Homework Assignment #2

Due: **March 4, 2025, before midnight (11:59 pm)**

1. Consider the following two loop examples:

Loop example #1	**Loop example #2**

`MOV #3, W3`
`LOOP:` 2 cyc
`BTG PORTB, #0` 1 cyc
`DEC W3` 1 cyc
`BRA Z, END_LOOP` 1 (2) cyc
`BRA LOOP` 1 (2) cyc
`END_LOOP:`
`MOV …`
	`MOV #3, W3`
	`LOOP:` 2 cyc
	`BTG PORTB, #0` 1 cyc
	`DEC W3` 1 cyc
	`BRA NZ, LOOP` 1 (2) cyc
	`MOV …`

a. Briefly explain what these two loop examples perform in one sentence. Do these two examples perform the same operation?

Both of these codes toggle the Last bit of Port B Register (RB0) 3 times

b. Compare the total number of instruction cycles required to execute the loop. Don't count the first and last `MOV` instructions.

20 instruction cycles for example #1
15 instruction cycles for example #2

c. Based on the total number of instruction cycles, which loop example would you choose?

I would choose example #2 due to fewer instruction cycle requirement

2. **Implement** an assembly loop that calculates the following mathematical operations and saves the result to W4. **Report** the value of W4 after the loop.

   a. $W4 = \sum_{i=1}^{50} i$ Turn in an ASM code (`hw2_p2_a.s`).

   WREG4 = 1275

   b. $W4 = \sum_{i=0}^{10} (2 \times i + 1)$ Turn in an ASM code (`hw2_p2_b.s`).

   WREG = 121

   c. $W4 = \sum_{i=-100}^{-1} i$ Turn in an ASM code (`hw2_p2_c.s`).

   WREG4 = -5050
   or
   WREG4 = 0xEC46

d. $W4 = \sum_{i=-10}^{10} i$ Turn in an ASM code (`hw2_p2_d.s`).

WREG4 = 0

e. $W4 = \sum_{i=1}^{10} i^2$ Turn in an ASM code (`hw2_p2_e.s`).

WREG4 = 385

f. $W4 = \sum_{i=1}^{10} 2^i$ Turn in an ASM code (`hw2_p2_f.s`).

WREG4 = 2046

3. Implement an assembly loop that calculates $\sum_{i=1}^{10} 4 \times i$ and saves the result to W4.

   a. For multiplication, use `MUL` instruction. Turn in an ASM code (`hw2_p3_a.s`).

W4 = 220

   b. For multiplication, use `SL` instruction. Turn in an ASM code (`hw2_p3_b.s`).

W4 = 220

   c. Don't use any multiplication operations. Turn in an ASM code (`hw2_p3_c.s`).

W4 = 220

4. Fill in the contents of the registers and data memory after the execution of the following instructions.

a.
```
BTG         W1, #10
BTG         W1, #3
BTG         W1, #9

W1 [ 0xABCD ]    ⇒    W1 [ 0xADC5 ]
```
1010 1011 1100 1101 → 1010 1101 1100 0101

b.
```
PUSH        W1
PUSH        W2
POP         W1
POP         W2

W1  [ 0x00AA ]  ⇒    W1  [ 0xAA00 ]
W2  [ 0xAA00 ]  ⇒    W2  [ 0x00AA ]
W15 [ 0x0812 ]  ⇒    W15 [ 0x0812 ]

Data Memory
0x0810 [ 0x1234 ]    ⇒    0x0810 [ 0x1234 ]
0x0812 [ 0x5678 ]    ⇒    0x0812 [ 0x00AA ]
0x0814 [ 0x9ABC ]    ⇒    0x0814 [ 0xAA00 ]
```

c.
```
REPEAT      #1
BTG         W1, #1
REPEAT      #3
BTG         W1, #3
REPEAT      #5
BTG         W1, #8
REPEAT      #7
BTG         W1, #12

W1 [ 0xABCD ]    ⇒    W1 [ 0xABCD ]
```
Toggling is repeated in odd numbers, which means no effect will be seen

d. `0x020204  MOV  [W15+0x2], W3`
`0x020206  RCALL FOO`
Assume that the subroutine FOO is located at the program address `0x018110`.

```
PC [ 0x020204 ] ⇒    PC [ 0x018110 ]
W3   [ 0x0017 ] ⇒    W3   [ 0x5678 ]
W15  [ 0x0818 ] ⇒    W15  [ 0x0818 ]

Data Memory
0x0818 [ 0x1234 ]    ⇒    0x0818 [ 0x1234 ]
0x081A [ 0x5678 ]    ⇒    0x081A [ 0x5678 ]
0x081C [ 0x9ABC ]    ⇒    0x081C [ 0x9ABC ]
```

e. `0x014020  RETURN`

```
PC [ 0x014020 ] ⇒    PC [ 0x01220E ]
W15  [ 0x0808 ] ⇒    W15  [ 0x0804 ]

Data Memory
0x0804 [ 0x220E ]    ⇒    0x0804 [ 0x220E ]
0x0806 [ 0x0001 ]    ⇒    0x0806 [ 0x0001 ]
0x0808 [ 0x0002 ]    ⇒    0x0808 [ 0x0002 ]
```

f. `0x016022  POP  W3`
`0x016024  RETURN`

```
PC [ 0x016022 ] ⇒    PC [ 0x012030 ]
W3   [ 0x0816 ] ⇒    W3   [ 0x0001 ]
W15  [ 0x081C ] ⇒    W15  [ 0x0816 ]

Data Memory
0x0814 [ 0x00AB ]    ⇒    0x0814 [ 0x00AB ]
0x0816 [ 0x2030 ]    ⇒    0x0816 [ 0x2030 ]
0x0818 [ 0x0002 ]    ⇒    0x0818 [ 0x0002 ]
0x081A [ 0x0001 ]    ⇒    0x081A [ 0x0001 ]
```