```
float sp = 625 ;      //degrees to turn

float kp = 90 ;       // P constant
float ki = 0.1 ;      // I Constant
float kd = 60;        // D constant

float Km = 0.0291 ;   // Motor Constant
float M = 0.000 ;     // Mass of Weight in kg
float g = 9.81 ;      // Acceleration due to gravity (m/s)

float u = 0;          // PID control
float u_static = 0 ;  // Reference Motor Speed
float up = 0 ;        // Total Motor Speed

float e = 625;        // Error in position
float eint = 0;       // Integral of Error
float ed = 0;         // Derivative of Error

float y = 0;          // Actual position
float dy = 0;         // Change in position

float tkm1;           // Initial tick for t=0
float tk ;            // Run Tick 1
float t2;             // Run Tick 2
float dt ;            // Change in time in ticks

byte DataLogFileX;    // Data Log Variables
string xdata;         // Data Container X
string tdata;         // Data Container T
string DatapairX;     //Data pair Strings
short bytesWritten;   //Number of bytes Written to file


task main()           // Main Function
{

// Delete Old Logs
DeleteFile( "DataLog-x.txt" );

// Create Log Files
if(CreateFile("DataLog-x.txt", 4096, DataLogFileX) == NO_ERR)
{

dt = 1;   //ms  - you may want to make this smaller, but you can't go less than one millisecond

// Start Time
tkm1 = CurrentTick();

// Engine Speed for Weight Cancellation
u_static = (M * g) / Km ;

// Loop until no Error
do
{
// Get Time
tk = CurrentTick() ;

// Find change in Angle
dy = MotorRotationCount(OUT_A) - y;

// Calculate new Position
y = y + dy;

// Data Values
xdata = NumToStr(y);
tdata = NumToStr(tk - tkm1);
```

```
// Record Data for Position
DatapairX = StrCat(tdata, ", " , xdata );
WriteLnString(DataLogFileX, DatapairX, bytesWritten);

// Calculate Error in Position
e = sp - y;

// Calculate Integral of Error (Newton's Method)
eint = eint + e * (dt/1000);

// Calculate Derivative of Error
ed = - dy/(dt/1000);

// Varying Control Signal
u = kp*e + ki*eint + kd*ed ;

// Total Control Signal
up = u + u_static;

// Observe Maximum Speed
if (up<-100.0)
  up = -100.0;
else if (u>100.0)
  up = 100.0;
else
  up = up;

// Command Motor
OnFwdReg(OUT_A, up, OUT_REGMODE_IDLE);

// Display Position
NumOut(15, LCD_LINE1, y);

// Get Next Time
t2 = CurrentTick();

// Wait until dt has passed
Wait( dt - (t2 - tk) );  //ensures that exact same amount of time happens in each loop

// Rolling Stop
Coast(OUT_A);

}
while(e != 0);

}

// Close Datalogs
CloseFile(DataLogFileX);

}
```