**//OTHER FUNCTION ON 2ND PAGE**

//Line-Following functions

//Function to compute sensor values

void follow() {

//Zero avg[]

for (int i = 0; i < 8; i++) {

avg[i] = 0;

}

//Get 10 data points from each sensor

//(How long does this for-loop take?)

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

qtrrc.read(sensorValues);

for (int i = 0; i < 8; i++) {

fArray[10 \* i + j] = sensorValues[i];

}

}

//Compute average sensor value

for (int i = 0; i < 8; i++) {

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

avg[i] = avg[i] + fArray[10 \* i + j];

}

}

for (int i = 0; i < 8; i++) {

avg[i] = avg[i] / FOLNUM;

}

}

//Function to determine robot drift and adjust accordingly

void adjust() {

//Figure out whether Robot is drifting and in which direction

//Check if it has drifted to right and, if so, how severely

//First check sensor6 to see if it has drifted

if (avg[5] > wMax[5] + DRIFTC) {

//check to see if robot was last seen drifting to right

if(driftState == 1){

adjC++;

}

//drifted to right, need to run rightmost motor slightly faster and leftmost slightly slower to correct

analogWrite(enA, BASESPEED + adjC);

analogWrite(enB, BASESPEED - adjC);

driftState = 1; //store that we last drifted to the right

}

//Now check to see if it has drifted to left

if (avg[3] > wMax[3] + DRIFTC) {

//check to see whether robot was last seen drifting to left

if(driftState == -1){

adjC++;

}

//drifted to left, need to run rightmost motor slightly slower and leftmost slightly faster to correct

analogWrite(enA, BASESPEED - adjC);

analogWrite(enB, BASESPEED + adjC);

driftState = -1;

}

else{

driftState = 0;

//this might need to be moved or just straight up removed

adjC--;

}

}