HW # 3: Theme: Data declarations using assembler directives, Small program

1. Declare the following:

   A. An un-initialized data declaration for an 8-bit signed integer

      num SBYTE ?

   B. An initialized data declaration for a 32-bit unsigned integer with the value "1359h"

      num DWORD 1359h

   C. A null terminated string variable containing your name and favorite color

      info BYTE "Seth Denney - Green"

   D. A symbolic constant named "*SecondsInAWeek*" using the equal-sign directive and assign it an arithmetic expression that calculates the total number of seconds in a week

      SecondsInAWeek = 7 * 24 * 60 * 60

2. Show the order of individual bytes in memory ***(lowest to highest)*** for the following Double word variable *(use little endian order):* var1 DWORD 87654321h

   21 – 43 – 65 – 87

3. Show the following using assembly instructions:

   A. How to declare an signed double-word array of five elements and initialize the array with the following values: 36h, 36h, 41h, 49h, 52h

      signArray SDWORD 36h, 36h, 41h, 49h, 52h

   B. Using the array created in part A of this question, show how to calculate the number of elements contained and assign the value to a symbolic constant named "*ArraySize*"

      ArraySize = ($ - signArray) / 4

4. Why is a string variable declared using the reserved word BYTE as opposed to WORD, DWORD or QWORD?

   String variables are declared as BYTE because, regardless of length, each string is an array of ASCII characters, each of which is a BYTE.

5. What is the linear address corresponding to the following segment-offset value: 86E8:1A08

   86E8h * 10h = 86E80h 86E80h + 1A08h = 88888h = 20-bit linear address

6. Small program: Using the *AddSub* Program from Section 3.2 as a reference, write a program that adds three integers using only ***16-bit registers***. Please embed program code into your homework submission.

```
TITLE Homework1                  (HW1.asm)

; This program adds three integers using only 16-bit registers.

.code
main PROC

        MOV ax, 1000h            ; sets the 16-bit version of eax to 1000h
        MOV ax, 2000h            ; adds 2000h to ax register
        MOV ax, 3000h            ; adds 3000h to ax register

        exit
main ENDP
END main
```