```
############################################################
#           cs315 Week 5 - part 4
#
#   ->  2's complement multiplication and practice examples
#
############################################################

2's complement multiplication:
    * MUST sign extend both values to 2 * n bits
    * 2's sign extension review:  copy sign bit left to required number of bits

Ex: 4 bits to 8 bits, 1001  --> 11111001

* although multiplication is commutative, values MAY NOT be swapped on homework or exam problems
    Ex:
        12 * 8 = 8 * 12, but if a problem is given as 12 * 8, it must be solved as 12 * 8, not as 8 * 12

* multiplication process:
    1. long-hand multiply to create 2n rows (Familiar process, same process as base 10, easier demonstrated through example than explained with words):
        A. Start at bit 0
        B. Multiply top value with corresponding bit in bottom value
        C. Pad n-1 zeros for the nth row
        D. Repeat from A for all 2n rows
    2. From left to right:
        A. Add a column AS BASE 10
        B. Place ENTIRE sum in the column result
        C. Divide by sum by 2, carry the QUOTIENT to the next column
        D. Repeat from A

Note: result will be 2n bits at most, so there is no need to carry past the 2nth column

    3. convert result to binary
        * Take each column mod 2 as the result bit
            * Even value    --> bit is 0
            * odd value     --> bit is 1

* use grid paper to keep rows and columns aligned
* do not take a shortcut and convert a column sum to binary until ALL columns have been calculated
* leaving column sums in base 10 allows us to retrace our work if a mistake is made.  Converting to binary too early does not allow this.

How to get good at multiplication:
    * practice, practice, practice
    * practice often
    * practice many times


############################################################
2's complement multiplication (example 1):

     23 =>  010111

    -15 =>  001111
            110000
                 1
          ---------
            110001  <= -15

    (+23) * (-15) = -345 (expected)

    Note:
        sign extend to 12 bits (2 * 16 bits)
```

```
            000000 010111
        *   111111 110001
        ------------------

    6 bit * 6 bit = 12 bits result

    carry:  333221 1
            000000 010111
            000000 000000
            000000 000000
            000000 000000
            000101 110000
            001011 100000
            010111 000000
            101110 000000
            011100 000000
            111000 000000
            110000 000000
    +       100000 000000
            -------------
            777654 320111

    %2      111010 100111 => -345 (correct)

            111010 100111
            000101 011000
                        1
            --------------
            000101 011001 => 1+8+16+64+256 = +345

#############################################################
2's complement multiplication (example 2):

    10 =>   001010

    -8 =>   001000
            110111
                 1
          ---------
            111000  <= -8

    (+10) * (-8) = -80 (expected)

    Note:
        sign extend to 12 bits (2 * 16 bits)

            000000 001010
        *   111111 111000
        ------------------

    6 bit * 6 bit = 12 bits result

    carry:  11111
            000000 000000
            000000 000000
            000000 000000
            000001 010000
            000010 100000
            000101 000000
            001010 000000
            010100 000000
            101000 000000
            010000 000000
```

```
            100000 000000
            000000 000000
            --------------
            333332 110000

    %2      111110 110000 => -80 (correct)

############################################################
```