# ENV-TMP-D Arduino Sample Code

## Code Overview

This code has intentionally been written to be very lengthy and contain many unnecessary steps. Efficiency was not considered. This code was created to be overly lengthy and includes unnecessary steps. The primary focus of this code is easy of understanding.

The code is designed to output data to the Arduino serial monitor. This code can be truncated. The primary focus of this code is ease of understanding.

---

## Code Details

### Variables

- `float float_tmp = 0;` - stores the floating point number
- `byte startup = 0;` - flag used when the arduino is controlling the ENV-TMP-D
- `byte arduino_only = 0;` - flag used when the arduino is controlling the ENV-TMP-D
- `byte received_from_sensor = 0;` - flag used when the arduino is controlling the ENV-TMP-D
- `byte received_from_computer = 0;` - flag used when the arduino is controlling the ENV-TMP-D
- `char computerdata[20];` - character array to hold incoming data
- `char tmp_data[20];` - character array to hold incoming data
- `int string_received = 0;` - flag used when the arduino is controlling the ENV-TMP-D

### Setup

```cpp
//set the var arduino_only to =1 to watch the ENV-TMP-D.
//This code will output data to the arduino serial monitor.
//Code efficiency was not considered. Modify this code as you see fit.
//Many parts of this code can be truncated. Easy of understanding was the primary focus of this code.
//This code has intentionally has been written to be overly lengthy and includes unnecessary steps.

Atlas-Scientific.com
Copyright © Atlas Scientific LLC  All Rights  Reserved

---

## Command Sent to the ENV-TMP-D

The code sends the following commands to the ENV-TMP-D:

- `mySerial.print("d6\r");` - sends the "d6" command to set the scale to Fahrenheit
- `mySerial.print("sc\r");` - sends the "sc" command to set the scale to Celsius
- `mySerial.print("sf\r");` - sends the "sf" command to set the scale to Kelvin
- `mySerial.print("sk\r");` - sends the "sk" command to set the scale to Kelvin

### Data Processing

- `float float_tmp = atof(tmp_data);` - converts the string to a floating point number
- `tmp_data[received_from_sensor] = 0;` - sets the received character to 0
- `string_received = 1;` - sets the string_received flag to 1
- `received_from_computer = Serial.readBytesUntil(13,computerdata,20);` - reads bytes until a CR character is received
- `if (string_received == 1) {` - checks if the data has been received

### Data Sending

- `mySerial.begin(38400);` - initializes the serial port
- `mySerial.print(computerdata);` - prints data to the serial monitor
- `delay(50);` - delays for 50 milliseconds
- `mySerial.print(computerdata);` - prints data to the serial monitor

### Loop

- `void loop(){` - defines the loop function
- `if (startup==0) {` - checks if the startup is completed
- `if (arduino_only==1) {` - checks if the arduino_only flag is set
- `}` - ends the if statement
- `}` - ends the `if` statement
- `while (true) {` - infinite loop
- `}` - ends the `while` loop
- `}` - ends the `loop` function

---

## Notes

- **Reset Pin**
  - `RESET` is used to reset the Arduino
- **GND**
  - Ground pin
- **VCC**
  - Power supply pin
- **TX**
  - Transmit pin
- **RX**
  - Receive pin
- **USB**
  - USB port
- **ICSP**
  - In-Circuit Serial Programming

---

## Additional Resources

For more information, visit the Atlas Scientific website: [AtlasScientific.com](http://AtlasScientific.com)