CS 218
Homework, Asst. #6

Purpose:	Become familiar with data conversion, addressing modes, and assembly language macro's.
Due:	Thursday (2/14)
Points:	80

Background:

The duodecimal[1] system (also known as base-12 or *dozenal*) is a positional notation numeral system using twelve as its base. The number twelve (that is, the number written as "12" in the base ten numerical system) is instead written as "10" in duodecimal (meaning "1 dozen and 0 units", instead of "1 ten and 0 units"), whereas the digit string "12" means "1 dozen and 2 units" (i.e. the same number that in decimal is written as "14"). The symbol X_{12} (or x) is used for 10_{10} and E_{12} (or e) is used for 11_{10}. For example.

base-10	1	2	3	4	5	6	7	8	9	10	11	12
base-12	1	2	3	4	5	6	7	8	9	X	E	10

The number twelve is the smallest number with four non-trivial factors (2, 3, 4, 6). As a result of this increased factorability and its divisibility, duodecimal representations fit more easily than decimal ones into many common patterns. As a result, duodecimal is sometimes considered the most optimal numbering system. The Dozenal Society of America[2] promotes research and education regarding the use of base twelve.

Assignment

Write an assembly language program to convert duodecimal/ASCII string to integers and integers to duodecimal/ASCII strings. Using the provided main, the program has four steps are follows:

1. Write the code to convert a string of ASCII digits representing an dozenal value into an integer (double-word sized). This code should be placed in the provided main at the marked location (step #1) and will convert the string **`dStr1`** (dozenal representation) into an integer stored in the variable **`iNum1`**.
2. Write the code to convert an integer into a string of ASCII digits representing the dozenal value (NULL terminated). This code should be placed in the provided main at the marked location (step #2) and will convert the integer stored in the variable **`iNum1`** into a string **`num1String`** (dozenal representation).
3. Convert the code from step #1 into a macro, ***dozenal2int***, which is called multiple times in the next part of the provided main.
4. Convert the code from step #2 into a macro, ***int2dozenal***, which is called multiple times in the next part of the provided main.

You may assume valid/correct data. As such, no error checking is required.

The provided main will also invoke a print macro, which will display the strings to the screen. The print macro does not perform any error checking, so the data must be correct in order for the display to work. *Note*, since the program displays the results to the screen, typing the program name (without the debugger), will display the results to the screen.

1 For more information regarding base-12 representation, refer to: http://en.wikipedia.org/wiki/Duodecimal
2 For more information regarding the DSA, refer to: http://www.dozenal.org

Debugging

Since macro's can be difficult to debug, the code for steps 1 and 2 should be working before attempting steps 3 and 4.

The code for a macro will not be displayed in the source window. In order to see the macro code, display the machine code window (**View → Machine Code Window**). In the window, the machine code for the instructions are displayed. The step and next instructions will execute the entire macro. In order to execute the macro instructions, the **stepi** and **nexti** commands must be used.

To help check results, an on-line conversion is available at the following URL: http://www.cut-the-knot.org/binary.shtml. *Note*, the on-line tool uses A=101 and B=11.

Submission:

When complete, submit:

- A copy of the ***source file*** via the class web page (assignment submission link) by class time (Section 001, 10:00am and Section 002, 4:00pm). Assignments received after the due date/time will not be accepted.

Example Output:

The results, as displayed to the screen, would be as follows:

```
--------------------------------------------
CS 218 - Assignment #6


Test Number (base-12):      +12X4E
Number * -2 (base-12):      -2589X

---------------------------
Final Results for List Sums

List Sum:
     -6143E

List Sum:
   +1XE1367
```

Note, since this program displays output to the screen, when working it can be executed without the debugger. The results will be displayed as shown above.

Debugger Commands:

Below is an example of some of the commands to display the variables within DDD.

```
x/dw &iNum1
x/s &num1String
```

Note, in DDD, select **View → Execution Window** to display a window that shows the output.