```
// tempsensor_engine.c
// for NerdKits with ATmega168 will change to atmega328
// need to change file directories would like to make thisinto a function
#define F_CPU 14745600
// line 5
#include <stdio.h>
#include <math.h>
#include <avr/io.h>
#include <avr/interrupt.h>
// line 10
#include <avr/pgmspace.h>
#include <inttypes.h>
#include "../libnerdkits/delay.h"
#include "../libnerdkits/lcd.h"
// line 15
#include "../libnerdkits/uart.h"
// PIN DEFINITIONS:
//
// PC0 -- temperature sensor analog input
//line 20
void adc_init() {
  // set analog to digital converter
  // for external reference (5v), single ended input ADC0
  ADMUX = 0;
// line 25
  // set analog to digital converter
  // to be enabled, with a clock prescale of 1/128
  // so that the ADC clock runs at 115.2kHz.
  ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);
// line 30
  // fire a conversion just to get the ADC warmed up
  ADCSRA |= (1<<ADSC);
}

uint16_t adc_read() {
  // read from ADC, waiting for conversion to finish
  // (assumes someone else asked for a conversion.)
  // wait for it to be cleared
  while(ADCSRA & (1<<ADSC)) {
    // do nothing... just hold your breath.
  }
  // bit is cleared, so we have a result.

  // read from the ADCL/ADCH registers, and combine the result
  // Note: ADCL must be read first (datasheet pp. 259)
  uint16_t result = ADCL;
```

```
  uint16_t temp = ADCH;
  result = result + (temp<<8);

  // set ADSC bit to get the *next* conversion started
  ADCSRA |= (1<<ADSC);

  return result;
}

double sampleToFahrenheit(uint16_t sample) {
  // conversion ratio in DEGREES/STEP:
  // (5000 mV / 1024 steps) * (1 degree / 10mV)
  //    ^^^^^^^^^^^^              ^^^^^^^^^^
  //     from ADC          from LM34
  return sample * (5000.0 / 1024.0 / 10.0);
}

int main() {
  // start up the LCD
  lcd_init();
  FILE lcd_stream = FDEV_SETUP_STREAM(lcd_putchar, 0, _FDEV_SETUP_WRITE);
  lcd_home();

  // start up the Analog to Digital Converter
  adc_init();

  // start up the serial port
  uart_init();
  FILE uart_stream = FDEV_SETUP_STREAM(uart_putchar, uart_getchar,
_FDEV_SETUP_RW);
  stdin = stdout = &uart_stream;

  // holder variables for temperature data
  uint16_t last_sample = 0;
  double this_temp;
  double temp_avg;
  uint8_t i;

  while(1) {
    // take 100 samples and average them!
    temp_avg = 0.0;
    for(i=0; i<100; i++) {
      last_sample = adc_read();
      this_temp = sampleToFahrenheit(last_sample);

      // add this contribution to the average
```

```
      temp_avg = temp_avg + this_temp/100.0;
    }


    // write message to LCD
    lcd_home();
    lcd_write_string(PSTR("Engine Temp "));
    lcd_write_int16(last_sample);
    lcd_write_string(PSTR(" of 1024   "));
    lcd_line_two();
    fprintf_P(&lcd_stream, PSTR("Temperature: %.2f"), temp_avg);
    lcd_write_data(0xdf);
    lcd_write_string(PSTR("F      "));

    // write message to serial port
    printf_P(PSTR("%.2f degrees F\r\n"), temp_avg);
  }


  return 0;
}
```