Michael Gil

**Assignment 3**

**3.1.11 Section Review**

1. -35d, DDh, 335o, 11011101b.

2. No. hexadecimal literal beginning with a letter must have a leading zero to prevent the assembler from interpreting it as an identifier.

3. No, same precedence.

4. (1 + 2) - (2 \* 2) + (1 / 1) + (5 MOD 2) = 1

**3.2.4 Section Review**

4. EAX

5. The exitprocess statement.

**3.3.3 Section Review**

1. The object file and listing file.

2. True.

3. True.

4. Loader.

**3.4.13 Section Review**

1. var1 SWORD ?

2. var2 BYTE ?

3. var3 SBYTE ?

4. var4 QWORD ?

5. SDWORD

**3.5.5 Section Review**

3. ArraySize = ($ - myArray)

**3.9.1 Section Review**

Short Answer:

4. Assemblers are programs that translate source code. Assembly language would be more specific and correct.

5. Big and little-endian describe the order in which a sequence of bytes are stored in computer memory. Big-endian is the order where the most significant value is stored first in the lowest storage address. Little-endian is an order where the least significant value is stored first.

6. integer literals have no meaning to anyone but the coder. symbols are self documenting.

25. DD.

**3.9.2 Section Review**

4. SDWORD is for signed, not DWORD. The type checking helps to avoid possible errors as mismatching variables and data.

7. myArray DWORD 120 DUP(?)

13. STR DWORD 500 DUP("TEST").

**3.10 Programming exercise #2**

; 3.10 Programming Exercises Question 2. Symbolic Integer Constants

.386

.model flat, stdcall

.stack 4096

ExitProcess PROTO, dwExitCode:DWORD

; symbolic constants for days of week

.data

Sun=0

Mon=1

Tue=2

Wed=3

Thu=4

Fri=5

Sat=6

warray BYTE Sun, Mon, Tue, Wed, Thu, Fri, Sat

.code

main PROC

INVOKE ExitProcess, 0

main ENDP

END main