//ARDUINO CODE-------------------------------------------------------------------------------------------------------------------

//Lance Einfeld ENME351 FINAL PROJECT 5-12-2020

//Sensor readings for callibration-----------------------------------------

//moist ~(air water dry wet):~(595 280 460 360)

//moist1 basil ~(air water dry wet):(595 280 460 320)

//light window ~100

//light low/indoor ~260

//temp1 [10k]: 523@74 535@72 600@62

//temp2 [503]: 542@74 553@72 626@62

//configurables------------------------------------------------------------

int plantnum=1;//number of plants (1 to 4)

int setlevels[][4]={{350,480,100,525},//plant 1 {wetsoil,drysoil,light,temp} [sensor readings]

{360,490,260,543},//plant 2 " "

{0,0,0,0},

{0,0,0,0}};

int pumppins[]={2,3,99,99};//{plant1,2,3,4}(99 is no pin)

int sensorpins[][3]={{0,2,4},{1,3,5},//{{moist1,light1,temp1},{moist2,light2,...}}(99 is no pin)

{99,99,99},{99,99,99}};

//-------------------------------------------------------------------------

bool watering=false;

bool wait=false;

void setup() {

for(int i=0;i<plantnum;i++){

pinMode(pumppins[i],OUTPUT);

digitalWrite(pumppins[i],LOW);

}

Serial.begin(9600);

}

void loop() {

for(int i=0;i<plantnum;i++){//cycles plant to water, still reads all plants

int moist=analogRead(sensorpins[i][0]);

if(moist>=setlevels[i][1]){

watering=true;

}

while(watering==true){

if(wait==false){

digitalWrite(pumppins[i],HIGH);

for(int m=0;m<30;m++){//water for 30\*100 ms while reading out

printout(plantnum,setlevels,sensorpins);

delay(100);

wait=true;

}

}

else if(wait==true){

digitalWrite(pumppins[i],LOW);

for(int m=0;m<150;m++){//wait for 150\*100 ms while reading out

printout(plantnum,setlevels,sensorpins);

delay(100);

wait=false;

}

}

moist=analogRead(sensorpins[i][0]);

if(moist<=setlevels[i][0]){//check if moisture level is met

watering=false;

}

}

printout(plantnum,setlevels,sensorpins);

delay(100);

}

}

//print values to serial monitor

void printout(int x,int setlevels[][4],int sensorpins[][3]){

Serial.print(x);

for(int j=0;j<x;j++){

for(int k=0;k<4;k++){

Serial.print(",");

Serial.print(setlevels[j][k]);

}

for(int l=0;l<3;l++){

Serial.print(",");

Serial.print(analogRead(sensorpins[j][l]));

}

}

Serial.println();

}

//PROCESSING CODE---------------------------------------------------------------------------------------------------------------

//Lance Einfeld ENME351 FINAL PROJECT 5-12-2020

int[][] maxmin={{300,600},{0,1023},{400,700}};//{{maxmoist,minmoist},{maxlight,minlight},{maxtemp,mintemp}} set ranges

int plantnum;

String[][] headers={{"Moist 1","Light 1","Temp 1"},{"Moist 2","Light 2","Temp 2"},{"Moist 3","Light 3","Temp 3"},{"Moist 4","Light 4","Temp 4"}};

int[][][][] plotvalues=new int[2][4][3][481];

long totals[][][]=new long[2][4][3];

int[][][][] averagesmin=new int[2][4][3][481];

int[][] key={{0,0,225},{185,185,0},{225,50,0}};

int savecounter=0;

int lastminute=minute();

int passedminutes=0;

String start;

String finish;

import processing.serial.\*;

Serial myPort;

// Data table and filename specifications

Table table;

void setup() {

start=str(month())+"-"+str(day())+"-"+str(year())+"--"+str(hour())+"-"+str(minute());

for(int p=0;p<2;p++){

for(int l=0;l<4;l++){

for(int j=0;j<3;j++){

totals[p][l][j]=0;

for(int k=0;k<481;k++){

plotvalues[p][l][j][k]=0;

averagesmin[p][l][j][k]=0;

}

}

}

}

size(1200, 800); // set the window size

//println(Serial.list()); // list all available serial ports

myPort = new Serial(this, Serial.list()[0], 9600); // define input port

myPort.clear(); // clear the port of any initial junk

table = new Table(); // Create a table to save data as a csv file

table.addColumn("Time");

for(int q=0;q<4;q++){

for(int s=0;s<3;s++){

table.addColumn(headers[q][s]);

}

}

}

void draw () {

int[] mdyhm={month(),day(),year(),hour(),minute()};

while (myPort.available () > 0) {

String inString = myPort.readStringUntil('\n');

if (inString != null) {

inString = trim(inString);

String[] rawdata = splitTokens(inString, ",");

if (rawdata.length == 1+7\*int(rawdata[0])) {

noLoop();

background(255);

plantnum=int(rawdata[0]);

fill(0);

textSize(30);

text("Current Readings",10,30);

text("Last 48 Hours",550,30);

textSize(20);

text("["+str(mdyhm[0])+"/"+str(mdyhm[1])+"/"+str(mdyhm[2])+" "+str(mdyhm[3])+":"+str(mdyhm[4])+"]",270,30);

text("Plant",10,70);

fill(key[0][0],key[0][1],key[0][2]);

text("Moisture",380,70);

fill(key[1][0],key[1][1],key[1][2]);

text("Light",240,70);

fill(key[2][0],key[2][1],key[2][2]);

text("Temp",100,70);

for(int i=0; i<plantnum;i++){

int[] data= {int(rawdata[1+7\*i]),int(rawdata[2+7\*i]),int(rawdata[3+7\*i]),int(rawdata[4+7\*i]),int(rawdata[5+7\*i]),int(rawdata[6+7\*i]),int(rawdata[7+7\*i])};//(set) moistwet,moistdry, light, temp, (live) moist, light, temp

float a=150-map(data[0],maxmin[0][1],maxmin[0][0],0,150)+90+i\*180;

float b=150-map(data[1],maxmin[0][1],maxmin[0][0],0,150)+90+i\*180;

float c=150-map(data[2],maxmin[1][1],maxmin[1][0],0,150)+90+i\*180;

float d=150-map(data[3],maxmin[2][1],maxmin[2][0],0,150)+90+i\*180;

int[] scale={int(map(data[4],maxmin[0][0],maxmin[0][1],0,150)),int(map(data[5],maxmin[1][0],maxmin[1][1],0,150)),int(map(data[6],maxmin[2][0],maxmin[2][1],0,150))};

//plot lines

fill(0);

line(550,240+i\*180,1030,240+i\*180);

line(550,90+i\*180,550,240+i\*180);

for(int f=1;f<49;f++){

line(550+10\*f,243+i\*180,550+10\*f,237+i\*180);

}

textSize(12);

stroke(key[0][0],key[0][1],key[0][2]);

fill(key[0][0],key[0][1],key[0][2]);

text(str(data[0]),520, a);

text(str(data[1]),520, b);

line(540, a, 1030, a);

line(540, b, 1030, b);

stroke(key[1][0],key[1][1],key[1][2]);

fill(key[1][0],key[1][1],key[1][2]);

text(str(data[2]),520, c);

line(540, c, 1030, c);

stroke(key[2][0],key[2][1],key[2][2]);

fill(key[2][0],key[2][1],key[2][2]);

line(540, d, 1030, d);

text(str(data[3]),520, d);

fill(0);

textSize(30);

text(str(i+1),20,165+i\*180);

for(int m=0;m<3;m++){

plotvalues[0][i][m][480]=data[4+m];

plotvalues[1][i][m][480]=int(map(data[4+m],maxmin[m][1],maxmin[m][0],0,150));

totals[0][i][m]=totals[0][i][m]+plotvalues[0][i][m][480];

totals[1][i][m]=totals[1][i][m]+plotvalues[1][i][m][480];

stroke(key[m][0],key[m][1],key[m][2]);

fill(key[m][0],key[m][1],key[m][2]);

for(int n=0;n<480;n++){

ellipse(550+n,240+i\*180-plotvalues[1][i][m][n],2,2);

//ellipse(550+n,240+i\*180-averagesmin[1][i][m][n],2,2);

}

}

stroke(0);

fill(0);

//level text

textSize(20);

text(str(data[4]),425,90+i\*180+scale[0]);

text(str(data[5]),285,90+i\*180+scale[1]);

text(str(data[6]),145,90+i\*180+scale[2]);

textSize(12);

text(str(data[0]),355, a);

text(str(data[1]),355, b);

text(str(data[2]),215, c);

text(str(data[3]),75, d);

text(str(maxmin[0][1]),355,240+i\*180);

text(str(maxmin[1][1]),208,240+i\*180);

text(str(maxmin[2][1]),75,240+i\*180);

text(str(maxmin[0][0]),355,90+i\*180);

text(str(maxmin[1][0]),230,90+i\*180);

text(str(maxmin[2][0]),75,90+i\*180);

//level bars

fill(map(sqrt(abs(pow(data[0],2)-pow(data[4],2))),0,maxmin[0][1]-data[0],0,200),map(sqrt(abs(pow(data[0],2)-pow(data[4],2))),0,maxmin[0][1]-data[0],255,50),0);

rect(380,90+i\*180,40,150);

fill(map(sqrt(abs(pow(data[2],2)-pow(data[5],2))),0,maxmin[1][1]-data[2],0,200),map(sqrt(abs(pow(data[2],2)-pow(data[5],2))),0,maxmin[1][1]-data[2],255,50),0);

rect(240,90+i\*180,40,150);

fill(map(sqrt(abs(pow(data[3],2)-pow(data[6],2))),0,maxmin[2][1]-data[3],0,200),map(sqrt(abs(pow(data[3],2)-pow(data[6],2))),0,maxmin[2][1]-data[3],255,50),0);

rect(100,90+i\*180,40,150);

fill(255);

rect(380,90+i\*180,40,scale[0]);

rect(240,90+i\*180,40,scale[1]);

rect(100,90+i\*180,40,scale[2]);

//set level lines

line(375, a, 425, a);

line(375, b, 425, b);

line(235, c, 285, c);

line(95, d, 145, d);

finish=str(month())+"-"+str(day())+"-"+str(year())+"--"+str(hour())+"-"+str(minute());

}

for(int p=0;p<2;p++){

for(int l=0;l<plantnum;l++){

for(int j=0;j<3;j++){

for(int k=0;k<480;k++){

plotvalues[p][l][j][k]= plotvalues[p][l][j][k+1];

}

}

}

}

savecounter=savecounter+1;

if(lastminute!=mdyhm[4]){

passedminutes=passedminutes+1;

if (passedminutes==1){//time interval for average

for(int p=0;p<2;p++){

for(int l=0;l<plantnum;l++){

for(int j=0;j<3;j++){

averagesmin[p][l][j][480]=int((float)totals[p][l][j]/savecounter);

totals[p][l][j]=0;

for(int k=0;k<480;k++){

averagesmin[p][l][j][k]=averagesmin[p][l][j][k+1];

}

}

}

}

passedminutes=0;

TableRow newRow = table.addRow();

newRow.setString("Time",str(mdyhm[0])+"/"+str(mdyhm[1])+"/"+str(mdyhm[2])+" "+str(mdyhm[3])+":"+str(mdyhm[4]));

for(int q=0;q<plantnum;q++){

for(int s=0;s<3;s++){

newRow.setInt(headers[q][s], averagesmin[0][q][s][480]);

}

}

savecounter=0;

}

}

lastminute=mdyhm[4];

loop();

}}

saveTable(table, "PlantManager\_log.csv");

}

}