**Andrew Tran**

**Computer Science 2253 Lab Experience Three**

Objectives:

1. Creating symbolic names for string literals
2. Understanding the built-in data types.
3. Understanding the Assembler environment.

The purpose of this lab is for you to create your own assembly language instructions. I recommend you load a project solution into Visual C++ express/studio so you do not have to reset the environment variables each time you are to create an assembler program.

NOTE: If you have changed the location of the Irvine libraries, you will have to correct the project properties settings of the project so Visual Studio can locate these files. The authors website contains detail instructions on this is accomplished.

**What you are to do:**

**Exercise 1:**

The following problems pertain to chapter three in your textbook. Place your answers to each question below the question your word document.

1. Deﬁne four symbolic constants that represent integer 25 in decimal, binary, octal, and hexadecimal formats.

Decimal: 25

Binary: 11001

Octal: 31

Hexadecimal: 19

1. Find out, by trial and error, if a program can have multiple code and data segments.

They cannot.

1. Find out if you can declare a variable of type DWORD and assign it a negative value. What does this tell you about the assembler’s type checking?

You can declare it as a negative value, so assembler doesn’t actually check types.

1. Given the number 456789ABh, list out its byte values in little-endian order.

AB 89 67 45

1. Declare an array of 120 uninitialized unsigned doubleword values.

HELLO DWORD 120 DUP(?)

1. Declare an array of byte and initialize it to the ﬁrst 5 letters of the alphabet.

hello BYTE “ABCDE”

1. Declare a 32-bit signed integer variable and initialize it with the smallest possible negative decimal value. (Hint: Refer to integer ranges in Chapter 1.)

hello SDWORD -2147483648

1. Declare an unsigned 16-bit integer variable named wArray that uses three initializers.

wArray WORD 1,2,3

1. Declare a string variable containing the name of your favorite color. Initialize it as a null terminated string.

favcolor BYTE “blue”

1. Declare an uninitialized array of 50 signed doublewords named dArray.

dArray DWORD 50 DUP(?)

1. Declare a string variable containing the word “TEST” repeated 500 times.

BYTE 500 DUP(“TEST”)

1. Declare an array of 20 unsigned bytes named bArray and initialize all elements to zero.

bArray BYTE 20 DUP(0)

**Exercise 2:**

Do problem one on page 94 in your textbook. Use the statement: call dumpregs in your program to verify your program works. Choose values for the variables listed in the problem statement.

Copy and paste your .asm file in your word document along with a snapshot of the console window showing the execution of your program.

include Irvine32.inc

.data

w dword 25

x dword 50

y dword 10

z dword 5

.code

main PROC

mov eax, w

mov ebx, x

mov ecx, y

mov edx, z

add eax, ebx; w + x

add ecx, edx; y + z

sub eax, ecx; subtracts them together

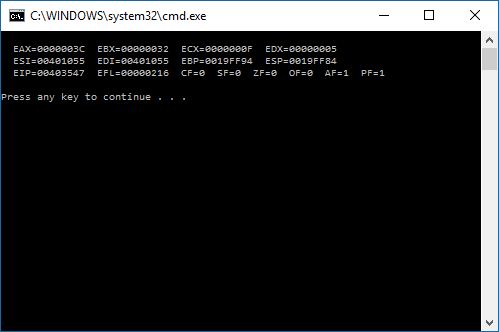
mov w, eax

call DumpRegs; print variables

invoke ExitProcess, 0

main endp

end main



**What to hand in:**

1. Compress all .asm file and your word document into a single file called {yourName}Lab3.zip. For example TimWrennLab3.zip.
2. Place the compressed file into the D2L DropBox labeled Lab3.
3. Hand-in print outs of your word document and your .asm file you created in exercises in 1 and 2.

**Due Date: At the start of next week’s lab session.**