```
/*--------------------------------------------------------------------------------
 *  main.cpp
 *
 *  A simple program to poll three Dallas DS18B20 sensor and provide an output
 *  to the serial interface as a CSV format, with option for heat control.
 *
 *  Author:
 *  Created:    23 Jul 2015
 *  Modified:   24 Jul 2015
 *
 *  Code based from https://github.com/milesburton/Arduino-Temperature-Control-Library/blob/master/examples/Multiple/Multiple.pde
 *
 *  Notes:
 *      OneWire http://playground.arduino.cc/Learning/OneWire
 *
 *  Code will check for correct communications three DS18B20 sensors.
 *  Any error will be reported and functionality for reading sensor disabled.
 *
 *  The sensors will be polled every 10 seconds for their temperature and printed out.
 *
 *  Output to the serial interface should be as follows:
 *      1234567890, 21.5, 22.7, 23.9
 *      1234567891, 21.5, 22.6, 23.8
 *      and so on ....
 *
 *  The first item is the internal timer millis() and is only used
 *  as a reference and can be removed or altered as required.
 *
 *
 *  Heating:
 *      The heating control relay is connected to pin 4.
 *      The heatin is controlled by the temperature of sensor_1 and is operated
 *      when the temperature is below the HEAT_ON setpoint.
 *      The heat control is removed when the temperature is above
 *      the HEAT_OFF setpoint.
 *
 */
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS            3       // Data wire is plugged into pin 3 on the Arduino:
#define TEMPERATURE_PRECISION   9       // sensor precision set to 9 bit, can be higher, 10, 11 or 12 bits:
#define HEAT_RELAY              4       // heat relay is connected to pin 4:
#define HEAT_ON                 18.5    // heat on temperature:
#define HEAT_OFF                24.0    // heat off temperature:

//-------------------------------------------------------------------------------------------------------------------
//  const   type            variable            =   value;                          comment

            boolean         heat_relay          =   false;                  // state of heat relay:
            uint32_t        lastScan            =   0;                      // last time sensors are scanned:
    const   uint32_t        periodScan          =   10000;                  // rate period to scan sensors, every 10000mSec, or 10 seconds:

            OneWire         oneWire(ONE_WIRE_BUS);                          // Setup a oneWire instance to communicate with any OneWire devices
            DallasTemperature sensors(&oneWire);                            // Pass oneWire reference to Dallas Temperature.

    struct  sensor_t {                                                      // structure to hold info for each sensor:
                DeviceAddress   address;                                    // sensor unique address:
                float           value;                                      // actual temperature value from sensor in °C:
                boolean         enabled;                                    // set if sensor is enabled and online:
                uint16_t        error_count;                                // error count if needed:
            };

            sensor_t        sensor_1 = {0x28, 0xBC, 0xBF, 0xE9, 0x03, 0x00, 0x00, 0x7A};    // instance and address for sensor 1:
            sensor_t        sensor_2 = {0x28, 0x09, 0xA9, 0xC0, 0x03, 0x00, 0x00, 0x95};    // instance and address for sensor 2:
            sensor_t        sensor_3 = {0x25, 0x04, 0xBC, 0xC7, 0x03, 0x00, 0x00, 0x85};    // instance and address for sensor 3:

/*--------------------------------------------------------------------------------
 *  void setup()
 *  Initial setup of board:
 */
void setup() {
    Serial.begin(9600);
    sensors.begin();

    pinMode(HEAT_RELAY, OUTPUT);                                            // set heat relay as an output:

    if (!sensors.getAddress(sensor_1.address, 0))
    {
        Serial.println("Error communicating to sensor 1");
        sensor_1.enabled = false;
    } else
    {
        sensor_1.enabled = true;
        sensors.setResolution(sensor_1.address, TEMPERATURE_PRECISION);     // set the resolution to defined precision, ie 9 bit:
    }

    if (!sensors.getAddress(sensor_2.address, 0))
    {
        Serial.println("Error communicating to sensor 2");
        sensor_2.enabled = false;
    } else
    {
        sensor_2.enabled = true;
        sensors.setResolution(sensor_2.address, TEMPERATURE_PRECISION);     // set the resolution to defined precision, ie 9 bit:
    }

    if (!sensors.getAddress(sensor_3.address, 0))
    {
        Serial.println("Error communicating to sensor 3");
        sensor_3.enabled = false;
    } else
    {
```

```
104         sensor_3.enabled = true;
105         sensors.setResolution(sensor_3.address, TEMPERATURE_PRECISION);     // set the resolution to defined precision, ie 9 bit:
106     }
107 }
108
109 /*-----------------------------------------------------------------------------
110  *  void loop()
111  *  Mission Control:
112  *  Routine to get sensor data and print out at a scan rate of every 10000mSec:
113  */
114 void loop() {
115     if (millis() - lastScan > periodScan) {                                 // call routine at a set rate:
116         sensors.requestTemperatures();                                      // initiate all sensors to do conversion:
117
118         Serial.print(millis());                                             // print something first such as time if needed:
119         Serial.print(", ");                                                 // insert delimiter:
120         if (sensor_1.enabled)
121         {
122             sensor_1.value = sensors.getTempC(sensor_1.address);            // get sensor 1 temperature:
123             Serial.print(sensor_1.value, 1);
124             Serial.print(", ");                                             // insert delimiter:
125         }
126
127         if (sensor_2.enabled)
128         {
129             sensor_1.value = sensors.getTempC(sensor_1.address);            // get sensor 2 temperature:
130             Serial.print(sensor_2.value, 1);
131             Serial.print(", ");                                             // insert delimiter:
132         }
133
134         if (sensor_3.enabled)
135         {
136             sensor_1.value = sensors.getTempC(sensor_1.address);            // get sensor 3 temperature:
137             Serial.print(sensor_3.value, 1);
138             Serial.print("\n");                                             // if needed, print a newline:
139         }
140
141         if (sensor_1.enabled)                                               // this sensor used for relay control:
142         {
143             if ((sensor_1.value <= HEAT_ON) && !heat_relay)                 // if temp below setpoint and not heating:
144                 digitalWrite(HEAT_RELAY, true);                             // then turn on heat:
145
146             if ((sensor_1.value >= HEAT_OFF) && heat_relay)                 // if temp above setpoint and heating:
147                 digitalWrite(HEAT_RELAY, false);                            // then turn off heating:
148         }
149
150         lastScan = millis();
151     }
152 }
153
```