ECE 376 - Homework #8

Timer2 & INT Interrupts - Due Monday, Marth 25th

Stoplight with 1ms Accuracy

1) Revise your previous code for a PIC controlled stoplight to use Timer2 interrupts to set the timing:
   - Set up Timer2 interrupts for every 1ms
   - Each interrupt, toggle pin RC0 (outputting a 500Hz square wave)
   - On the LCD display, display the running time with a resolution of 1ms

Assume PORTC displays the E/W and N/S lights:

7	6	5	4	3	2	1	0
-	R	Y	G	-	R	Y	G
E/W				N/S			

The stoplight cycles every 14 seconds

Duration	E/W	N/S	PORTC
5s	G	R	0x14
2s	Y	R	0x24
5s	R	G	0x41
2s	R	Y	0x42

Code

```
// Global Variables

const unsigned char MSG0[21] = "E/W                 ";
const unsigned char MSG1[21] = "N/S                 ";
const unsigned char Gtxt[9] = "Green   ";
const unsigned char Ytxt[9] = "Yellow  ";
const unsigned char Rtxt[9] = "Red     ";

unsigned long int TIME;

// Subroutine Declarations
#include <pic18.h>

// Subroutines
#include        "lcd_portd.c"

// High-priority service
void interrupt IntServe(void)
{
   if (TMR2IF) {
      RA1 = !RA1;
      if(TIME) TIME -= 1;
      TMR2IF = 0;
      }
   }
```

```
// Main Routine
void main(void)
{
   unsigned char i, j;
   TRISA = 0;
   TRISB = 0;
   TRISC = 0;
   TRISD = 0;
   TRISE = 0;
   ADCON1 = 0x0F;

   LCD_Init();                    // initialize the LCD
   LCD_Move(0,0);  for (i=0; i<20; i++) LCD_Write(MSG0[i]);
   LCD_Move(1,0);  for (i=0; i<20; i++) LCD_Write(MSG1[i]);
   Wait_ms(100);

// set up Timer2 for 1ms
   T2CON = 0x4D;
   PR2 = 249;
   TMR2ON = 1;
   TMR2IE = 1;
   TMR2IP = 1;
   PEIE = 1;

// turn on all interrupts
   GIE = 1;

   while(1) {
// green / red for 5 seconds
      PORTC = 0x14;
      TIME = 5000;
      LCD_Move(0,  8);
      for (i=0; i<8; i++) LCD_Write(Gtxt[i]);
      LCD_Move(1,  8);
      for (i=0; i<8; i++) LCD_Write(Rtxt[i]);
      while(TIME);
// yellow / red for 2 seconds
      PORTC = 0x24;
      TIME = 2000;
      LCD_Move(0,  8);
      for (i=0; i<8; i++) LCD_Write(Ytxt[i]);
      LCD_Move(1,  8);
      for (i=0; i<8; i++) LCD_Write(Rtxt[i]);
      while(TIME);
// red / green for 5 seconds
      PORTC = 0x41;
      TIME = 5000;
      LCD_Move(0,  8);
      for (i=0; i<8; i++) LCD_Write(Rtxt[i]);
      LCD_Move(1,  8);
      for (i=0; i<8; i++) LCD_Write(Gtxt[i]);
      while(TIME);
// red / yellow for 2 seconds
      PORTC = 0x42;
      TIME = 2000;
      LCD_Move(0,  8);
      for (i=0; i<8; i++) LCD_Write(Gtxt[i]);
      LCD_Move(1,  8);
      for (i=0; i<8; i++) LCD_Write(Rtxt[i]);
      while(TIME);
// repeat
      }
   }
```

Version 2 (short version):

- You can use interrupts as servents of the main routine (as in the previous example), or
- You can have the interrupt do all of the work (this example)

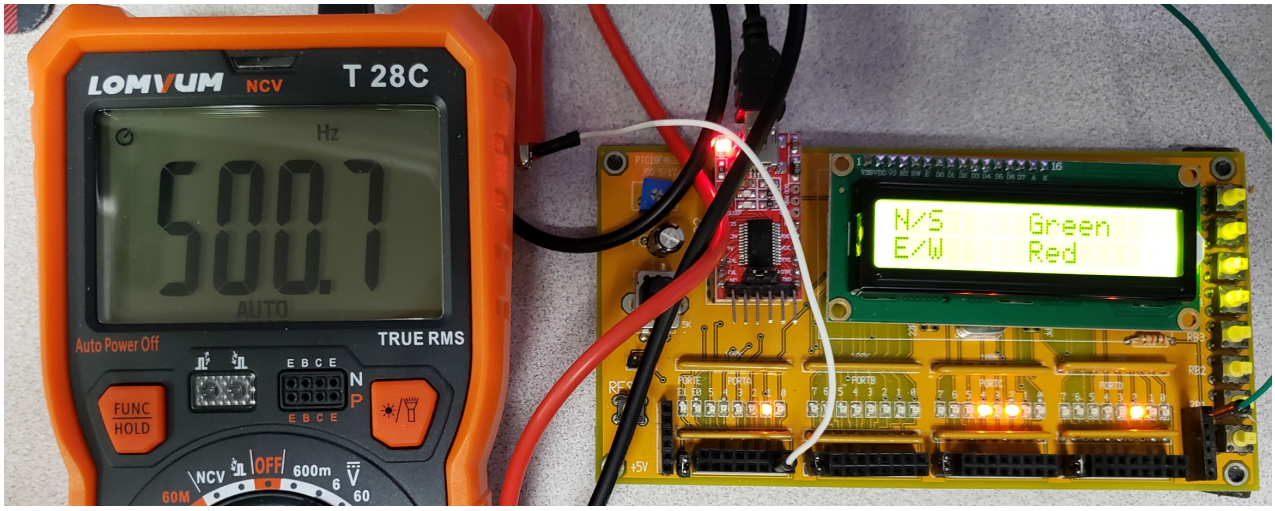
```
// Global Variables

const unsigned char MSG0[21] = "Stoplight           ";

unsigned long int TIME;

// Subroutine Declarations
#include <pic18.h>

// Subroutines
#include        "lcd_portd.c"

// High-priority service
void interrupt IntServe(void)
{
   if (TMR2IF) {
      RA1 = !RA1;
      TIME += 1;
      if(TIME > 14000) TIME = 0;
      if(TIME < 5000) PORTC = 0x14;
      elseif(TIME < 7000) PORTC = 0x24;
      elseif(TIME < 12000) PORTC = 0x41;
      else PORTC = 0x42;
      TMR2IF = 0;
      }
   }
// Main Routine
void main(void)
{
   unsigned char i, j;
   TRISC = 0;
   ADCON1 = 0x0F;

   LCD_Init();                  // initialize the LCD
   LCD_Move(0,0);  for (i=0; i<20; i++) LCD_Write(MSG0[i]);
   Wait_ms(100);

// set up Timer2 for 1ms
   T2CON = 0x4D;
   PR2 = 249;
   TMR2ON = 1;
   TMR2IE = 1;
   TMR2IP = 1;
   PEIE = 1;

// turn on all interrupts
   GIE = 1;
   TIME = 0;

   while(1) {
      LCD_Move(1,0);  LCD_Out(TIME, 5, 3);
      }
   }
```

2) Validate your program works

The frequency on RA1 = 500.7Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 9986.02$$

Timing is 5 seconds green / 2 seconds yellow / 7 seconds red



Generating Frequencies with Timer2 Interrupts

3) Write a routine which turns plays your PIC into a 1-string banjo using Timer2 interrupts

- Play note frequency of music note E4 (329.63Hz) on pin RC0 when button RB0 is pressed
- Check the accuracy of your music note using your cell phone (or whatever else you have on hand)

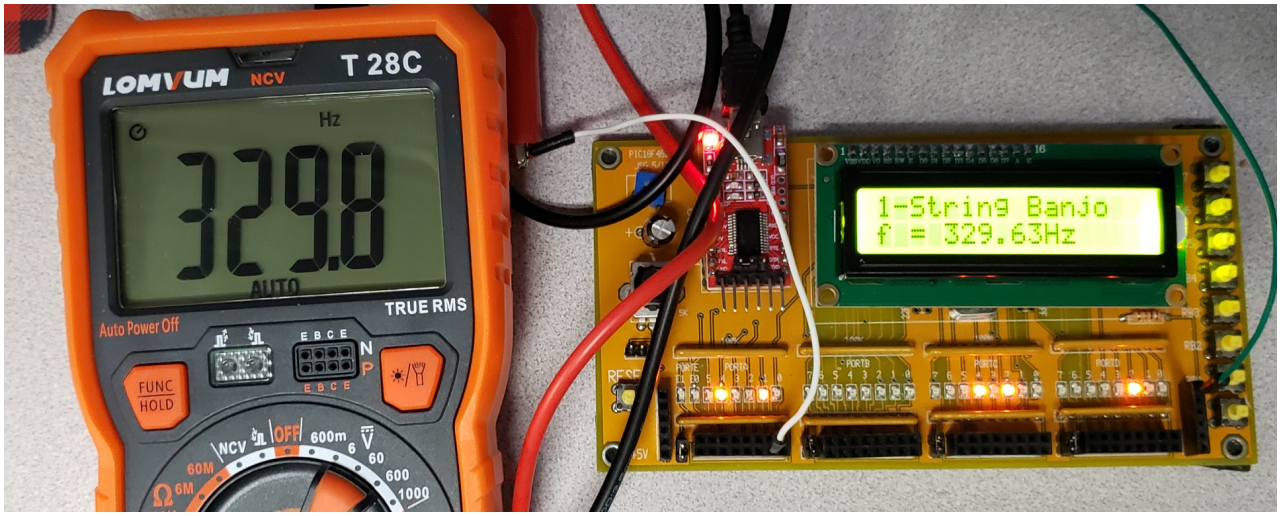
$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 15,168.52$$

Come up with (A, B, C) so that A*B*C is close. One combination is close is

- A = 15, B = 253, C = 4

This results in T2CON = 75

T2CON = 0x75							
7	6	5	4	3	2	1	0
0	1	1	1	0	1	0	1
	A = 15					C = 4	



Code:

```
// Stoplight Controller

// Global Variables

const unsigned char MSG0[21] = "1-String Banjo      ";
const unsigned char MSG1[21] = "f = 329.63Hz        ";

// Subroutine Declarations
#include <pic18.h>

// Subroutines
#include        "lcd_portd.c"

// High-priority service
void interrupt IntServe(void)
{
   if (TMR2IF) {
      if(RB0) RC0 = !RC0;
      RA1 = !RA1;
      TMR2IF = 0;
      }
   }

// Main Routine

void main(void)
{
   unsigned char i;

   TRISA = 0;
   TRISB = 0xFF;
   TRISC = 0;
   TRISD = 0;
   TRISE = 0;
   ADCON1 = 0x0F;

   LCD_Init();                  // initialize the LCD
   LCD_Move(0,0);  for (i=0; i<20; i++) LCD_Write(MSG0[i]);
   LCD_Move(1,0);  for (i=0; i<20; i++) LCD_Write(MSG1[i]);
   Wait_ms(100);


// set up Timer2 for 329.63Hz
   T2CON = 0x75;
   PR2 = 252;
   TMR2ON = 1;
   TMR2IE = 1;
   TMR2IP = 1;
   PEIE = 1;

// turn on all interrupts
   GIE = 1;

   while(1) {

         }
   }
```

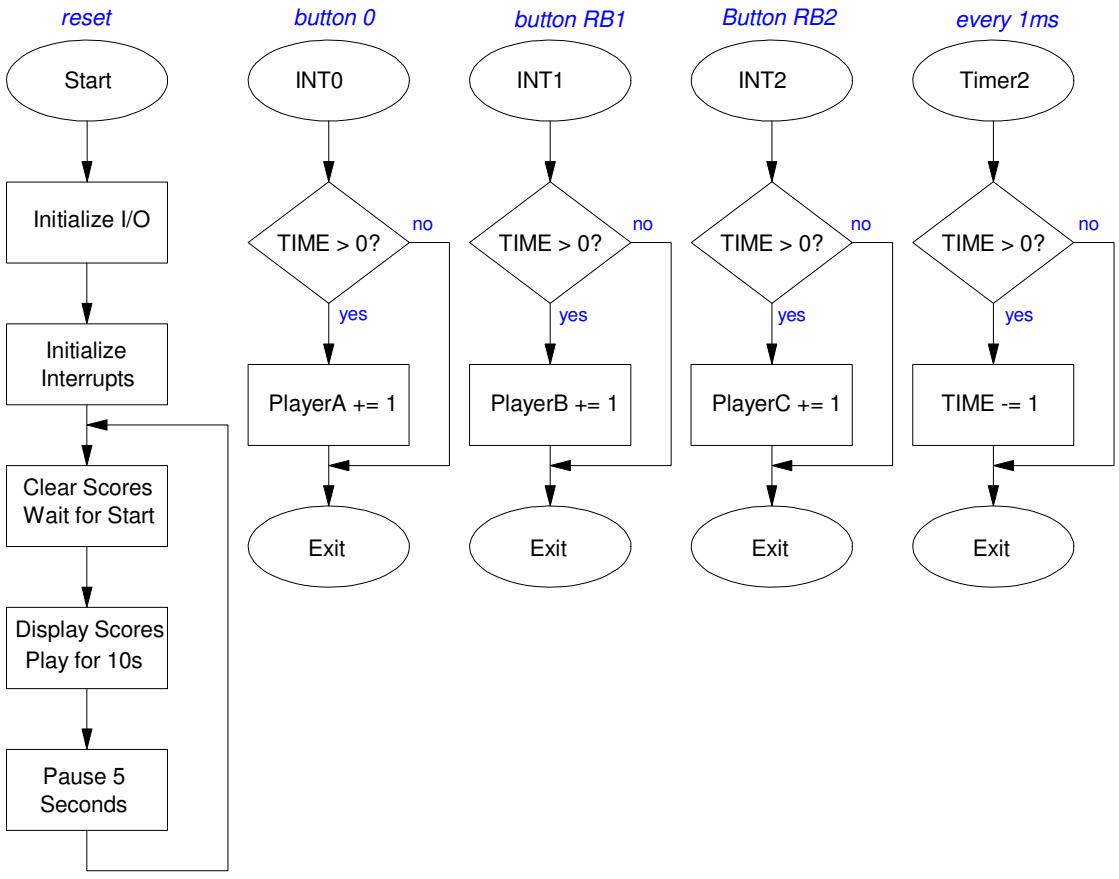
Hungry-Hungry Hippo!

Problem 4-9) Write a program which uses INT and Timer2 interrupts to play a game of Hungry-Hungry Hippo

- The game has three players: A, B , and C
  - Player A presses RB0
  - Player B presses RB1
  - Player C presses RB2
- The game starts when someone presses their button. Once pressed
  - All player scores are reset to zero and
  - A 10 second timer starts (controlled with Timer2)
- When the game is on, INT interrupts count how many times each player presses their button
  - Rising edge interrupts
- Once the game is over (10 seconds runs out), you quit counting button presses.
- As the game is running, display
  - The scores for player A, B, and C, and
  - The time remaining in the game with a resolution of 1ms

4) Write a flow-chart for this program

- *note: you should have five flow charts: one for the main routine, one for each interrupt*



5) Write the corresponding C code

```
// Stoplight Controller

// Global Variables

const unsigned char MSG0[21] = "Time =              ";
const unsigned char MSG1[21] = "Press Any Button    ";
const unsigned char MSG2[21] = "                    ";
unsigned int TIME, PlayerA, PlayerB, PlayerC;

// Subroutine Declarations
#include <pic18.h>

// Subroutines
#include        "lcd_portd.c"

// High-priority service
void interrupt IntServe(void)
{
   if (TMR2IF) {
      if(TIME) TIME -= 1;
      RA1 = !RA1;
      TMR2IF = 0;
      }
   if(INT0IF){
      if(TIME) PlayerA += 1;
      INT0IF = 0;
      }
   if(INT1IF){
      if(TIME) PlayerB += 1;
      INT1IF = 0;
      }
   if(INT2IF){
      if(TIME) PlayerC += 1;
      INT2IF = 0;
      }
   }

// Main Routine

void main(void)
{
   unsigned char i;

   TRISA = 0;
   TRISB = 0xFF;
   TRISC = 0;
   TRISD = 0;
   TRISE = 0;
   ADCON1 = 0x0F;

   LCD_Init();                  // initialize the LCD
   LCD_Move(0,0);  for (i=0; i<20; i++) LCD_Write(MSG0[i]);
   LCD_Move(1,0);  for (i=0; i<20; i++) LCD_Write(MSG1[i]);
   Wait_ms(100);
```

```
// set up Timer2 for 1ms
   T2CON = 0x4D;
   PR2 = 249;
   TMR2ON = 1;
   TMR2IE = 1;
   TMR2IP = 1;
   PEIE = 1;

// Turn on INT0
   INT0IE = 1;
   INTEDG0 = 1;
// Turn on INT1
   INT1IE = 1;
   INTEDG1 = 1;
// Turn on INT2
   INT2IE = 1;
   INTEDG2 = 1;
// turn on all interrupts
   GIE = 1;

   while(1) {
       PlayerA = 0;
       PlayerB = 0;
       PlayerC = 0;
// press a button to start
       while(PORTB == 0);
       LCD_Move(1,0);  for (i=0; i<20; i++) LCD_Write(MSG2[i]);
       TIME = 10000;
       while(TIME) {
          LCD_Move(1,0);   LCD_Out(PlayerA, 3, 0);
          LCD_Move(1,5);   LCD_Out(PlayerB, 3, 0);
          LCD_Move(1,10);  LCD_Out(PlayerC, 3, 0);
          LCD_Move(0,8);   LCD_Out(TIME, 5, 3 );
          }
// game over  Wait 5 seconds then start a new game on a button press
      Wait_ms(5000);
      LCD_Move(1,0);  for (i=0; i<20; i++) LCD_Write(MSG1[i]);
      }
   }
```

6) Validate your code

Pressing a button starts the game (scores are reset to zero, time is set to 10.000 seconds0

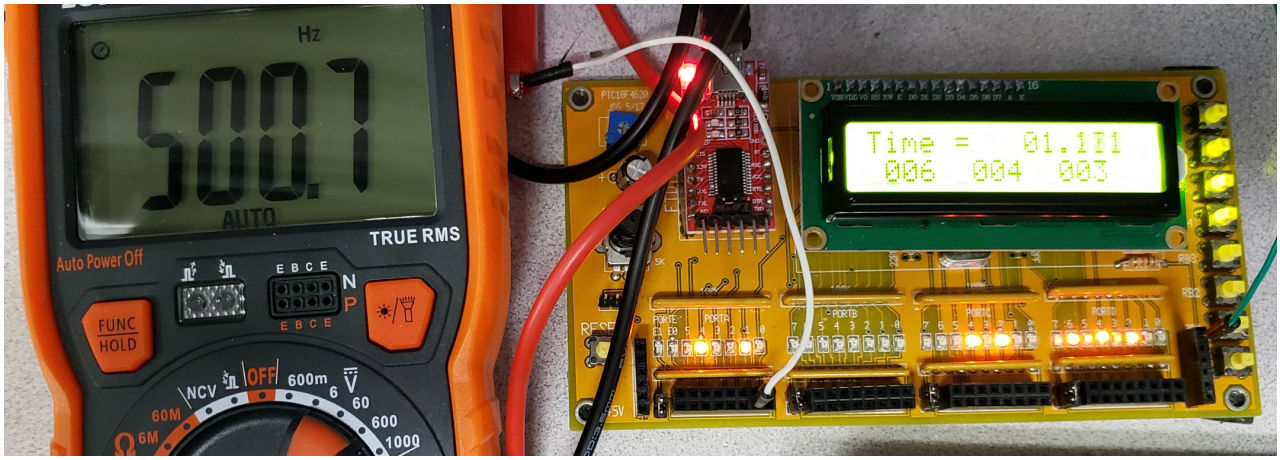
- check - pressing any button starts the game

When the game is on, pressing each player's button scores a point

- check: RB0 increases A's score
- RB1 increases B's score
- RB2 increases C's score
- Holding the button down doesn't change your score - only counts edges

Timer2 is running at 1ms (500Hz is output on a pin if you toggle it inside the Timer2 interrupts)

- Check: 500.7Hz appears on RA1



Fun with Hungry-Hungry Hippo:

7) Determine the 90% confidence interval for how many points you score when playing the game

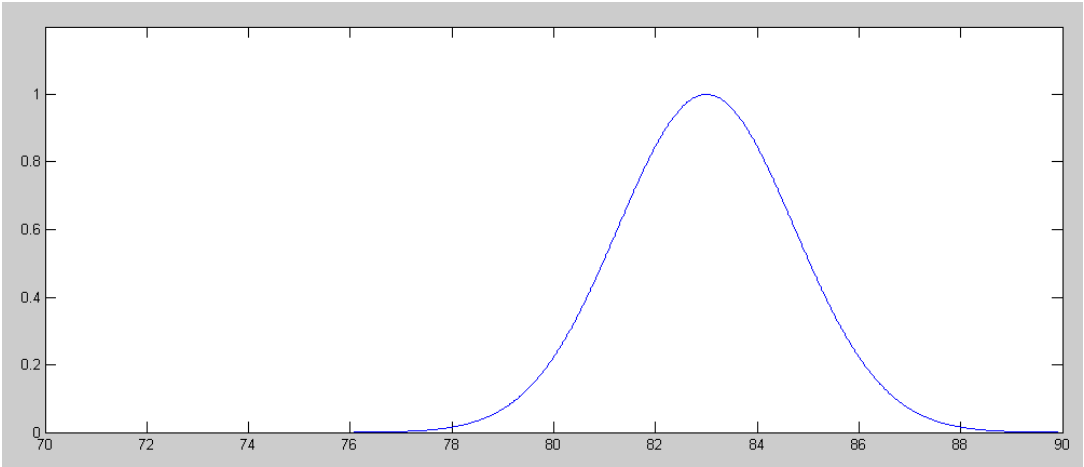
- Play two or more games (population A)
- Find the mean and standard deviation of your score
- Determine the 90% confidence interval using a student-t test.

Scores: {84, 81, 84}

In Matlab:

```
>> A = [84,81,84]

A =    84    81    84

>> Xa = mean(A)

Xa =    83

>> Sa = std(A)

Sa =    1.7321

>> Xa - 2.92*Sa

ans =   77.9424

>> Xa + 2.92*Sa

ans =   88.0576
```

Based upon this data, my score should be in the range of {77.9. 88.0} with a probability of 0.9



pdf for the score with my dominant hand

8) Collect a second set of data (use your off-hand, have someone else play the game, etc.)

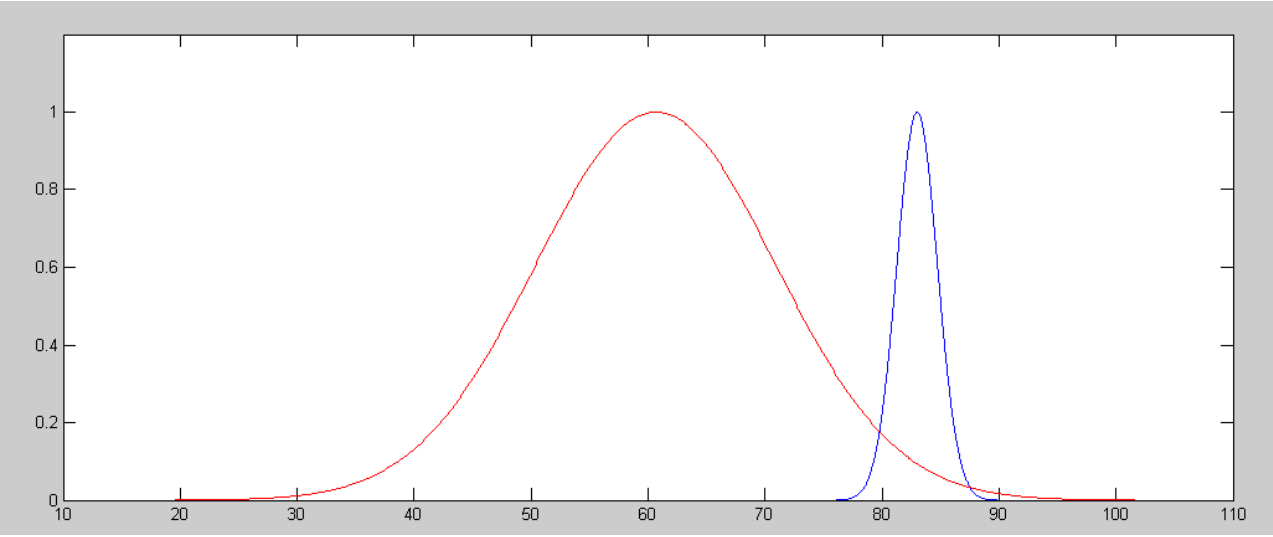
- Determine the 90% confidence interval for this data set (population B)

Left Hand: {58, 72, 52}

In Matlab

```
>> B = [58, 72, 52]

B =    58    72    52

>> Xb = mean(B)

Xb =   60.6667

>> Sb = std(B)

Sb =   10.2632

>> Xb - 2.92*Sb

ans =   30.6981

>> Xb + 2.92*Sb

ans =   90.6352
```

From the data, I should score with my left hand between {30.69, 90.63} points with a probability of 0.9



pdf for my score with my dominant hand (blue) and non-dominant hand (red)

9) Determine probability that A will beat B

- The next time you play (individual)
- Over a 1000 game match (population)

In Matlab, for one more game (individual)

```
>> Xw = Xa - Xb
Xw =    22.3333

>> Sw = sqrt(Sa^2 + Sb^2)
Sw =    10.4083

>> t = Xw / Sw
t =     2.1457
```

From StatTrek, with 2 degrees of freedom, this corresponds to a probability of 0.917

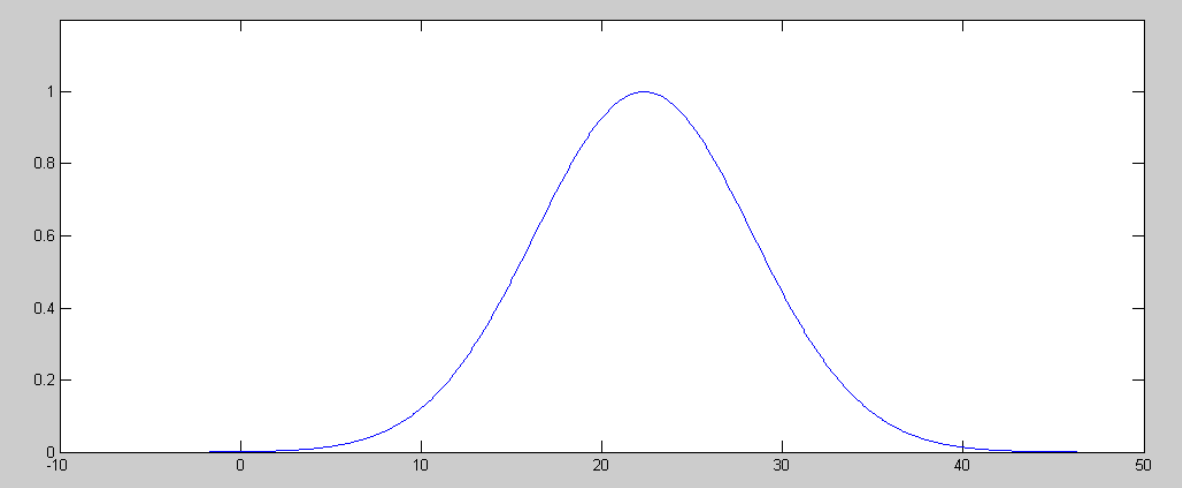
My dominant hand should win 91.7% of the time

For an infinite game series (population)

```
>> Xw = Xa - Xb
Xw =    22.3333

>> Sw = sqrt( (Sa^2)/3 + (Sb^2)/3 )
Sw =     6.0093

>> t = Xw / Sw
t =     3.7165
```

From StatTrek, this corresponds to a probability of 0.967

I am 96.7% certain that I should play *hungry-hungry-hippo* using my dominant hand



pdf for W = A - B