```
#include <OneWire.h>
#include <DallasTemperature.h>
#include "thingProperties.h"
// Data wire is plugged into digital pin 2 on the Arduino
#define ONE_WIRE_BUS 8

// Setup a oneWire instance to communicate with any OneWire device
OneWire oneWire(ONE_WIRE_BUS);

// Pass oneWire reference to DallasTemperature library
DallasTemperature sensors(&oneWire);

#define SensorPin A0  //pH meter Analog output to Arduino Analog Input 0
#define Offset 0.00   //deviation compensate
#define LED 13
#define samplingInterval 20
#define printInterval 800
#define ArrayLenth 40     //times of collection
int pHArray[ArrayLenth];  //Store the average value of the sensor feedback
int pHArrayIndex = 0;

void setup(void)
{
  sensors.begin();           // Start up the library
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
  Serial.println("pH meter experiment!");  //Test the serial monitor
```

```
  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop(void)
{

  ArduinoCloud.update();
  // Send the command to get temperatures
  sensors.requestTemperatures();

  //print the temperature in Celsius
  Serial.print("Temperature: ");
  Serial.print(sensors.getTempCByIndex(0));
  Serial.print((char)176);//shows degrees character
  Serial.print("C  |  ");
  temp = sensors.getTempCByIndex(0);
  //pH
  static unsigned long samplingTime = millis();
  static unsigned long printTime = millis();
  static float pHValue, voltage;
  if (millis() - samplingTime > samplingInterval) {
```

```
    pHArray[pHArrayIndex++] = analogRead(SensorPin);
    if (pHArrayIndex == ArrayLenth) pHArrayIndex = 0;
    voltage = avergearray(pHArray, ArrayLenth) * 5.0 / 1024;
    pHValue = 3.5 * voltage + Offset;
    samplingTime = millis();
  }
  if (millis() - printTime > printInterval)  //Every 800 milliseconds, print a numerical, convert the state of the LED indicator
  {
    Serial.print("Voltage:");
    Serial.print(voltage, 2);
    Serial.print("    pH value: ");
    Serial.println(pHValue, 2);
    digitalWrite(LED, digitalRead(LED) ^ 1);
    printTime = millis();
  }
  pH= pHValue;
}
double avergearray(int* arr, int number) {
  int i;
  int max, min;
  double avg;
  long amount = 0;
  if (number <= 0) {
    Serial.println("Error number for the array to avraging!/n");
    return 0;
  }
  if (number < 5) {  //less than 5, calculated directly statistics
```

```
  for (i = 0; i < number; i++) {
    amount += arr[i];
  }
  avg = amount / number;
  return avg;
 } else {
  if (arr[0] < arr[1]) {
    min = arr[0];
    max = arr[1];
  } else {
    min = arr[1];
    max = arr[0];
  }
  for (i = 2; i < number; i++) {
    if (arr[i] < min) {
      amount += min;  //arr<min
      min = arr[i];
    } else {
      if (arr[i] > max) {
        amount += max;  //arr>max
        max = arr[i];
      } else {
        amount += arr[i];  //min<=arr<=max
      }
    }  //if
  }   //for
  avg = (double)amount / (number - 2);
```

```
  } //if
  return avg;
}
/*
  Since PH is READ_WRITE variable, onPHChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onPHChange()  {
  // Add your code here to act upon PH change
}
/*
  Since Temp is READ_WRITE variable, onTempChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onTempChange()  {
  // Add your code here to act upon Temp change
}
```