HW # 9: Theme: Advanced Procedures, Stack Parameters, Locals and BCD

*(All main questions carry equal weight.  Credit awarded to only those answers for which work has been shown.)*

1. Write a procedure named *Arithmetic Expression which computes X=A-B + C.* The 32-bit variables A, B, and C must be passed as value to the stack.  The  value X should be returned on the top of the stack upon return from the procedure and its address must be returned as reference in the register EDI of the program.

**hw9-1.asm**

include irvine32.inc

.data

xVal DWORD ?

aVal DWORD ?

bVal DWORD ?

cVal DWORD ?

prompt BYTE "A program that computes X = A - B + C", 0

prompta BYTE "Enter a value for A: ", 0

promptb BYTE "Enter a value for B: ", 0

promptc BYTE "Enter a value for C: ", 0

promptx BYTE "X = ", 0

.code

main proc

mov edx, OFFSET prompt

call WriteString

call crlf

mov edx, OFFSET prompta

call WriteString

call ReadInt

mov aVal, eax

mov edx, OFFSET promptb

call WriteString

call ReadInt

mov bVal, eax

mov edx, OFFSET promptc

call WriteString

call ReadInt

mov cVal, eax

push aVal

push bVal

push cVal

push xVal

call ArithmeticExpression

invoke ExitProcess, 0

main endp

;-----------------------------------------------------

; ArithmeticExpression PROC

;

; Computes X=A-B + C.

; The 32-bit variables A, B, and C must be passed as value to the stack.

; The value X should be returned on the top of the stack upon return

; from the procedure and its address must be

; returned as reference in the register EDI of the program.

;

; Requires: A, B, and C must all be passed by value to the stack.

; Receives: aVal, bVal, cVal

; Returns: xVal on the top of the stack and returns its address in the EDI register

;-----------------------------------------------------

x\_local EQU DWORD PTR [esp + 8]

ArithmeticExpression PROC

push ebp

mov ebp, esp

mov eax, aVal

sub eax, bVal

add eax, cVal

mov x\_local, eax

lea edi, x\_local

mov edx, OFFSET promptx

call WriteString

call WriteInt

pop ebp

ret

ArithmeticExpression endp

END main

**hw9-1.lst**

Microsoft (R) Macro Assembler Version 14.28.29913.0 04/06/21 20:01:19

hw9-1.asm Page 1 - 1

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 00000000 xVal DWORD ?

00000004 00000000 aVal DWORD ?

00000008 00000000 bVal DWORD ?

0000000C 00000000 cVal DWORD ?

00000010 41 20 70 72 6F prompt BYTE "A program that computes X = A - B + C", 0

67 72 61 6D 20

74 68 61 74 20

63 6F 6D 70 75

74 65 73 20 58

20 3D 20 41 20

2D 20 42 20 2B

20 43 00

00000036 45 6E 74 65 72 prompta BYTE "Enter a value for A: ", 0

20 61 20 76 61

6C 75 65 20 66

6F 72 20 41 3A

20 00

0000004C 45 6E 74 65 72 promptb BYTE "Enter a value for B: ", 0

20 61 20 76 61

6C 75 65 20 66

6F 72 20 42 3A

20 00

00000062 45 6E 74 65 72 promptc BYTE "Enter a value for C: ", 0

20 61 20 76 61

6C 75 65 20 66

6F 72 20 43 3A

20 00

00000078 58 20 3D 20 00 promptx BYTE "X = ", 0

00000000 .code

00000000 main proc

00000000 BA 00000010 R mov edx, OFFSET prompt

00000005 E8 00000000 E call WriteString

0000000A E8 00000000 E call crlf

0000000F BA 00000036 R mov edx, OFFSET prompta

00000014 E8 00000000 E call WriteString

00000019 E8 00000000 E call ReadInt

0000001E A3 00000004 R mov aVal, eax

00000023 BA 0000004C R mov edx, OFFSET promptb

00000028 E8 00000000 E call WriteString

0000002D E8 00000000 E call ReadInt

00000032 A3 00000008 R mov bVal, eax

00000037 BA 00000062 R mov edx, OFFSET promptc

0000003C E8 00000000 E call WriteString

00000041 E8 00000000 E call ReadInt

00000046 A3 0000000C R mov cVal, eax

0000004B FF 35 00000004 R push aVal

00000051 FF 35 00000008 R push bVal

00000057 FF 35 0000000C R push cVal

0000005D FF 35 00000000 R push xVal

00000063 E8 00000007 call ArithmeticExpression

invoke ExitProcess, 0

00000068 6A 00 \* push +000000000h

0000006A E8 00000000 E \* call ExitProcess

0000006F main endp

;-----------------------------------------------------

; ArithmeticExpression PROC

;

; Computes X=A-B + C.

; The 32-bit variables A, B, and C must be passed as value to the stack.

; The value X should be returned on the top of the stack upon return

; from the procedure and its address must be

; returned as reference in the register EDI of the program.

;

; Requires: A, B, and C must all be passed by value to the stack.

; Receives: aVal, bVal, cVal

; Returns: xVal on the top of the stack and returns its address in the EDI register

;-----------------------------------------------------

= DWORD PTR [esp + 8] x\_local EQU DWORD PTR [esp + 8]

0000006F ArithmeticExpression PROC

0000006F 55 push ebp

00000070 8B EC mov ebp, esp

00000072 A1 00000004 R mov eax, aVal

00000077 2B 05 00000008 R sub eax, bVal

0000007D 03 05 0000000C R add eax, cVal

00000083 89 44 24 08 mov x\_local, eax

00000087 8D 7C 24 08 lea edi, x\_local

0000008B BA 00000078 R mov edx, OFFSET promptx

00000090 E8 00000000 E call WriteString

00000095 E8 00000000 E call WriteInt

0000009A 5D pop ebp

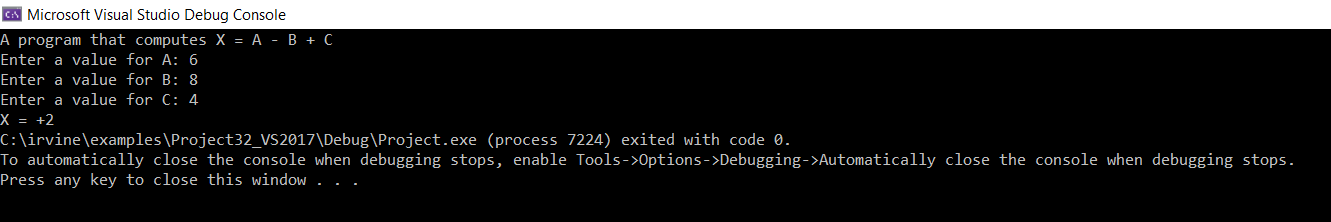
0000009B C3 ret

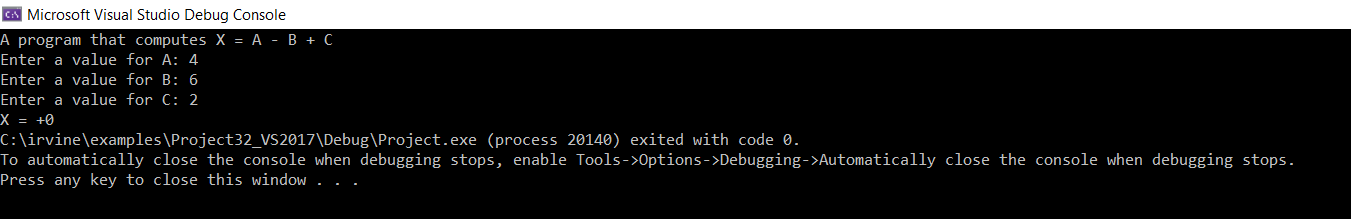
0000009C ArithmeticExpression endp

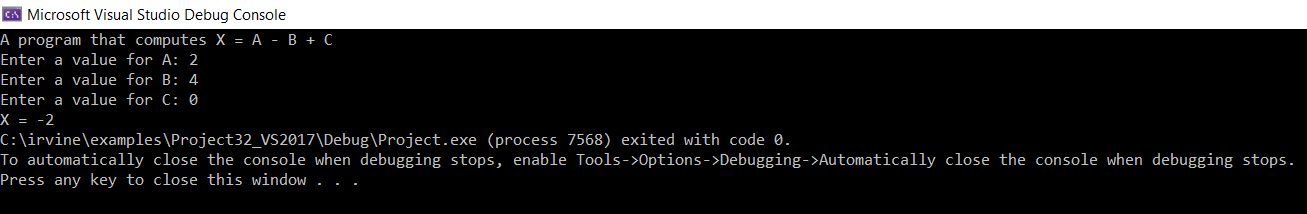
END main

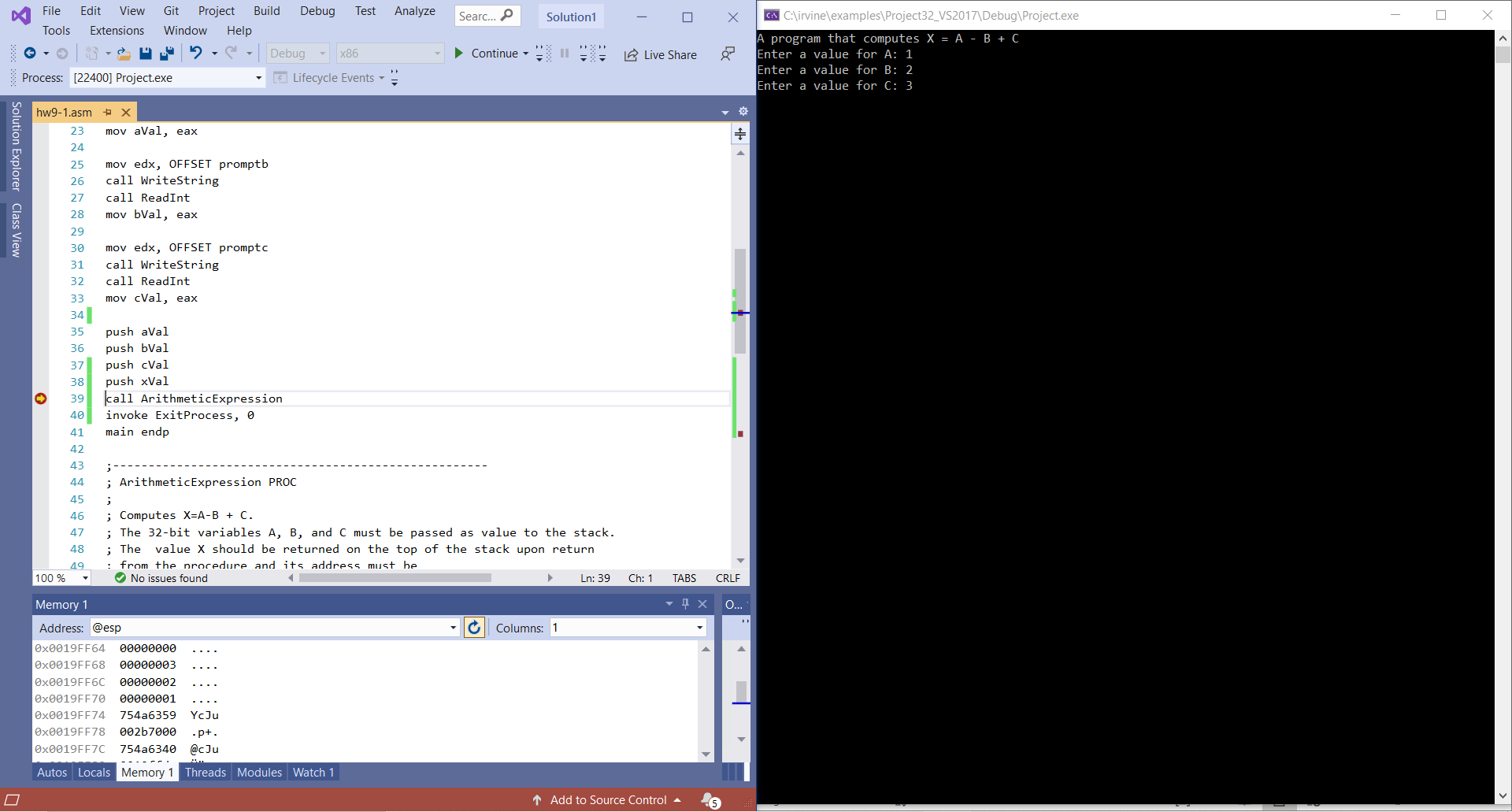
\_Microsoft (R) Macro Assembler Version 14.28.29913.0 04/06/21 20:01:19

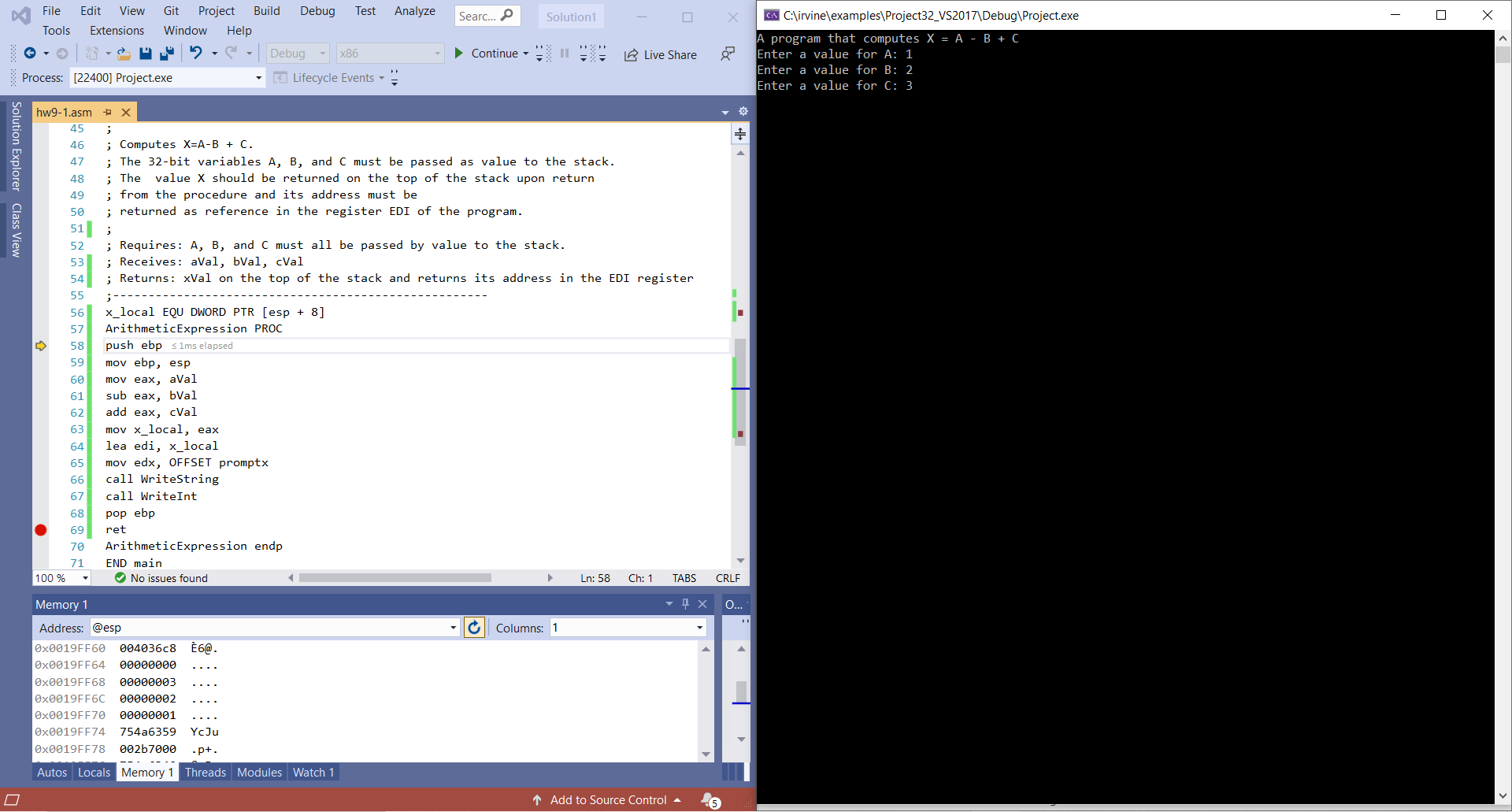
Screenshots for HW9-1 on next page

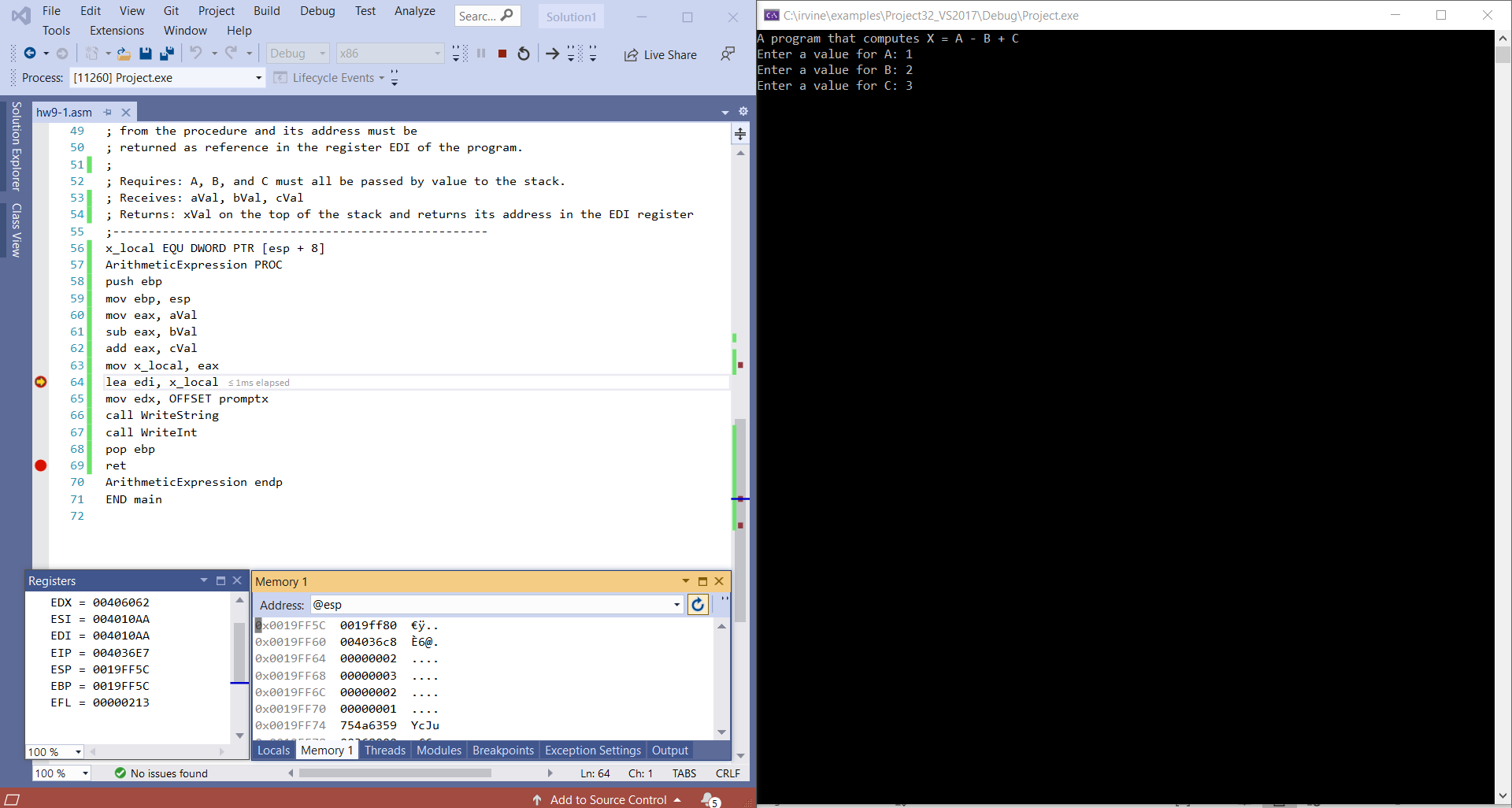


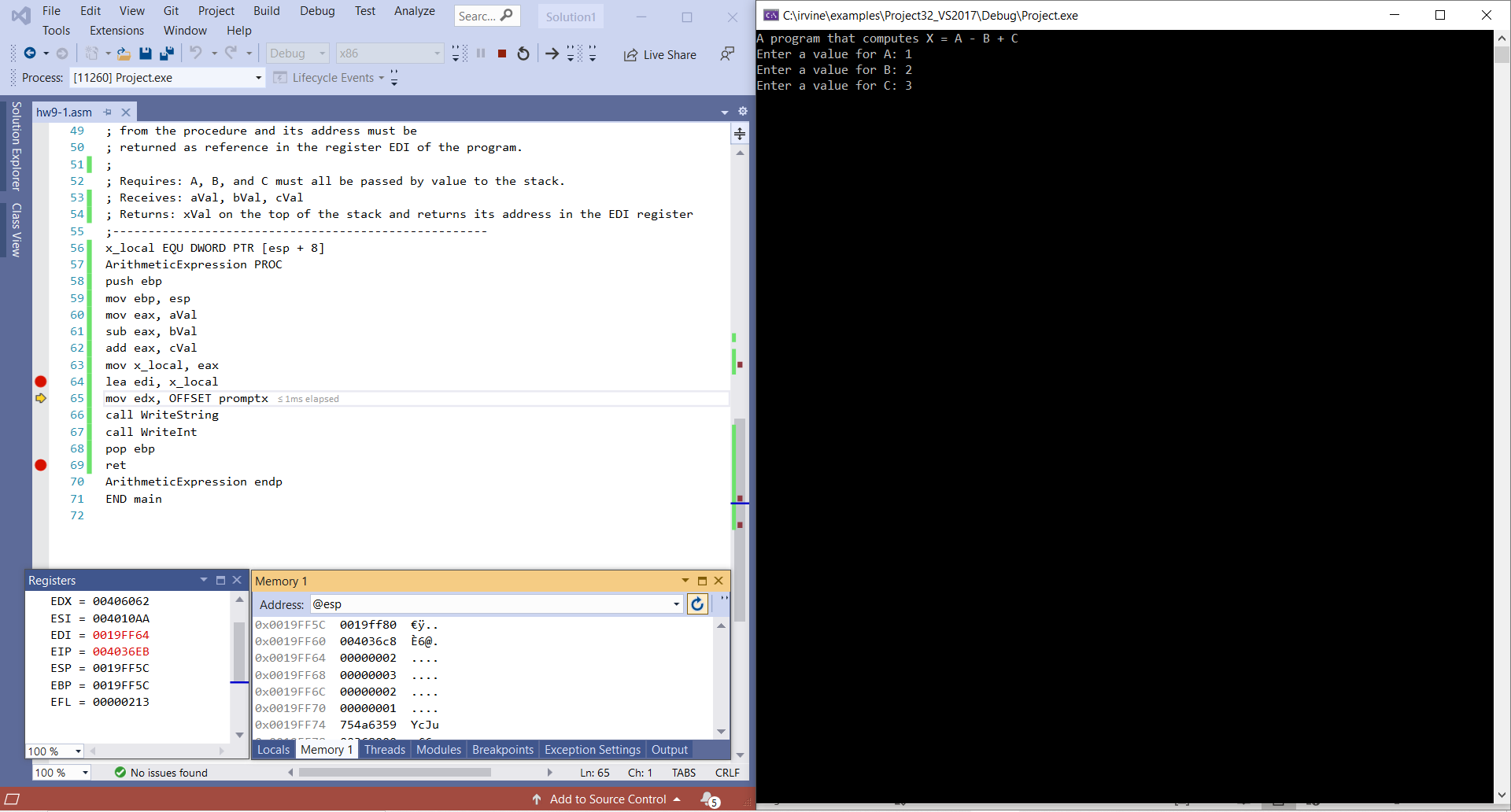


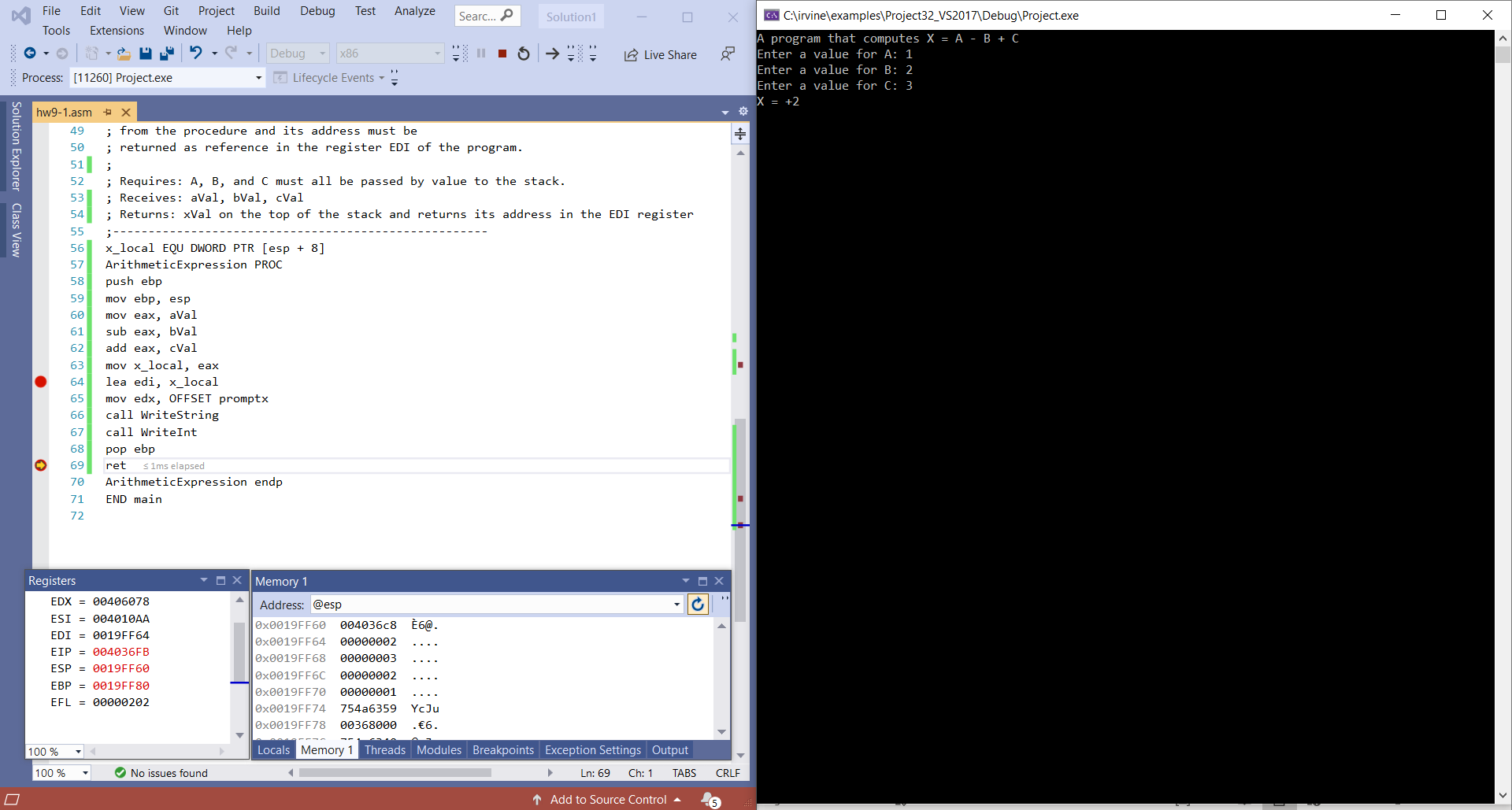


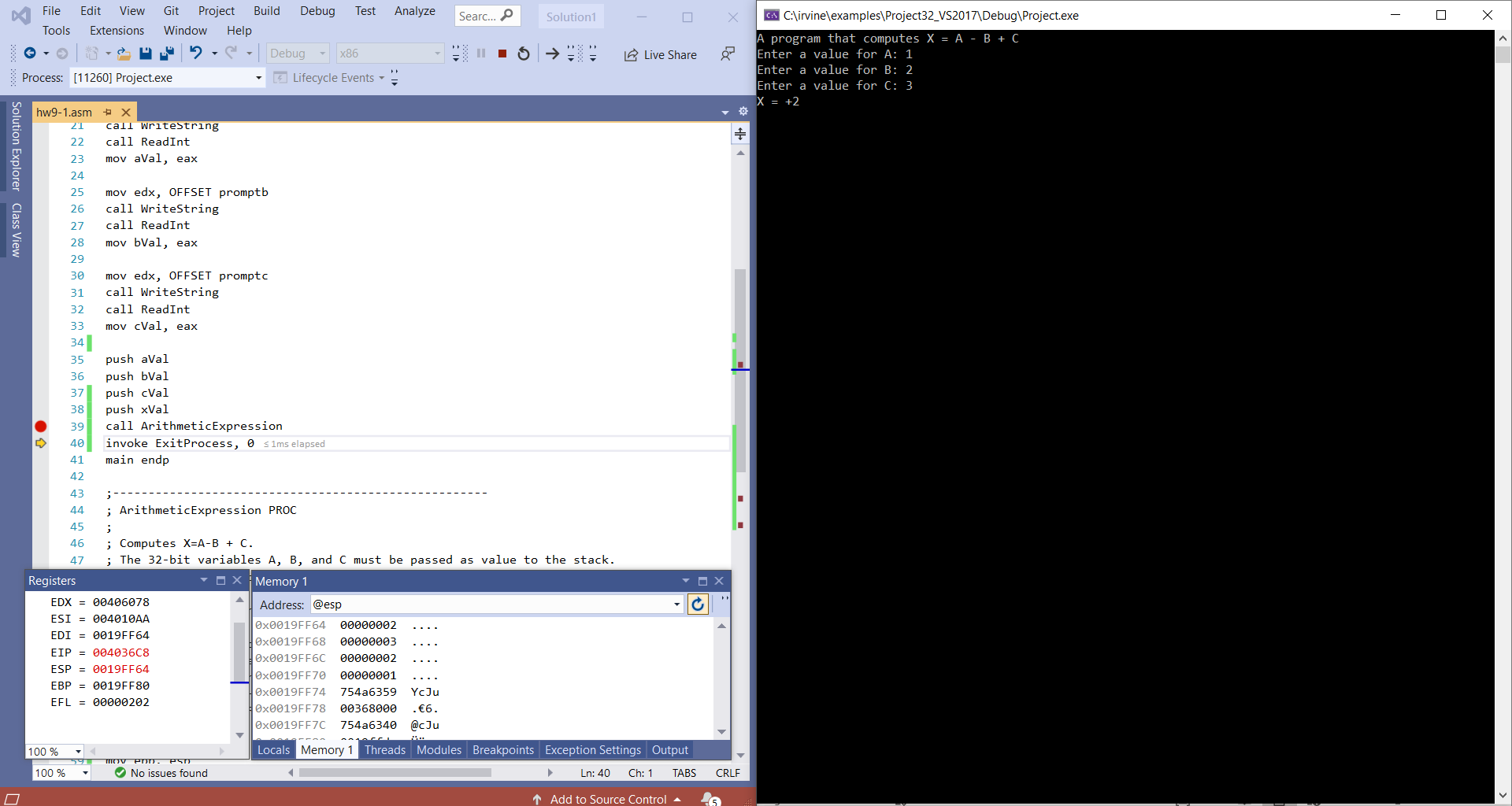






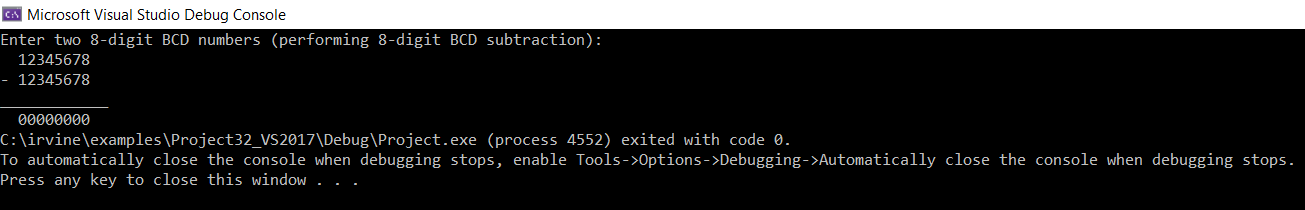


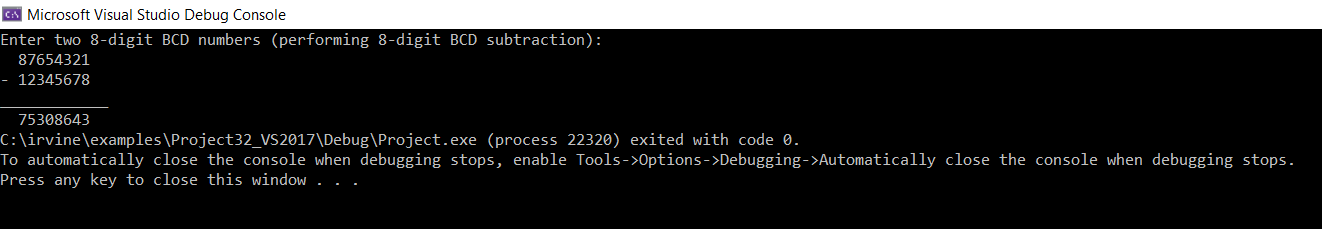


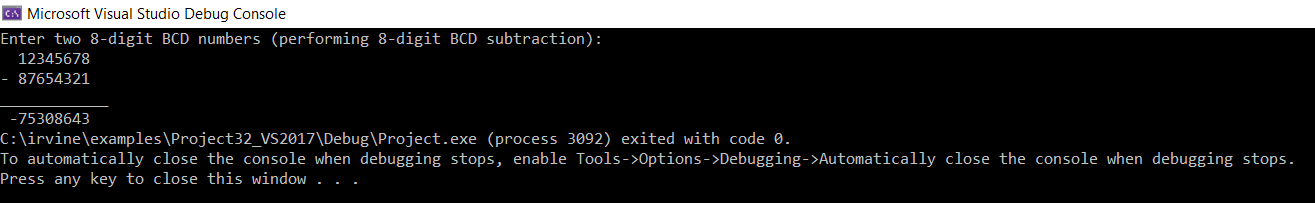


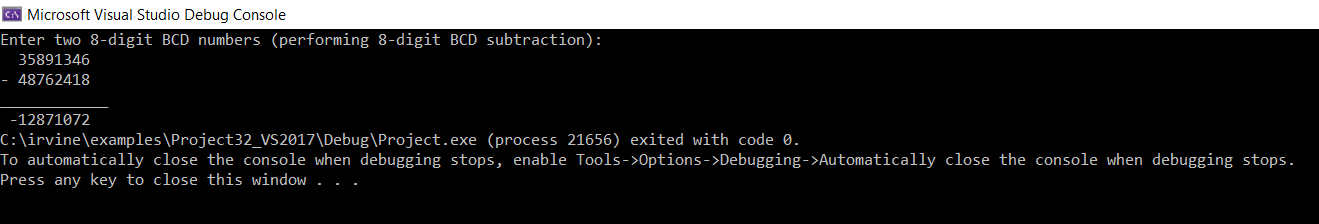
2. Draft a program that subtracts two 8-digit BCD numbers.  Please display the two input 8-digit numbers and the result on the screen.  Please try 3 different 8-digit numbers.

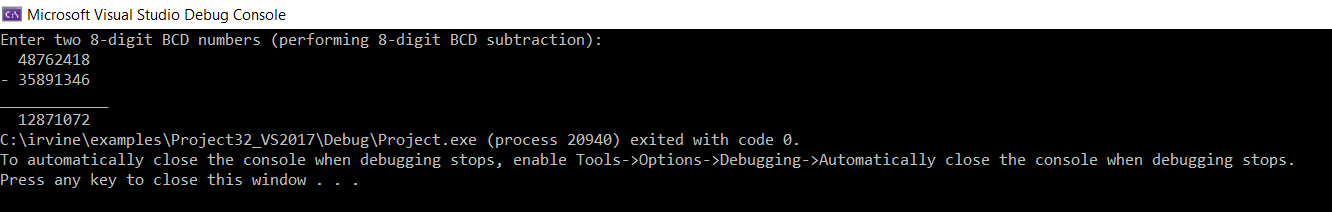
Please embed your code into your homework solution along with a screen shot post execution.











**Hw9-2.asm**

include irvine32.inc

.data

BCDnum1 DWORD ?

BCDnum2 DWORD ?

diff DWORD ?

prompt1 BYTE "Enter two 8-digit BCD numbers (performing 8-digit BCD subtraction): ", 0

extraSpace BYTE " ", 0

equalBar BYTE "\_\_\_\_\_\_\_\_\_\_\_\_", 0

subOP BYTE "- ", 0

negOP BYTE " -", 0

.code

main proc

;Initialize difference and index

mov diff, 0

mov esi, 0

mov edx, OFFSET prompt1

call WriteString

call crlf

mov edx, OFFSET extraSpace

call WriteString

call ReadHex

mov BCDnum1, eax

mov edx, OFFSET subOP

call WriteString

call ReadHex

mov BCDnum2, eax

mov edx, OFFSET equalBar

call WriteString

call Crlf

mov eax, BCDnum1

cmp eax, BCDnum2

JC Count

mov edx, OFFSET extraSpace

call WriteString

mov ecx, 4

L1:

mov al, BYTE PTR BCDnum1[esi]

sbb al, BYTE PTR BCDnum2[esi]

das

mov BYTE PTR diff[esi], al

inc esi

loop L1

mov eax, diff

call WriteHex

jmp return

Count: mov ecx, 4

clc

L2:

mov al, BYTE PTR BCDnum2[esi]

sbb al, BYTE PTR BCDnum1[esi]

das

mov BYTE PTR diff[esi], al

inc esi

loop L2

mov edx, OFFSET negOP

call WriteString

mov eax, diff

call WriteHex

return: exit

invoke ExitProcess, 0

main endp

end main

**hw9-2.lst**

hw9-2.asm Page 1 - 1

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 00000000 BCDnum1 DWORD ?

00000004 00000000 BCDnum2 DWORD ?

00000008 00000000 diff DWORD ?

0000000C 45 6E 74 65 72 prompt1 BYTE "Enter two 8-digit BCD numbers (performing 8-digit BCD subtraction): ", 0

20 74 77 6F 20

38 2D 64 69 67

69 74 20 42 43

44 20 6E 75 6D

62 65 72 73 20

28 70 65 72 66

6F 72 6D 69 6E

67 20 38 2D 64

69 67 69 74 20

42 43 44 20 73

75 62 74 72 61

63 74 69 6F 6E

29 3A 20 00

00000051 20 20 00 extraSpace BYTE " ", 0

00000054 5F 5F 5F 5F 5F equalBar BYTE "\_\_\_\_\_\_\_\_\_\_\_\_", 0

5F 5F 5F 5F 5F

5F 5F 00

00000061 2D 20 00 subOP BYTE "- ", 0

00000064 20 2D 00 negOP BYTE " -", 0

00000000 .code

00000000 main proc

;Initialize difference and index

00000000 C7 05 00000008 R mov diff, 0

00000000

0000000A BE 00000000 mov esi, 0

0000000F BA 0000000C R mov edx, OFFSET prompt1

00000014 E8 00000000 E call WriteString

00000019 E8 00000000 E call crlf

0000001E BA 00000051 R mov edx, OFFSET extraSpace

00000023 E8 00000000 E call WriteString

00000028 E8 00000000 E call ReadHex

0000002D A3 00000000 R mov BCDnum1, eax

00000032 BA 00000061 R mov edx, OFFSET subOP

00000037 E8 00000000 E call WriteString

0000003C E8 00000000 E call ReadHex

00000041 A3 00000004 R mov BCDnum2, eax

00000046 BA 00000054 R mov edx, OFFSET equalBar

0000004B E8 00000000 E call WriteString

00000050 E8 00000000 E call Crlf

00000055 A1 00000000 R mov eax, BCDnum1

0000005A 3B 05 00000004 R cmp eax, BCDnum2

00000060 72 31 JC Count

00000062 BA 00000051 R mov edx, OFFSET extraSpace

00000067 E8 00000000 E call WriteString

0000006C B9 00000004 mov ecx, 4

00000071 L1:

00000071 8A 86 00000000 R mov al, BYTE PTR BCDnum1[esi]

00000077 1A 86 00000004 R sbb al, BYTE PTR BCDnum2[esi]

0000007D 2F das

0000007E 88 86 00000008 R mov BYTE PTR diff[esi], al

00000084 46 inc esi

00000085 E2 EA loop L1

00000087 A1 00000008 R mov eax, diff

0000008C E8 00000000 E call WriteHex

00000091 EB 30 jmp return

00000093 B9 00000004 Count: mov ecx, 4

00000098 F8 clc

00000099 L2:

00000099 8A 86 00000004 R mov al, BYTE PTR BCDnum2[esi]

0000009F 1A 86 00000000 R sbb al, BYTE PTR BCDnum1[esi]

000000A5 2F das

000000A6 88 86 00000008 R mov BYTE PTR diff[esi], al

000000AC 46 inc esi

000000AD E2 EA loop L2

000000AF BA 00000064 R mov edx, OFFSET negOP

000000B4 E8 00000000 E call WriteString

000000B9 A1 00000008 R mov eax, diff

000000BE E8 00000000 E call WriteHex

000000C3 return: exit

000000C3 6A 00 \* push +000000000h

000000C5 E8 00000000 E \* call ExitProcess

invoke ExitProcess, 0

000000CA 6A 00 \* push +000000000h

000000CC E8 00000000 E \* call ExitProcess

000000D1 main endp

end main