```
/* Initial Robot Testing
 *  20200117
 *  Program to get your robot's unique driving constants
 */

/* Program TODO LIST
  1) Write code for the button pause functionality
  2) Experiment with the wire connections to your motors until
     positive motor commands cause both motors to drive forward.
     Switch the two wires of a motor to flip the direction it spins
  3) Find the relative power settings to drive completely straight A 200 B 210
     What pwms do you put to each motor so they spin at the same speed?
  4) Find the time in ms to drive 1 cm 22ms/ cm
     Change this program to drive straight for a set amount of time. Measure
     the distance traveled, and divide. Use the average of multiple trials.
  5) Find the time in ms to turn 90 degrees
     Change this program to spin in place for a set amount of time. Measure
     the total degrees traveled, and divide. Use the average of multiple trials. 273 ms
*/

#define pushButton 2
// If you have a kit with the moto shield, set this to true
// If you have the Dual H-Bridge controller w/o the shield, set to false
#define SHIELD false

// Defining these allows us to use letters in place of binary when
// controlling our motor(s)
#define A 0
#define B 1

//SHIELD Pin varables
#define motorApwm 3
#define motorAdir 12
#define motorBpwm 11
#define motorBdir 13

//Driver Pin variable
#define IN1 9
#define IN2 10
#define IN3 5
#define IN4 6
bool RUN = false;

void setup()
{
    // set up the motor drive ports
    motor_setup();
    // make the pushbutton's pin an input:
    pinMode(pushButton, INPUT_PULLUP); //CHANGE TO INPUT_PULLUP
    // initialize serial communication at 9600 bits per second:
    Serial.begin(9600);
}

void loop()
{

    /* Write code here to make the program pause until
     * the button has been pressed. Remember that getting a value
```

```
     * from the board is digitalRead(pinNumber) and if statements
     * need a double equals sign (==) for comparisons
     */
    if (!digitalRead(pushButton))
    {
        /*
    * Change your wiring so that positive numbers drive forward
    * After wiring has been fixed:
    * To drive forward, motors should have positive commands
    * To turn Left, the left motor should have a negative number,
    *               the right motor should have a positive number
    * To turn Right, the left motor should have a positive number,
    *                the right motor should have a negative number
    */

        // Run once to calibrate the motors driving straight
        for (int i = 0; i < 1; i++)
        {
            // Give me time to get my hand away from the robot
            delay(1000);
            // Found Motor A needs to be lower
            run_motor(A, 185); //set this to a number between -255 and 255
            run_motor(B, 200); //set this to a number between -255 and 255
            delay(2700);       //set this to a time in ms for the motors to
run
            run_motor(A, 0);   //motors stop
            run_motor(B, 0);
            delay(750);
        }

        // RIGHT
        {
            run_motor(A, 200);  //set this to a number between -255 and 255
            run_motor(B, -200); //set this to a number between -255 and 255
            delay(900);         //set this to a time in ms for the motors to
run
            run_motor(A, 0);    //motors stop
            run_motor(B, 0);
        }
        delay(2000);

        // LEFT
        {
            run_motor(A, -200); //set this to a number between -255 and 255
            run_motor(B, 200);  //set this to a number between -255 and 255
            delay(900);         //set this to a time in ms for the motors to
run
            run_motor(A, 0);    //motors stop
            run_motor(B, 0);
        }
    }
}

```