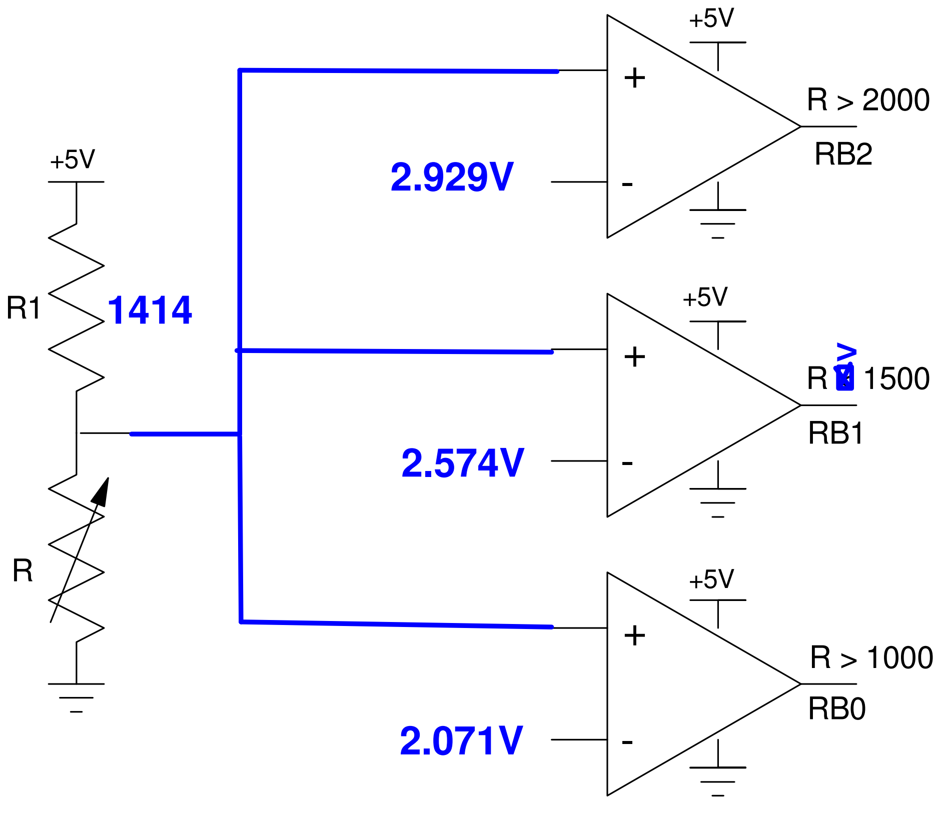
ECE 376 - Test #1: Name ____________________

1) **Digital Inputs.** Design a circuit which has three digital outputs (5V when true, 0V when false)

- R > 2000 Ohms (RB2)
- R > 1500 Ohms (RB1)
- R > 1000 Ohms (RB0)

Assume

- R1 = 900 + 100*(your birth month) + (your birth date).
- May 14th, for example, gives R1 = 1414 Ohms



2) Digital Outputs: Design a circuit which allows your PIC to drive a 10W LED at 500mA

Assume a 10W UV LED has the following characteristics

- Vf = 12V @ 800mA
- 1,000 Lumens @ 800mA

Assume a 6144 NPN transistor

- Vbe = 700mV
- Vce(sat) = 360mV
- Current gain = β = 200

Determine the light output, Rb, and Rc

Lumens	Ic (mA)	Rb	Rc
625	**500 mA**	860 Ohms 172 < Rb < 1720	15.28 Ohms

$$R_c = \left(\frac{20V - 12V - 0.36V}{500mA}\right) = 15.28\Omega$$

To saturate

$$\beta I_b > I_c$$

$$I_b > \left(\frac{500mA}{200}\right) = 2.5mA$$

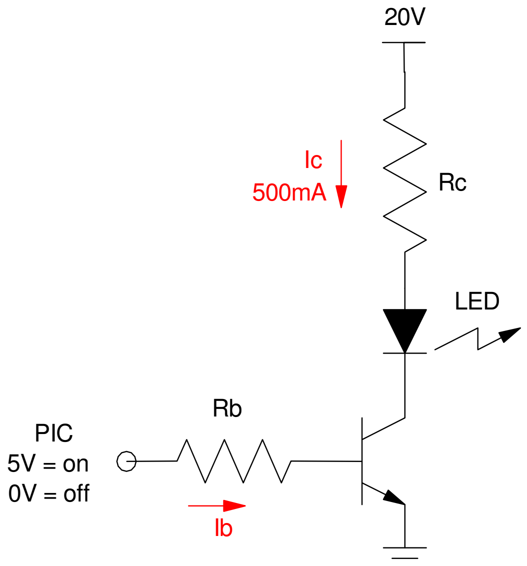
Let Ib = 5mA

$$R_b = \left(\frac{5V - 0.7V}{5mA}\right) = 860\Omega$$

Lumens

$$L = \left(\frac{500mA}{800mA}\right) 1000 \cdot \text{lumens}$$

$$L = 625$$



3) **Assembler:** Determine the contents of the W, PORTB, and PORTC registers after each operation. Assume

- PORTB and PORTC are output.
- Default is decimal

	W	PORTB	PORTC
Start:	Birth Month (1..12) 5	Birth Date (1..31) 14	7
addwf PORTB, F	5	**19**	7
subwf PORTC, W	**2**	19	7
btg PORTB, 2	2	**23**	7
incf PORTB, W	**24**	23	7
andwf PORTB, F	24	**16**	7
iorwf PORTC, W	**31**	16	7
negf PORTB, F	31	**-16 = 240**	7
comf PORTC, W	**-8 = 248**	-16	7
movf PORTB, W	**-16**	-16	7
movwf PORTC	-16	-16	**-16**

4) Assembler & Timing:

a) Determine the number of clocks the following assembler subroutine takes to execute.

- Assume MONTH and DAY be your birth month and day.

b) Modify this routine (change A, B, and C) so that it takes 1,500,000 clocks (150ms seconds) to execute

- +/- 50,000 clocks

A birth month 1..12	B birth day: 1..31	C	N number of clocks Wait takes
5	14	200	**154,542**
A	B	C	N
10	**54**	**252**	**1,500,757**

```
Wait:
    movlw    A                 N = 7
    movwf    CNT2
     nop
     nop
     nop
W2:
         movlw    B            N = 9 * 5
         movwf    CNT1
         nop
         nop
         nop
         nop

W1:
              movlw    C       N = 7 * 14 * 5
              movwf    CNT0
              nop
              nop
W0:
                   nop         N = 11 * 200 * 14 * 5
                   nop
                   nop
                   nop
                   nop
                   nop
                   nop
                   nop
                   decfsz CNT0,F
                   goto   W0

              decfsz  CNT1,F
              goto    W1

         decfsz    CNT2,F
         goto W2
    return
```

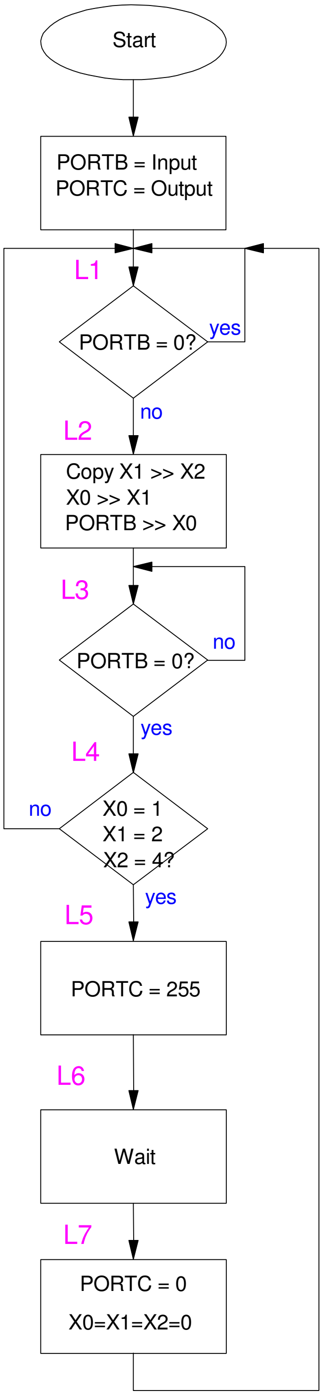
5) Assember & Flow Charts. Write an assembler program to turn your PIC processor into a combination lock

- Press and release PORTB pin 2 then 1 then 0
- If done in this order, the lights on PORTC turn on for 1 second
- Assume a wait routine (Wait:) exists which kills 10,000,000 clocks (one second)
- X0, X1, and X2 are 8-bit spots in memory

Bonus: (Due Monday 2pm): Program and demonstrate the combination lock on your PIC board

```
X0 equ 0
X1 equ 1
X2 equ 2
;  Start of program


     org       0x800
     movlw     0xFF
     movwf     TRISB
     clrf      TRISC
     movlw     0x0F
     movwf     ADCON1
L1:
     movlw     0
     cpfseq    PORTB
     goto      L2
     goto      L1
L2:
     movff     X1,X2
     movff     X0,X1
     movff     PORTB,X0
L3:
     movlw     0
     cpfseq    PORTB
     goto      L3
L4:
     movlw     1
     cpfseq    X0
     goto      L1
     movlw     2
     cpfseq    X1
     goto      L1
     movlw     4
     cpfseq    X2
     goto      L1
L5:
     movlw     0xFF
     movwf     PORTC
L6:
     call      Wait
L7:
     clrf      PORTC
     clrf      X0
     clrf      X1
     clrf      X2
     goto      L1
```



Memory Read & Write		
MOVWF PORTA	memory write	w → PORTA
MOVFF PORTA PORTB	copy	PORTA → PORTB
MOVF PORTA,W	memory read	PORTA → W
MOVLW 234	Move Literal to WREG	123 → W
Memory Clear, Negation		
CLRF PORTA	clear memory	0x00 → PORTA
COMF PORTA, W	toggle bits	!PORTA → W (bit toggle)
NEGF PORTA, W	negate	-PORTA → W (2's compliment)
Addition & Subtraction		
INCF PORTA,F	increment	PORTA + 1 → PORTA
ADDWF PORTA, F	add	PORTA + W → PORTA
ADDWFC PORTA, W	add with carry	PORTA + W + carry → W
ADDLW	Add Literal and WREG	
DECF PORTA,F	decrement	PORTA -1 → PORTA
SUBFWB PORTA,F	subtract with borrow	PORTA - W - c → PORTA
SUBWF PORTA,F	subtract no borrow	PORTA - W → PORTA
SUBWFB PORTA,F	subtract with borrow	PORTA - W - c → PORTA
SUBLW 223	Subtract WREG from #	223 - W → W
Shift left (*2), shift right (/2)		
RLCF PORTA,F	rotate left through carry (9-bit rotate)	
RLNCF PORTA,F	rotate left no carry	
RRCF PORTA,F	rotate right through carry	
RRNCF PORTA,F	rotate right no carry	
Bit Operations		
BCF PORTA, 3	Bit Clear f	clear bit 3 of PORTA
BSF PORTA, 4	Bit Set f	set bit 4 of PORTA
BTG PORTA, 2	Bit Toggle f	toggle bit 2 of PORTA
Logical Operations		
ANDWF PORTA, F	logical and	PORTA = PORTA and W
ANDLW 0x23	AND Literal with WREG	W = W and 0x23
IORWF PORTA,F	logical or	PORTA = PORTA or W
IORLW 0x23	Inclusive OR Literal	W = W or 0x23
XORWF PORTA,F	logical exclusive or	PORTA = PORTA xor W
XORLW 0x23	Exclusive OR Literal	W = W xor 0x23
Tests (skip the next instruction if...)		
CPFSEQ PORTA	Compare PORTA to W, skip if PORTA = W	
CPFSGT PORTA	Compare PORTA to W, Skip if PORTA > W	
CPFSLT PORTA	Compare PORTA to W, Skip if PORTA < W	
DECFSZ PORTA,F	decrement, skip if zero	
DCFSNZ PORTA,F	decrement, skip if not zero	
INCFSZ PORTA,F	increment, skip if zero	
INFSNZ PORTA,F	increment, skip if not zero	
BTFSC PORTA, 5	Bit Test f, Skip if Clear	
BTFSS PORTA, 1	Bit Test f, Skip if Set	
Flow Control		
GOTO Label	Go to Address 1st word	
CALL Label	Call Subroutine 1st word	
RETURN	Return from Subroutine	
RETLW 0x23	Return with 0x23 in WREG	