1. Declare the following:
2. An un-initialized data declaration for a 32-bit unsigned integer

**Var1 DWORD ?**

1. An initialized data declaration for a 32-bit unsigned integer with the value “1234h”

**Var2 DWORD 1234h**

1. A null terminated string variable containing your name and favorite vegetable

**Var1 BYTE “Walter Conway Collard Greens”,0**

1. A symbolic constant named “*MinutesInMonth*” using the equal-sign directive and assign it an arithmetic expression that calculates the total number of minutes in a month.

**MinutesInMonth = (31\*24)\*60**

1. Show the order of individual bytes in memory *(lowest to highest****)*** for the following double word variable *(use little endian order****):*** var1 DWORD 33129887h

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **22** | **33** | **12** | **98** | **87** |

**Little endian order**

|  |  |
| --- | --- |
| **0000** | **87** |
| **0001** | **98** |
| **0002** | **12** |
| **0003** | **33** |

1. Show the following using assembler directives:
2. How to declare a signed double-word array of eight elements and initialize the array with the following values: 43h, 32h, 25h, 12h, 99h, 44h, 77h, 88h

**Array SDWORD 43h,32h,25h,12h,99h,44h,77h,88h**

1. 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 = ($ - Array) / 4**

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

Each character uses a byte of storage. The string variable can be thought as an array of characters.

Ex.   
feeling1 BYTE “Good”

feeling2 BYTE ‘G’, ‘o’, ‘o’, ‘d’

feeling1 and feeling2 are the same. Since there is an exception with defining strings we can use feeling1 method.

1. Using the *AddSub* program from the textbook as a reference, write a program that adds three word sized integers using only *16-bit registers*. Please embed program code into your homework submission.

TITLE Add three integers

; This program add three word sized integers using only 16-bit registers.

INCLUDE Irvine32.inc

.data

integer1 WORD 1000h

integer2 WORD 2000h

integer3 WORD 3000h

.code

MAIN PROC

mov ax,integer1 ; moves 1000h to the 16bit portion of EAX to result in 1000h

add ax,integer2 ; adds 2000h to the 16bit portion of EAX to result in 3000h

add ax,integer3 ; adds 3000h to the 16bit portion of EAX to result in 6000h

exit

main ENDP

END main

1. Write a program that prints *War Eagle* on your screen. You can use the following. Assemble and generate the output using MASM and Visual Studio. Embed your output in your submission.

Sourcecode:

Output:

WAR EAGLE

Press any key to continue . . .

TITLE PRINT WAR EAGLE

COMMENT !

This program prints "WAR EAGLE" on the screen

with a carriage return and line feed.

!

INCLUDE Irvine32.inc

.data

message BYTE "WAR EAGLE",0dh,0ah,0

.code

main PROC

mov edx, offset message

Call WriteString

exit

main ENDP

END main