1. (5 pts each) Complete the following table by writing in values (or updating values) for variables changed by the expression in the “Operation” column. Assume all variables have datatype uint8_t. Perform each operation independent from the others; that is do not carry the results of one operation into the next row of the table. If a value is updated, then strikethrough the value and write-in a new value. If an empty cell does not change value, leave the cell empty.

<table>
<thead>
<tr>
<th>Operation</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>( z = z \times y - x; )</td>
<td>80</td>
<td>30</td>
<td>440</td>
</tr>
<tr>
<td>( z += x; )</td>
<td>100</td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>( z = x \times y )</td>
<td>25</td>
<td>13</td>
<td>325-256=69</td>
</tr>
</tbody>
</table>

**Values and answers in decimal**

<table>
<thead>
<tr>
<th>Operation</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>( z = x \gg 2; )</td>
<td>0b1001 1010</td>
<td></td>
<td>0b0001 0110</td>
</tr>
<tr>
<td>( z = x &gt; y; )</td>
<td>0b0110 1010</td>
<td>0b1001 1001</td>
<td>0b0000 0000</td>
</tr>
</tbody>
</table>

**Values and answers in binary (space put between every four bits for readability)**

2. (10 pts) Evaluate the following program and determine the value of sum when the for-loop exit.

```c
uint8_t i; in    sum
uint8_t sum = 0; 5     5
for(i=5; i<12; i+=2) {
    sum += 1;
}
```

3. (5 pts each) Given the following C-code, find the value of \( a \) after executing the code with the value of \( x, y, z \) given in each row of the table.

```c
if ((x < 30) && (y >= 20) && (z > 40)) {
    a = 0;
} else if (((x > 25) && ((y != 30) || (z == 20))) {
    a = 1;
} else if (((x >= 15) || (y == 15)) {
    a = 2;
} else {
    a = 3;
} // end if
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>10</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>30</td>
<td>20</td>
<td>2</td>
</tr>
</tbody>
</table>

4. (10 pts) Write a single line of C-code that waits (does nothing) while an active-high push button on RA2 is pressed. Use the correct register/field syntax.

```c
while(PORTAbits.RA2 == 1);
```

5. (10 pts) Write a single line of C-code that configure pin RC2 as an output.

```c
TRISCbits.TRISC2 = 0;
```
6. (10 pts) Complete the following code-snippet that moves all the values of array to an index one greater - for example, the value in array[5] is moved into array[6]. Moves 0 into the value at array index 0.

```c
uint8_t i=0;
uint8_t array[10];
for (i=9; i>0; i--) { // Must count down
    array[i] = array[i-1];
} // end for
array[0] = 0;
```

7. (10 pts) Write a valid line of C-code that turns-off the LED at right using register and bit-field naming convention for our PIC.

```c
LATCbits.LATC2 = 1;
```

Use the following C-code snippet to answer questions 8-10.

1. TOCONbits.PSA = 0;
2. TOCONbits.T0PS = <question #9>;
3. TOCONbits.TMR0ON = 1;
4. TMR0 = 0x10000 - delayValue;
5. while(INTCONbits.TMR0IF == 0);

8. (10 pts) What prescaler value (in binary) for line #2 will create the most accurate 100 ms delay on line 5 when run on the PIC as normally configured? State your answer in binary.

A 1:32 prescaler is the smallest prescaler with a maximum period over 100 ms. It has a maximum period of 131 ms. The binary code is 0b100;


<table>
<thead>
<tr>
<th>1 second</th>
<th>1000 ms</th>
<th>16 clk</th>
</tr>
</thead>
<tbody>
<tr>
<td>--------- *</td>
<td>--------- *</td>
<td>x counts = 25 ms</td>
</tr>
<tr>
<td>16*10^6 clks</td>
<td>1 second</td>
<td>1 count</td>
</tr>
</tbody>
</table>

OR

<table>
<thead>
<tr>
<th>16*10^6 clks</th>
<th>1 second</th>
<th>1 count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 second 1000 ms</td>
<td>16 clk</td>
<td></td>
</tr>
<tr>
<td>25 ms = 25,000 counts</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10. (10 pts) What delay, in milliseconds, is generated in line 5 with TOCONbits.T0PS = 0b101; and delayValue=20,000? State your answer using dimensional analysis.

<table>
<thead>
<tr>
<th>1 second</th>
<th>1000 ms</th>
<th>64 clk</th>
</tr>
</thead>
<tbody>
<tr>
<td>--------- *</td>
<td>--------- *</td>
<td>20,000 counts = 80 ms</td>
</tr>
<tr>
<td>16*10^6 clks</td>
<td>1 second</td>
<td>1 count</td>
</tr>
</tbody>
</table>
11. (10 pts) Timer 1, configured with a 1:8 prescaler, starts at 50,000 and counts up to 10,000. How much time, in ms, has gone by? State your answer using dimensional analysis.

\[
\begin{align*}
2^{16} - 50,000 &= 15,536 + 10,000 = 25,536 \\
&\text{counts with a 1:8 prescaler} \\
&\text{1 second} \quad 1000 \text{ ms} \quad 8 \text{ clk} \\
&\frac{16 \times 10^6 \text{ clks}}{1 \text{ second}} \quad 1 \text{ count} \\
&\frac{16 \times 10^6 \text{ clks}}{1 \text{ second}} \quad 1 \text{ count} \\
= 12.768 \text{ms}
\end{align*}
\]

A sensor placed near a rotating gear nominally outputs logic 0, but outputs a brief logic 1 pulse every time the magnet embedded into the gear passes by. Your job is to write a program which calculates the number of rotations per second for the gear which rotates between 25 and 250 rotations per second. Your PIC is attached to the sensor through RC2. You will use a CCP module to solve this problem.

12. (10 pts) What prescaler should timer 1 run at?

- 25 rotations per second = 40ms
- 250 rotations per second = 4ms

So we will need to use a 1:16 prescaler because it can measure up to 65.5ms

13. (40 pts) Write a code snippet to measure the number of timer counts required for the rotating gear to make a single rotation using a CCP module. Assume that the timer and CCP subsystems are initialized for you and associated with the CCP module. Declare any additional variables at the top of your program.

```c
uint16_t rotationTime;
uint16_t prevCnt;

CCP1CONbits.CCP1M = 0b0101; // Choose either edge

for (;;) {
    PIR1bits.CCP1IF = 0; // Wait for the edge
    while(PIR1bits.CCP1IF == 0);
    rotationTime = CCP1 - prevCnt; // record period
    prevCnt = CCP1; // setup for next iteration
} // end infinite loop
```
14. (5 pts) Converts the result from the previous question into rotations per second. State your answer using dimensional analysis.

\[
16 \times 10^6 \text{ clks} \quad \text{1 count} \quad \text{1 rotation} \quad \frac{1}{1 \text{ second}} \cdot \frac{16 \text{ clk}}{1 \text{ count}} = 1,000,000/x \text{ RPS}
\]