CSC 230 Assignment 5
Fall 2017

This assignment is an AVR Assembly Language programming assignment needs to be tested on the Arduino 2560 boards in the ECS 249 lab. There are two parts to this assignment:

Part 1: Submit one program that solves Programming Problem #2 below (and uses Programming Problem #1) via connex before 8 am on Tuesday, November 7, 2017.

Programming Problem #1: 16-bit add (macro)

(Similar to #5 on Page 239) Define a macro that takes 4 arguments in 4 registers representing two 16-bit integers and places the result of adding them together into 2 of the registers. In particular, the macro will create a pseudo-instruction with the following format:

```
Addw ah, al, bh, bl
```

It represents the operation `ah:al` ← `ah:al + bh:bl`. Include concise documentation that describes the macro

Programming Problem #2: Multiply: (8-bit) byte x (8-bit) byte (function)

(Similar to #5 on Page 239) Write a function to multiply two bytes (8-bits each). The function arguments are to be passed on the stack. The return value, a (2-byte) word, is to be left in registers R25:R24. The multiplication algorithm is repeated addition:

```
word multiply (byte factor, byte multiplier) {
     word answer = 0;
     while (factor--  > 0) answer += multiplier;
     return answer;
}
```

This function *must* use the macro written in problem #1 (above). Observe that the documentation of this subroutine really should include the algorithm's pseudo-code (above).

Part 2: Submit one program that solves Programming Problem #4 below (and uses Programming Problem #3) via connex before 8 am on Friday, November 17, 2017.

Programming Problem #3: Factorial: (16-bit) word x byte (function)

(Similar to #6 on Page 240) Write a recursive function to calculate n-factorial. Pass the argument, *n*, on the stack. Assume that n is in the range $1 \leq n \leq 6$ ~~8~~. The multiply function (above) must be called by this function to perform the multiplication operation.

Programming Problem #4: Full Program

(Similar to #6 on Page 240) Write program that calculates *n!* using the function above. The input number, *n*, will be found in a program memory location called `init`. The output will be stored into a data memory location called `result`. In addition, the lower nibble of the result should be output on the 4 LEDs that are attached to Port L (bits 1,3,5 and 7).

Please note for all programing assignments, ***you must include your name and student number*** and ***edit the documentation*** appropriately***.*** Make sure the submitted work is yours and not someone else. It is expected that each of the above solutions build on the previous. The final submission should include all parts. A shell program is provided for this final submission: `A5_factorial.asm`. Add your code and documentation to the shell.