORD 3
16 var2 DWORD 3
17 var3 DWORD 0
18
19 .code
20 main PROC
21 mov eax, 0 ; cleaning registers with 0 so that there are no more garbage values
22 mov ebx, 0
23 mov ecx, 0
24 mov edx, 0
25
26 mov ecx, j
27 mov eax, var1 ; value of var1 is moved into the eax register
28 mov ebx, var2 ; moving the value of var2 into the ebx register
29 mov edx, var3 ; moving the value of var3 into the edx register
30 mov edi, sum ; moving sum into the edi register
31 mov esi, i ; moving the value of i into the esi register
32
33 for_loop:
34 cmp esi, ecx ;compares the values of the ecx and esi register
35 jge exit_for_loop
36 cmp eax, edx
37 jle else_block
38 sub eax, esi
39 jmp next
40 else_block:
41 add edx, esi
42 next:
43 mov edi, eax
44 add edi, ebx
45 add edi, edx
46 inc esi
47 loop for_loop
48 exit_for_loop:
49 mov sum, edi
50 INVOKE ExitProcess, 0 ; line elapsed
51 main ENDP
52 END main
```

Watch window

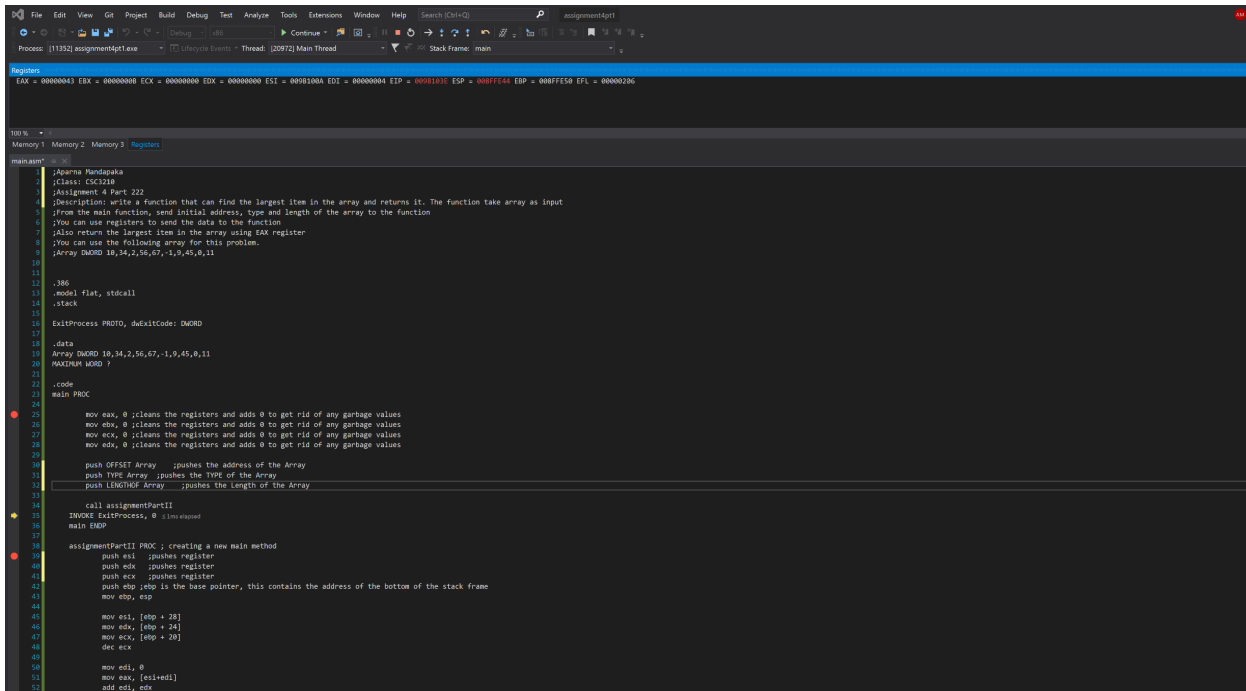


| Name | Value | Type |
|------|-------|---------------|
| sum | 15 | unsigned long |

Output

```
Assignment4Ap1.exe (WinSxS): Loaded 'C:\Windows\System32\user32.dll'.
Assignment4Ap1.exe (WinSxS): Loaded 'C:\Windows\System32\ole32.dll'.
Assignment4Ap1.exe (WinSxS): Loaded 'C:\Windows\System32\advapi32.dll'.
Assignment4Ap1.exe (WinSxS): Loaded 'C:\Windows\System32\kernel32.dll'.
Assignment4Ap1.exe (WinSxS): Loaded 'C:\Windows\System32\gdi32.dll'.
Assignment4Ap1.exe (WinSxS): Loaded 'C:\Windows\System32\user32.dll'.
```

Question 2



```
1 ;Aparna Mundapaka
2 ;Class: CSCI210
3 ;Assignment 4 Part 222
4 ;Description: write a function that can find the largest item in the array and returns it. The function take array as input
5 ;From the main function, send initial address, type and length of the array to the function
6 ;You can use registers to send the data to the function
7 ;Also return the largest item in the array using EAX register
8 ;You can use the following array for this problem.
9 ;Array DWORD 18,34,2,56,67,-1,9,45,8,11
10
11
12 .386
13 .model flat, stdcall
14 .stack
15
16 ExitProcess PROTO, dwExitCode: DWORD
17
18 .data
19 Array DWORD 18,34,2,56,67,-1,9,45,8,11
20 MAXIMUM WORD ?
21
22 .code
23 main PROC
24
25     mov eax, 0 ;cleans the registers and adds 0 to get rid of any garbage values
26     mov ebx, 0 ;cleans the registers and adds 0 to get rid of any garbage values
27     mov ecx, 0 ;cleans the registers and adds 0 to get rid of any garbage values
28     mov edx, 0 ;cleans the registers and adds 0 to get rid of any garbage values
29
30     push OFFSET Array ;pushes the address of the Array
31     push TYPE Array ;pushes the TYPE of the Array
32     push LENGTHOF Array ;pushes the Length of the Array
33
34     call assignmentPartII
35     INVOKE ExitProcess, 0 ;Exit returned
36     main ENDP
37
38 assignmentPartII PROC ; creating a new main method
39     push esi ;pushes register
40     push edx ;pushes register
41     push ecx ;pushes register
42     push ebp ;ebp is the base pointer, this contains the address of the bottom of the stack frame
43     mov ebp, esp
44
45     mov esi, [ebp + 28]
46     mov edx, [ebp + 24]
47     mov ecx, [ebp + 20]
48     dec ecx
49
50     mov edi, 0
51     mov eax, [esi+edi]
52     add edi, edx
```

```
53
54     L1:
55         mov ebx, [esi+edi]
56         cmp ebx, eax
57         jle next
58         mov eax, ebx
59
60     next:
61         add esi, edx
62
63     loop L1
64         pop ebp ;restores the outer loop count
65         pop ecx
66         pop edx
67         pop esi
68
69     ret 12 ;returns, updates and clears the stack
70 assignmentPartII ENDP
71 END main
```

Watch window

