```
#include  <OneWire.h>
#include  <Adafruit_MotorShield.h>

// OneWire DS18S20, DS18B20, DS1822 Temperature Example
//
//  http://www.pjrc.com/teensy/td_libs_OneWire.html
//
// The DallasTemperature library can do all this work for you!
//   http://milesburton.com/Dallas_Temperature_Control_Library

OneWire  ds(10);  // on pin 10 (a 4.7K resistor is necessary)

Adafruit_MotorShield  shield  =  Adafruit_MotorShield();
Adafruit_DCMotor  *myMotor  =  shield.getMotor(1);

/*-----------------------------------------*/
float  temp(){
  byte i;
  byte present = 0;
  byte type_s;
  byte data[12];
  byte addr[8];
  float celsius;

  if ( !ds.search(addr)) {
    ds.reset_search();
    delay(250);
    return;
  }

  // the first ROM byte indicates which chip
  switch (addr[0]) {
    case 0x10:
      type_s = 1;
      break;
    case 0x28:
      type_s = 0;
      break;
    case 0x22:
      type_s = 0;
      break;
    default:
      return;
  }

  ds.reset();
```

```
  ds.select(addr);
  ds.write(0x44, 1);         // start conversion, with parasite power on at
the end

  delay(1000);     // maybe 750ms is enough, maybe not
  // we might do a ds.depower() here, but the reset will take care of it.

  present = ds.reset();
  ds.select(addr);
  ds.write(0xBE);          // Read Scratchpad

  for ( i = 0; i < 9; i++) {           // we need 9 bytes
    data[i] = ds.read();
  }

  // Convert the data to actual temperature
  // because the result is a 16 bit signed integer, it should
  // be stored to an "int16_t" type, which is always 16 bits
  // even when compiled on a 32 bit processor.
  int16_t raw = (data[1] << 8) | data[0];
  if (type_s) {
    raw = raw << 3; // 9 bit resolution default
    if (data[7] == 0x10) {
      // "count remain" gives full 12 bit resolution
      raw = (raw & 0xFFF0) + 12 - data[6];
    }
  } else {
    byte cfg = (data[4] & 0x60);
    // at lower res, the low bits are undefined, so let's zero them
    if (cfg == 0x00) raw = raw & ~7;  // 9 bit resolution, 93.75 ms
    else if (cfg == 0x20) raw = raw & ~3; // 10 bit res, 187.5 ms
    else if (cfg == 0x40) raw = raw & ~1; // 11 bit res, 375 ms
    //// default is 12 bit resolution, 750 ms conversion time
  }
  celsius = (float)raw / 16.0;

  return celsius;
}
/*-----------------------------------------*/
void runMotor(int dir, int speed){
  myMotor->setSpeed(speed);  // 0-255
   myMotor->run(dir);
  delay(500);
   //myMotor->run(RELEASE);
  //delay(1000);
}
```

```
/*-----------------------------------------*/

void setup(void) {
   Serial.begin(9600);
   shield.begin();
}

void loop(void) {
  float c = temp();
  Serial.print("  Temperature = ");
   Serial.print(c);
   Serial.println(" Celsius");
  int speed = map(c, 20, 30, 0, 255);
   Serial.println(speed);
   runMotor(BACKWARD, speed);
}
```