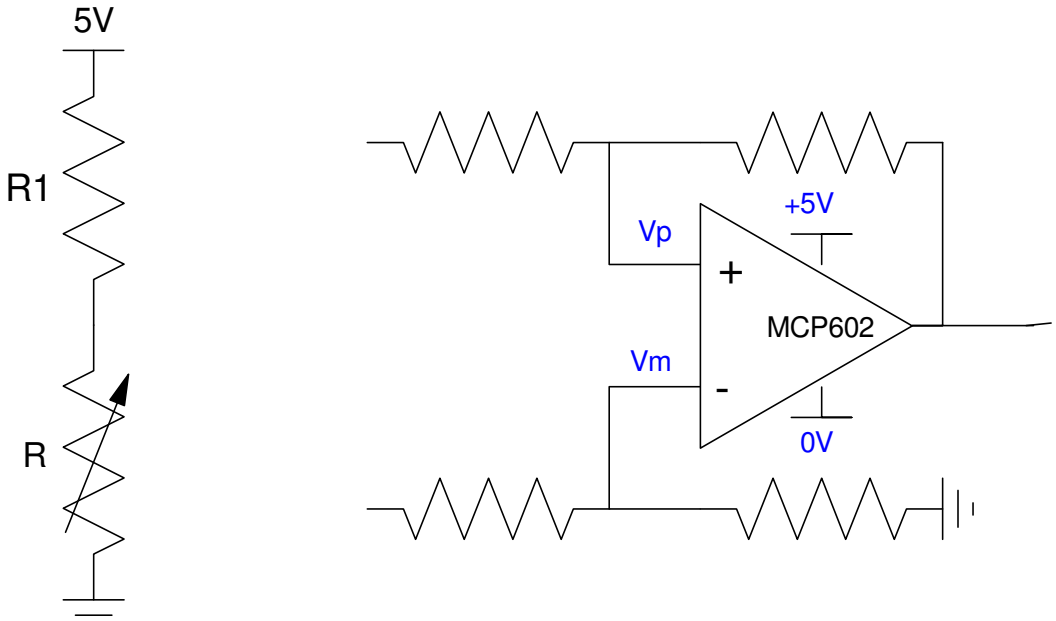
ECE 376 - Test #1: Name _____

1) **Digital Inputs.** Design a circuit which outputs

- 0V when R > 1750 Ohms
- 5V when R < 1350 Ohms

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 turn on and off a 50W LED

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

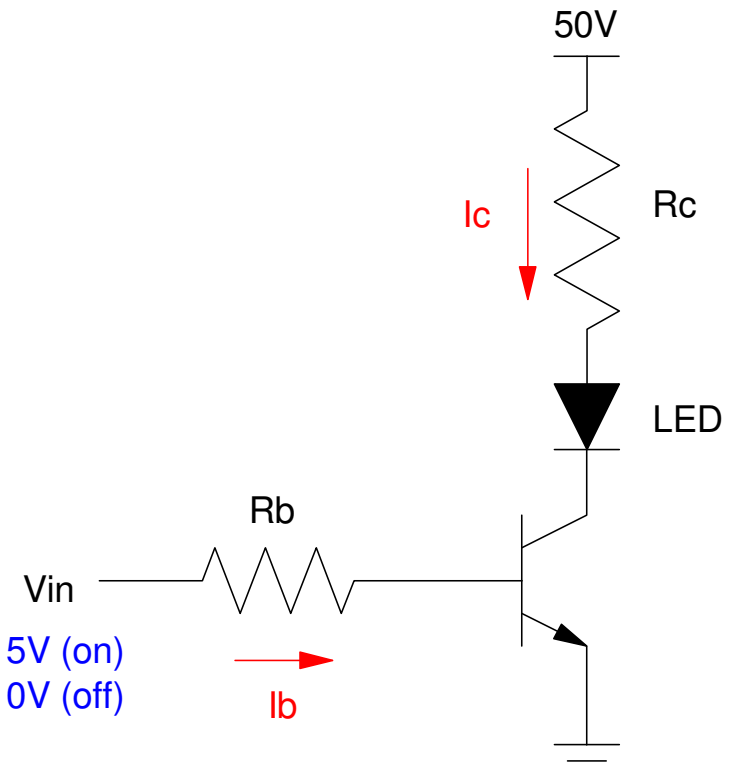
Assume a 50W LED has the following characteristics

- Vf = 25V @ 2A
- 5000 Lumens @ 2A

Assume a 6144 NPN transistor

- Vbe = 700mV
- Vce(sat) = 200mV
- Current gain = β = 300

Lumens	Ic (mA)	Rb	Rc 900 + 100*Month + Day



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)	Birth Date (1..31)	15
addlw 2			
subwf PORTC,F			
addwf PORTB,F			
movf PORTB,W			
iorlw 0x15			
movwf PORTB			
btg PORTC,0			
iorwf 0x0F			
negf PORTB,F			
comf PORTC,F			

4) Assembler & Timing:

a) Convert the following C code to assembler.

- Assume Aand B are 8-bit numbers

b) How long does your program take to execute when A = 25?

Clocks

```
unsigned char A, B;

if(A < 10)
     B = 0
elseif(A < 20)
     B = 1
elseif(A == 25)
     B = 2
else
     B = 3
```

5) Assember & Flow Charts. Write an assembler program that corresponds to the following flow chart. This program gives you points based upon which button you press:

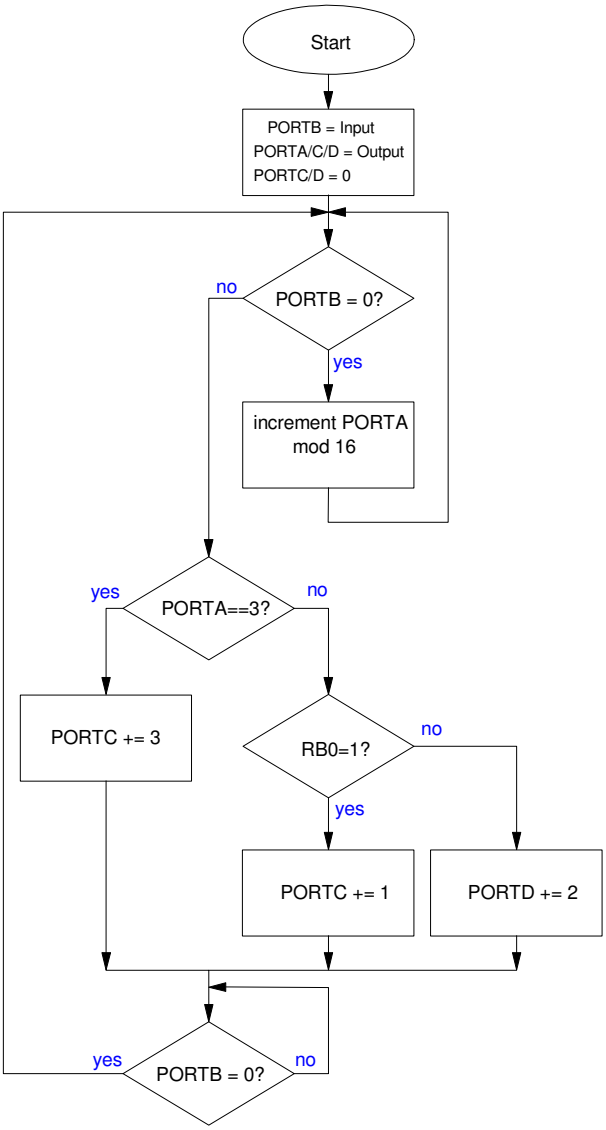
- Button RB0: Usually one point, sometimes 3 points
- Button RB1: Always 2 points

Test #1: (due Friday) Write the assembler code

Bonus (due Monday): Demonstrate your program on your PIC board

```
#include <p18f4620.inc>

; Start of Program
     org  0x800
```



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	