Prog # 1

Demetrius Johnson

CIS 310 (Yoon)

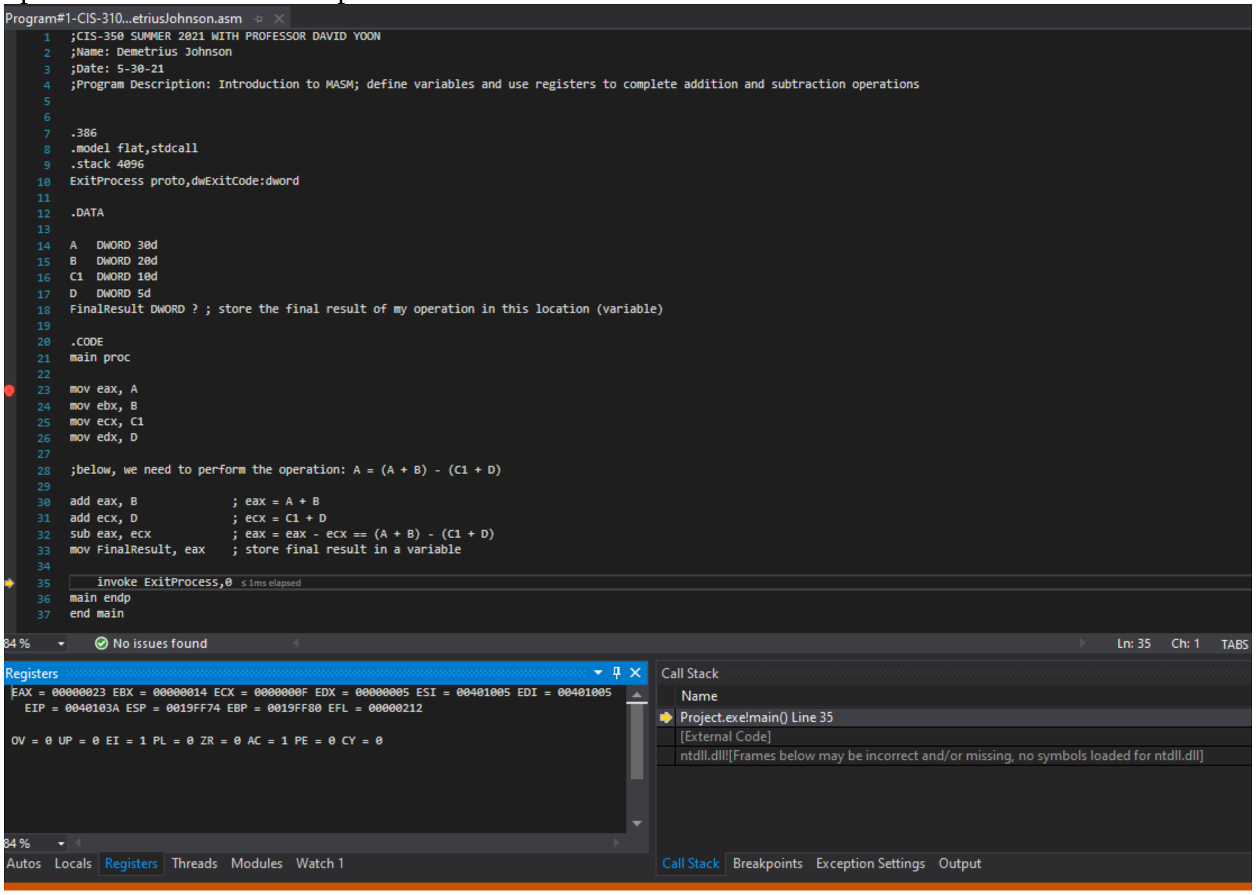
June 2, 2021

CIS 310 – YOON – SUMMER 21 – PROGRAM #1 – DEMETRIUS JOHNSON

1. **Build/run the program without debugging.**

2. **Run the program in the debugging mode and add the register window to a debugging session (see Fig. 3.5).**

Notice, I added another variable to store the result of the arithmetic operation, called **FinalResult**; also in this screenshot I show what the value of all of my registers are after all operations have been completed:



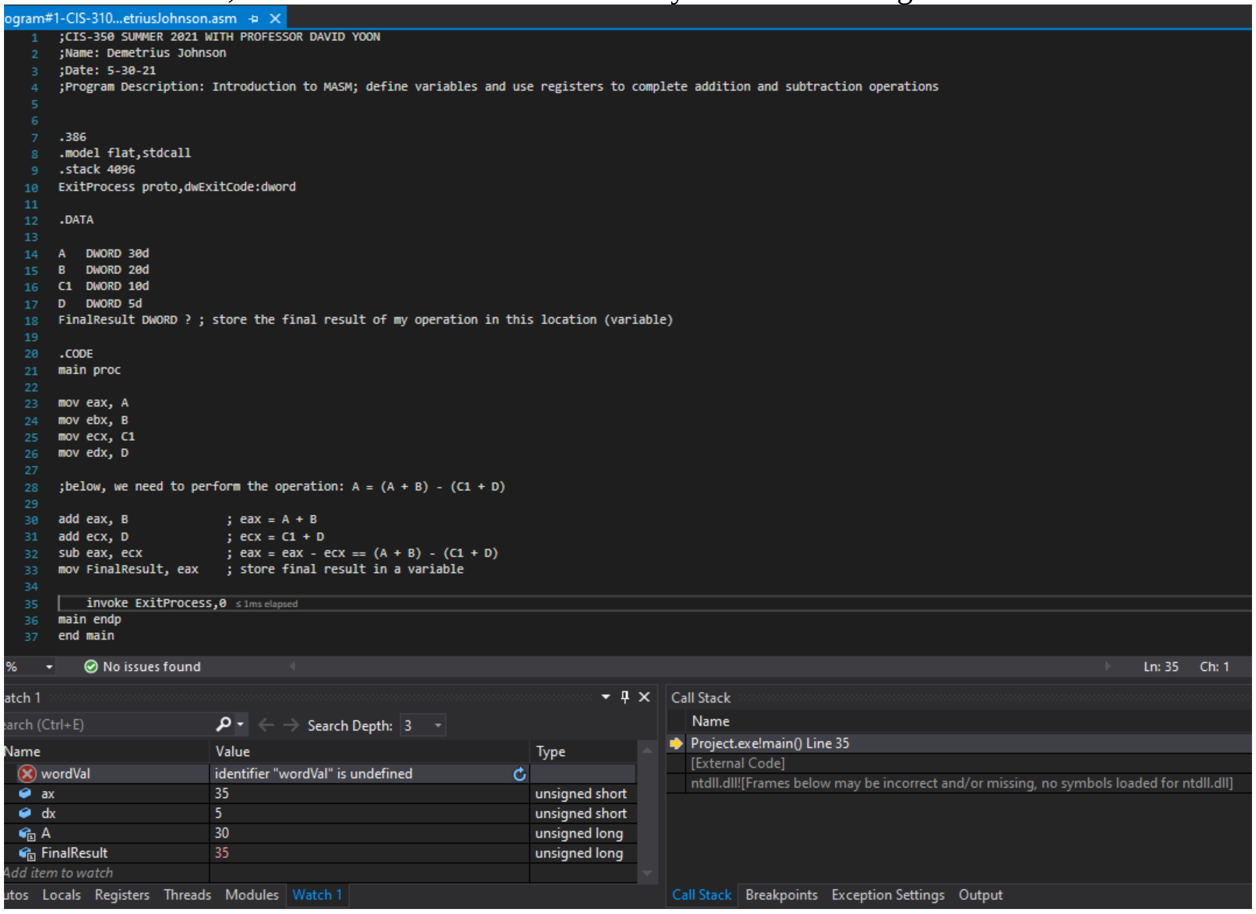
```
;CIS-350 SUMMER 2021 WITH PROFESSOR DAVID YOON
;Name: Demetrius Johnson
;Date: 5-30-21
;Program Description: Introduction to MASM; define variables and use registers to complete addition and subtraction operations


.386
.model flat,stdcall
.stack 4096
ExitProcess proto,dwExitCode:dword

.DATA

A   DWORD 30d
B   DWORD 20d
C1  DWORD 10d
D   DWORD 5d
FinalResult DWORD ? ; store the final result of my operation in this location (variable)

.CODE
main proc

mov eax, A
mov ebx, B
mov ecx, C1
mov edx, D

;below, we need to perform the operation: A = (A + B) - (C1 + D)

add eax, B              ; eax = A + B
add ecx, D              ; ecx = C1 + D
sub eax, ecx            ; eax = eax - ecx == (A + B) - (C1 + D)
mov FinalResult, eax    ; store final result in a variable

    invoke ExitProcess,0  ≤ 1ms elapsed
main endp
end main
```

```
EAX = 00000023 EBX = 00000014 ECX = 0000000F EDX = 00000005 ESI = 00401005 EDI = 00401005
  EIP = 0040103A ESP = 0019FF74 EBP = 0019FF80 EFL = 00000212

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 1 PE = 0 CY = 0
```

```
Project.exe!main() Line 35
[External Code]
ntdll.dll![Frames below may be incorrect and/or missing, no symbols loaded for ntdll.dll]
```

CIS 310 – YOON – SUMMER 21 – PROGRAM #1 – DEMETRIUS JOHNSON

In this screenshot, I demonstrate what is stored in my variables and registers:



ogram#1-CIS-310...etriusJohnson.asm

```
;CIS-350 SUMMER 2021 WITH PROFESSOR DAVID YOON
;Name: Demetrius Johnson
;Date: 5-30-21
;Program Description: Introduction to MASM; define variables and use registers to complete addition and subtraction operations


.386
.model flat,stdcall
.stack 4096
ExitProcess proto,dwExitCode:dword

.DATA

A   DWORD 30d
B   DWORD 20d
C1  DWORD 10d
D   DWORD 5d
FinalResult DWORD ? ; store the final result of my operation in this location (variable)

.CODE
main proc

mov eax, A
mov ebx, B
mov ecx, C1
mov edx, D

;below, we need to perform the operation: A = (A + B) - (C1 + D)

add eax, B              ; eax = A + B
add ecx, D              ; ecx = C1 + D
sub eax, ecx            ; eax = eax - ecx == (A + B) - (C1 + D)
mov FinalResult, eax    ; store final result in a variable

    invoke ExitProcess,0  ≤ 1ms elapsed
main endp
end main
```

No issues found   Ln: 35   Ch: 1

atch 1

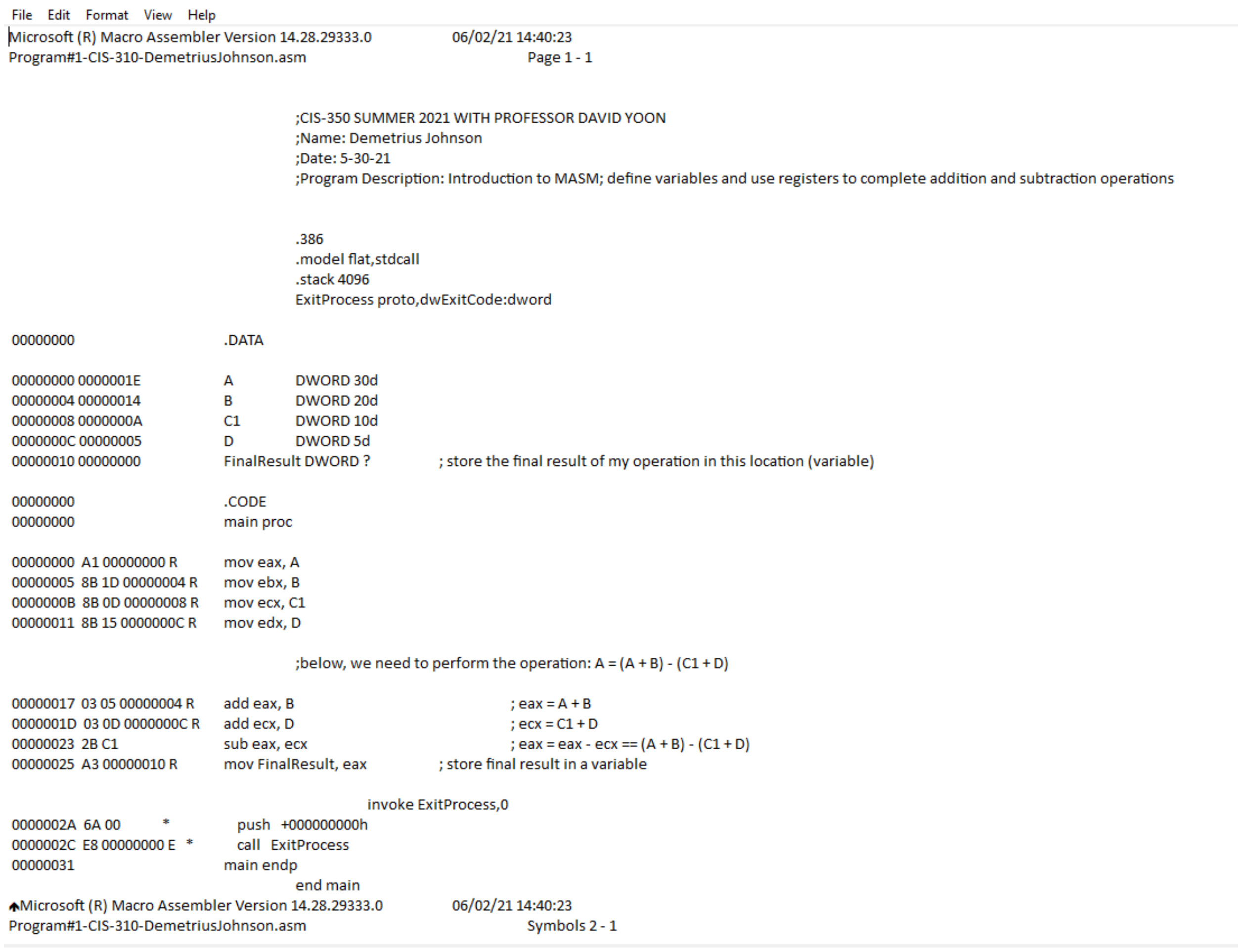
Name	Value	Type
wordVal	identifier "wordVal" is undefined	
ax	35	unsigned short
dx	5	unsigned short
A	30	unsigned long
FinalResult	35	unsigned long
Add item to watch		

Call Stack

Name
Project.exe!main() Line 35
[External Code]
ntdll.dll![Frames below may be incorrect and/or missing, no symbols loaded for ntdll.dll]

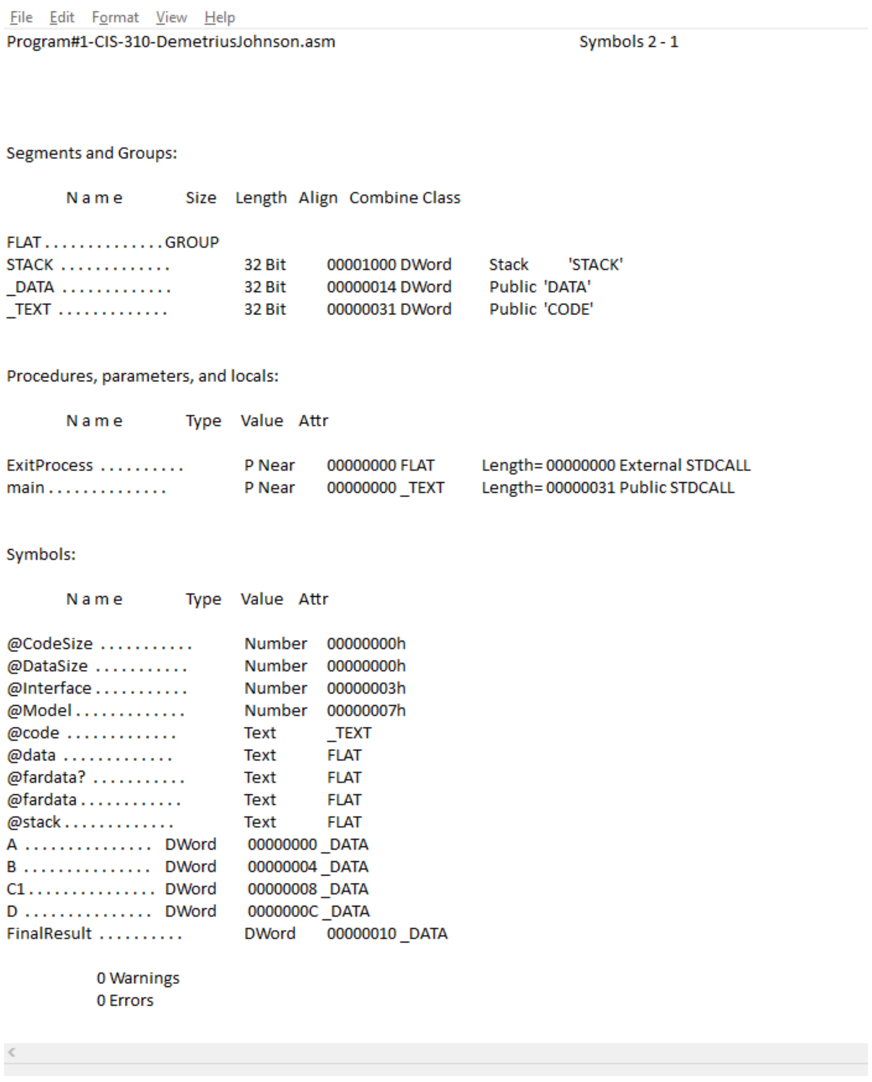
3. Generate the listing file like the one in Fig. 3.8.

In the next two screenshots, you see the listing file generated from my program. I read from the Irvine book that the memory locations correspond to some starting address relative to wherever the operating system begins assigning memory; as an example, the start memory location 00000000 (32-bit address represented as 8 hex digits) is really some other starting location value in memory, and thus every value after the start is also relative to that location. They simply generate the listing file starting from a relative 0-address location so it is easier for us programmers to identify and analyze memory usage and allocation that occurs for and during the program:



```
Microsoft (R) Macro Assembler Version 14.28.29333.0          06/02/21 14:40:23
Program#1-CIS-310-DemetriusJohnson.asm                                   Page 1 - 1


                                        ;CIS-350 SUMMER 2021 WITH PROFESSOR DAVID YOON
                                        ;Name: Demetrius Johnson
                                        ;Date: 5-30-21
                                        ;Program Description: Introduction to MASM; define variables and use registers to complete addition and subtraction operations


                                        .386
                                        .model flat,stdcall
                                        .stack 4096
                                        ExitProcess proto,dwExitCode:dword

 00000000                         .DATA

 00000000 0000001E                A          DWORD 30d
 00000004 00000014                B          DWORD 20d
 00000008 0000000A                C1         DWORD 10d
 0000000C 00000005                D          DWORD 5d
 00000010 00000000                FinalResult DWORD ?          ; store the final result of my operation in this location (variable)

 00000000                         .CODE
 00000000                         main proc

 00000000  A1 00000000 R          mov eax, A
 00000005  8B 1D 00000004 R       mov ebx, B
 0000000B  8B 0D 00000008 R       mov ecx, C1
 00000011  8B 15 0000000C R       mov edx, D

                                        ;below, we need to perform the operation: A = (A + B) - (C1 + D)

 00000017  03 05 00000004 R       add eax, B                            ; eax = A + B
 0000001D  03 0D 0000000C R       add ecx, D                            ; ecx = C1 + D
 00000023  2B C1                  sub eax, ecx                          ; eax = eax - ecx == (A + B) - (C1 + D)
 00000025  A3 00000010 R          mov FinalResult, eax         ; store final result in a variable

                                              invoke ExitProcess,0
 0000002A  6A 00         *          push   +000000000h
 0000002C  E8 00000000 E  *          call   ExitProcess
 00000031                         main endp
                                        end main
Microsoft (R) Macro Assembler Version 14.28.29333.0          06/02/21 14:40:23
Program#1-CIS-310-DemetriusJohnson.asm                                   Symbols 2 - 1
```

CIS 310 – YOON – SUMMER 21 – PROGRAM #1 – DEMETRIUS JOHNSON



```
Program#1-CIS-310-DemetriusJohnson.asm                    Symbols 2 - 1



Segments and Groups:

        N a m e          Size   Length  Align  Combine Class

FLAT . . . . . . . . . . . . . .GROUP
STACK . . . . . . . . . . . . .          32 Bit    00001000 DWord    Stack     'STACK'
_DATA . . . . . . . . . . . . .          32 Bit    00000014 DWord    Public  'DATA'
_TEXT . . . . . . . . . . . . .          32 Bit    00000031 DWord    Public  'CODE'


Procedures, parameters, and locals:

        N a m e          Type   Value   Attr

ExitProcess . . . . . . . . . .          P Near    00000000 FLAT      Length= 00000000 External STDCALL
main . . . . . . . . . . . . . .         P Near    00000000 _TEXT     Length= 00000031 Public STDCALL


Symbols:

        N a m e          Type   Value   Attr

@CodeSize . . . . . . . . . . .          Number    00000000h
@DataSize . . . . . . . . . . .          Number    00000000h
@Interface . . . . . . . . . . .         Number    00000003h
@Model . . . . . . . . . . . . .         Number    00000007h
@code . . . . . . . . . . . . .          Text      _TEXT
@data . . . . . . . . . . . . .          Text      FLAT
@fardata? . . . . . . . . . . .          Text      FLAT
@fardata . . . . . . . . . . . .         Text      FLAT
@stack . . . . . . . . . . . . .         Text      FLAT
A . . . . . . . . . . . . . . .  DWord    00000000 _DATA
B . . . . . . . . . . . . . . .  DWord    00000004 _DATA
C1 . . . . . . . . . . . . . . .  DWord    00000008 _DATA
D . . . . . . . . . . . . . . .  DWord    0000000C _DATA
FinalResult . . . . . . . . . .          DWord     00000010 _DATA

           0 Warnings
           0 Errors
```

Upload the screenshots from 1, 2 and 3.