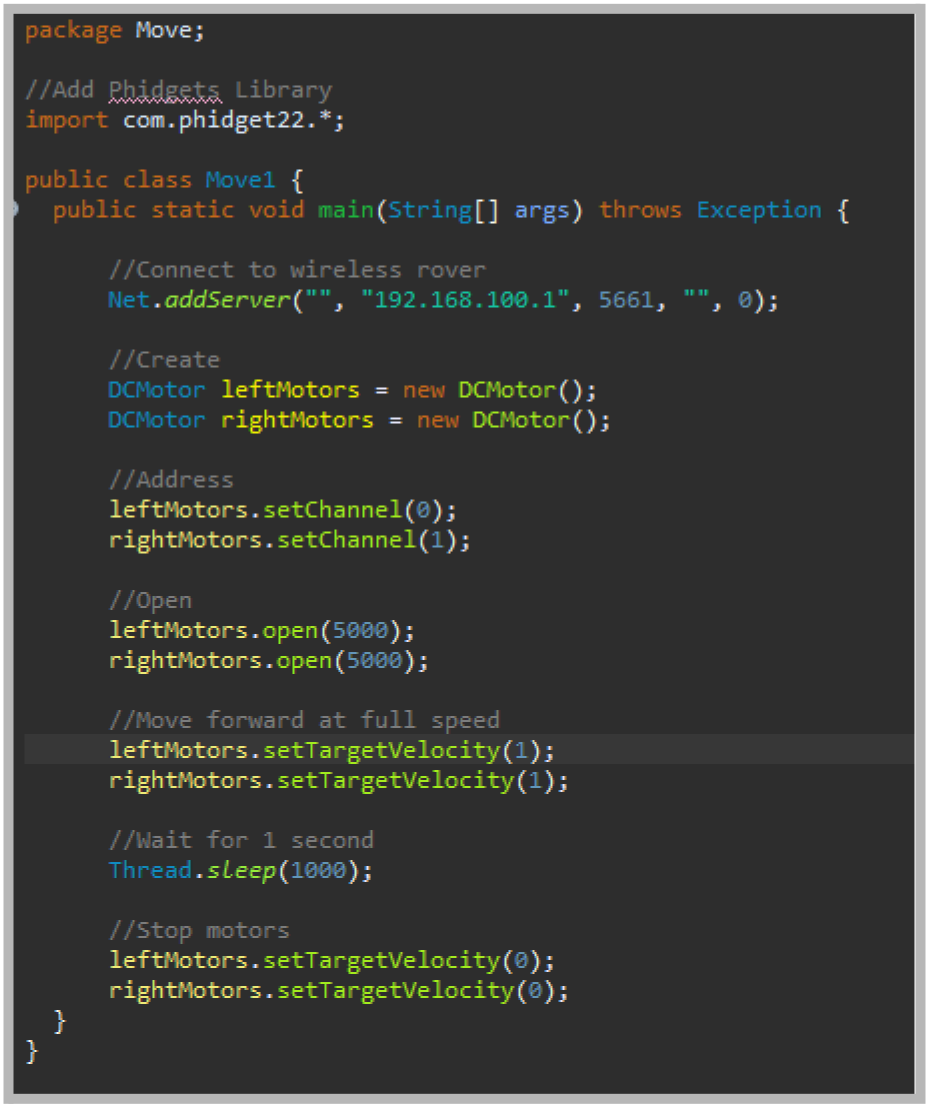
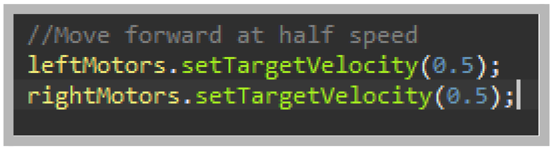
Credit Name: Project D
Assignment Name: Phidgets

Move:



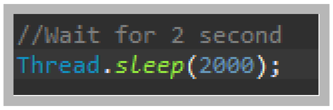
```
package Move;

//Add Phidgets Library
import com.phidget22.*;

public class Move1 {
  public static void main(String[] args) throws Exception {

      //Connect to wireless rover
      Net.addServer("", "192.168.100.1", 5661, "", 0);

      //Create
      DCMotor leftMotors = new DCMotor();
      DCMotor rightMotors = new DCMotor();

      //Address
      leftMotors.setChannel(0);
      rightMotors.setChannel(1);

      //Open
      leftMotors.open(5000);
      rightMotors.open(5000);

      //Move forward at full speed
      leftMotors.setTargetVelocity(1);
      rightMotors.setTargetVelocity(1);

      //Wait for 1 second
      Thread.sleep(1000);

      //Stop motors
      leftMotors.setTargetVelocity(0);
      rightMotors.setTargetVelocity(0);
  }
}
```

I started the move lesson by copying and pasting the code that was on the website.



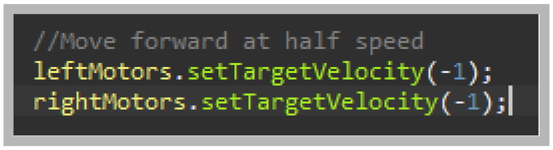
```
//Move forward at half speed
leftMotors.setTargetVelocity(0.5);
rightMotors.setTargetVelocity(0.5);
```

Next I modified the speed to half of its maximum to complete practice 1 which slowed it down.



```
//Wait for 2 second
Thread.sleep(2000);
```

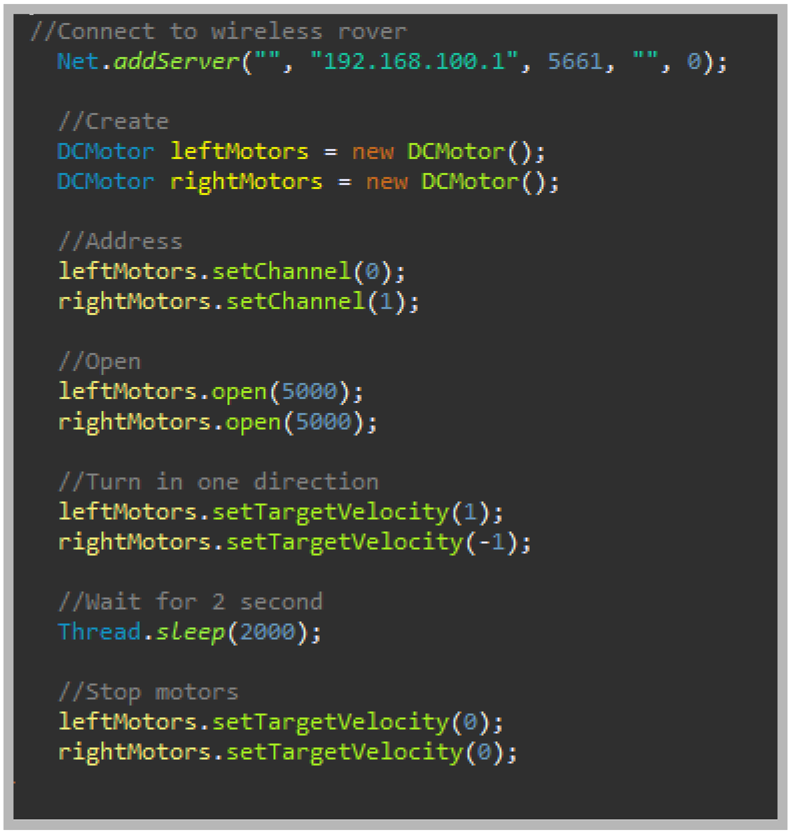
I then started work on the second practice problem and changed the sleep duration to 2 seconds, making it run for twice as long.



```
//Move forward at half speed
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(-1);
```

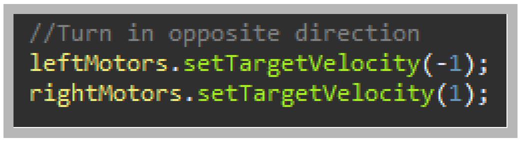
Finally to finish off the move lesson I changed the velocity to -1 and the bot now went forwards instead of backwards.

Turn:



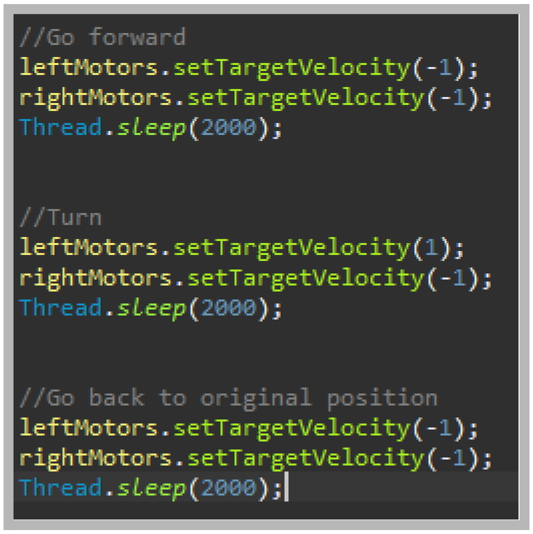
```
//Connect to wireless rover
  Net.addServer("", "192.168.100.1", 5661, "", 0);

  //Create
  DCMotor leftMotors = new DCMotor();
  DCMotor rightMotors = new DCMotor();

  //Address
  leftMotors.setChannel(0);
  rightMotors.setChannel(1);

  //Open
  leftMotors.open(5000);
  rightMotors.open(5000);

  //Turn in one direction
  leftMotors.setTargetVelocity(1);
  rightMotors.setTargetVelocity(-1);

  //Wait for 2 second
  Thread.sleep(2000);

  //Stop motors
  leftMotors.setTargetVelocity(0);
  rightMotors.setTargetVelocity(0);
```

I then started to work on the Turn class and I copied the code and pasted it and the bot successfully turned.



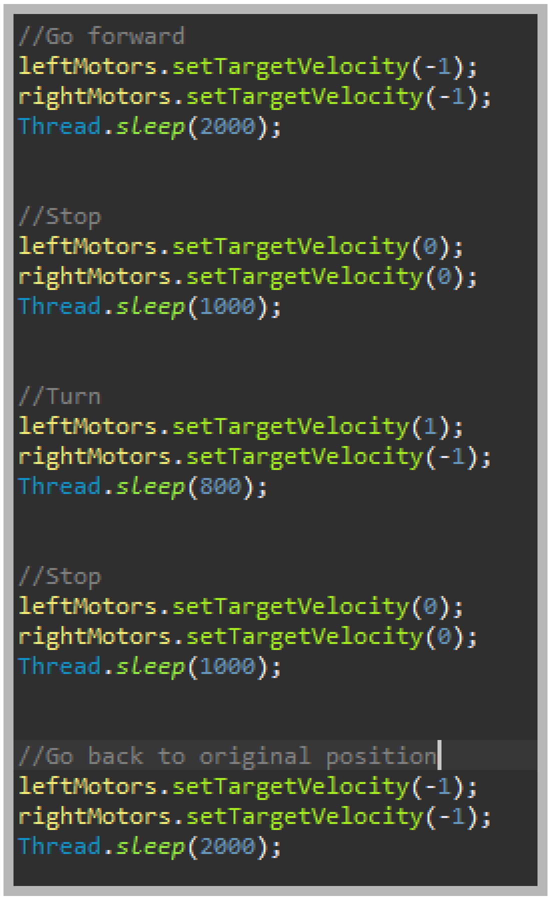
```
//Turn in opposite direction
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(1);
```

I then tried the first practice and simply changed the two velocities to be reversed from what they were in the original Turn class.



```
//Go forward
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(2000);


//Turn
leftMotors.setTargetVelocity(1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(2000);


//Go back to original position
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(2000);
```

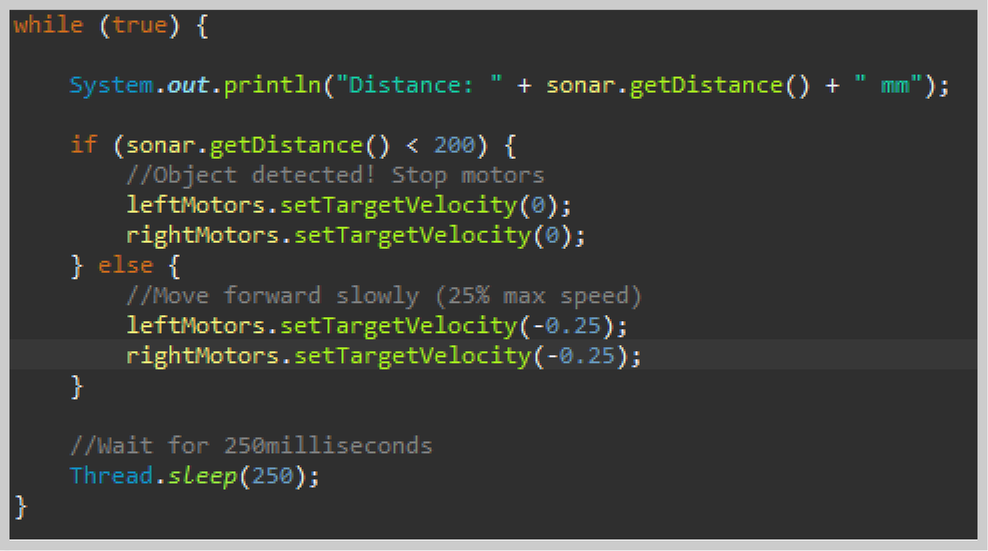
Next I started working on the second practice which I started with the basics of moving forwards, turning, and moving back. I didn’t know how long I wanted them to go for so I tested it and changed the code.



```
//Go forward
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(2000);


//Stop
leftMotors.setTargetVelocity(0);
rightMotors.setTargetVelocity(0);
Thread.sleep(1000);


//Turn
leftMotors.setTargetVelocity(1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(800);


//Stop
leftMotors.setTargetVelocity(0);
rightMotors.setTargetVelocity(0);
Thread.sleep(1000);


//Go back to original position
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(2000);
```

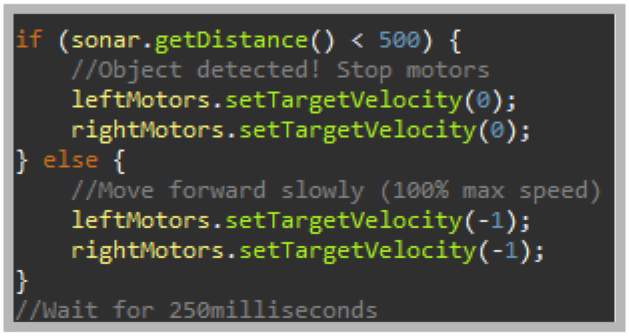
I added code to stop the motors after each step was completed for consistency. I also modified the timing of the turning by testing with trial and error until I landed on 800ms which worked most of the time.

Avoid Obstacles:



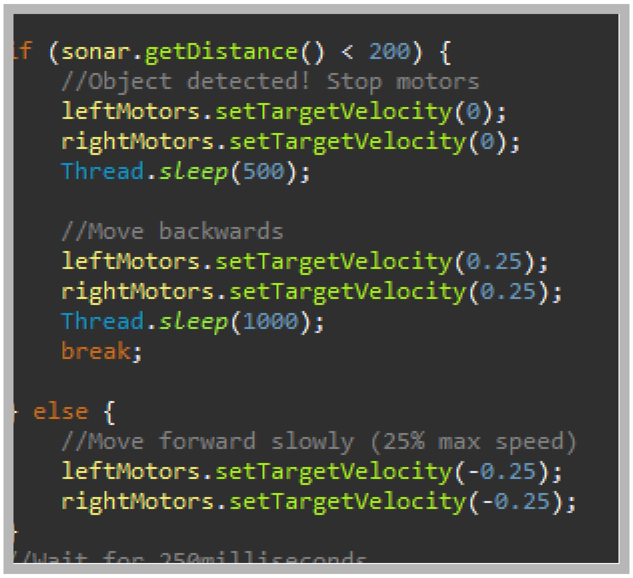
```
while (true) {

    System.out.println("Distance: " + sonar.getDistance() + " mm");

    if (sonar.getDistance() < 200) {
        //Object detected! Stop motors
        leftMotors.setTargetVelocity(0);
        rightMotors.setTargetVelocity(0);
    } else {
        //Move forward slowly (25% max speed)
        leftMotors.setTargetVelocity(-0.25);
        rightMotors.setTargetVelocity(-0.25);
    }

    //Wait for 250milliseconds
    Thread.sleep(250);
}
```

When I moved onto Avoid Obstacles I started by copying the code on the website and it seemed to work well so I moved on to the practice problems.



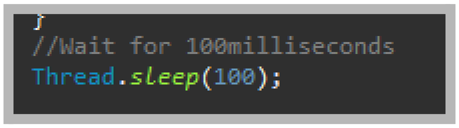
```
if (sonar.getDistance() < 500) {
    //Object detected! Stop motors
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
} else {
    //Move forward slowly (100% max speed)
    leftMotors.setTargetVelocity(-1);
    rightMotors.setTargetVelocity(-1);
}
//Wait for 250milliseconds
```

For the first practice problem I tried modifying the speed to max and changing the distance it would stop at to be higher to account for that. The bot still crashed into the wall but more gently than if it was still at 200.

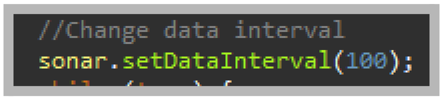


```java
if (sonar.getDistance() < 200) {
    //Object detected! Stop motors
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
    Thread.sleep(500);

    //Move backwards
    leftMotors.setTargetVelocity(0.25);
    rightMotors.setTargetVelocity(0.25);
    Thread.sleep(1000);
    break;

} else {
    //Move forward slowly (25% max speed)
    leftMotors.setTargetVelocity(-0.25);
    rightMotors.setTargetVelocity(-0.25);
}
//Wait for 250milliseconds
```

For the second practice problem I added code to stop the phidget and then back it up as soon as it sensed an object.



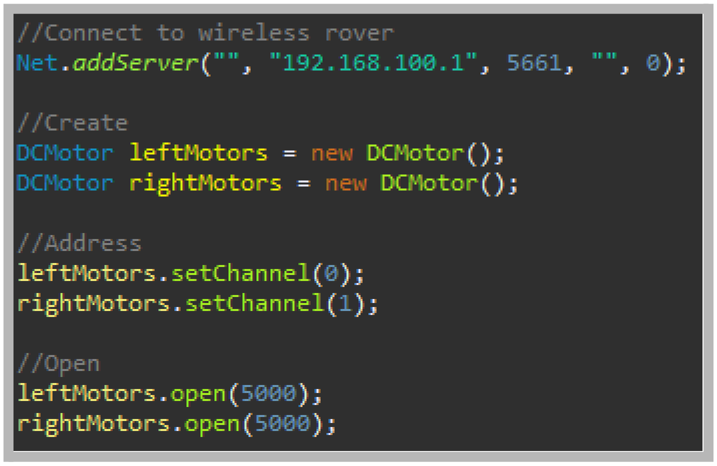
```java
}
//Wait for 100milliseconds
Thread.sleep(100);
```



```java
//Change data interval
sonar.setDataInterval(100);
```

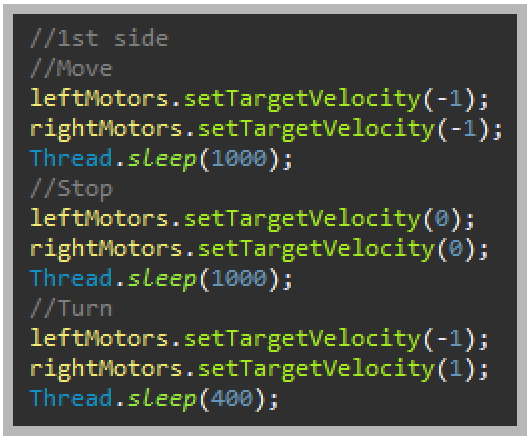
Next for the final practice problem I changed the time that the program would wait every 10th of a second and I went back to the getting started phidgets to remember to add the setDataInterval method.

Challenge:



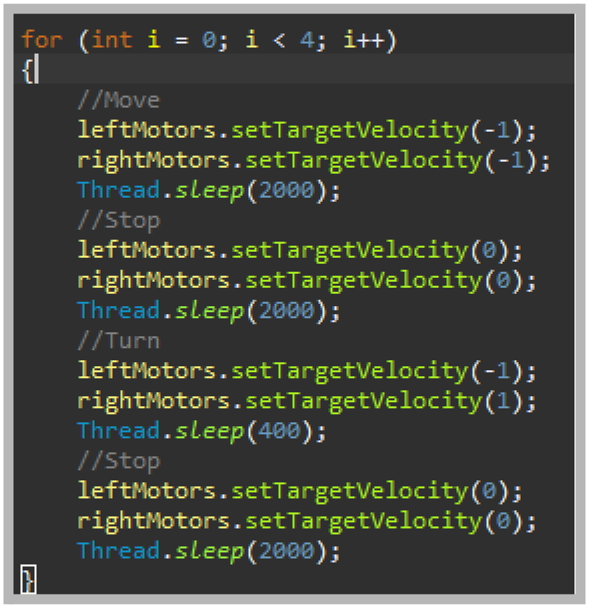
```java
//Connect to wireless rover
Net.addServer("", "192.168.100.1", 5661, "", 0);

//Create
DCMotor leftMotors = new DCMotor();
DCMotor rightMotors = new DCMotor();

//Address
leftMotors.setChannel(0);
rightMotors.setChannel(1);

//Open
leftMotors.open(5000);
rightMotors.open(5000);
```

Finally for Challenge I started from scratch adding the basics.



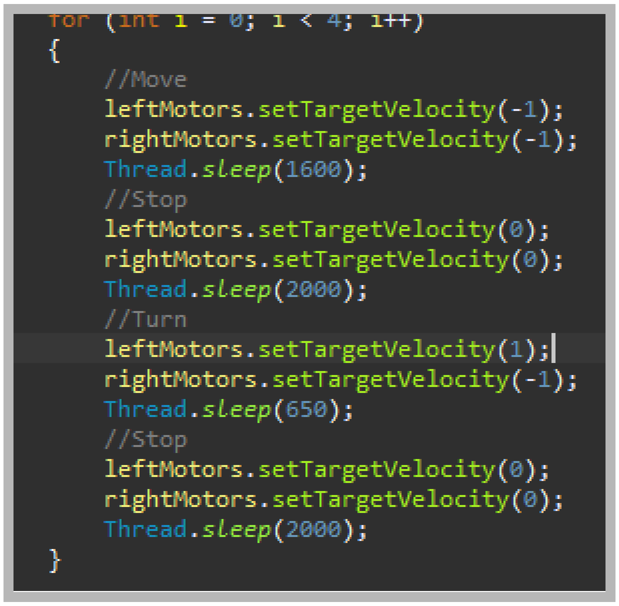
```
//1st side
//Move
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(-1);
Thread.sleep(1000);
//Stop
leftMotors.setTargetVelocity(0);
rightMotors.setTargetVelocity(0);
Thread.sleep(1000);
//Turn
leftMotors.setTargetVelocity(-1);
rightMotors.setTargetVelocity(1);
Thread.sleep(400);
```

I then added 4 sections of the same code for moving forward, stopping and then turning which I would edit the values for later.



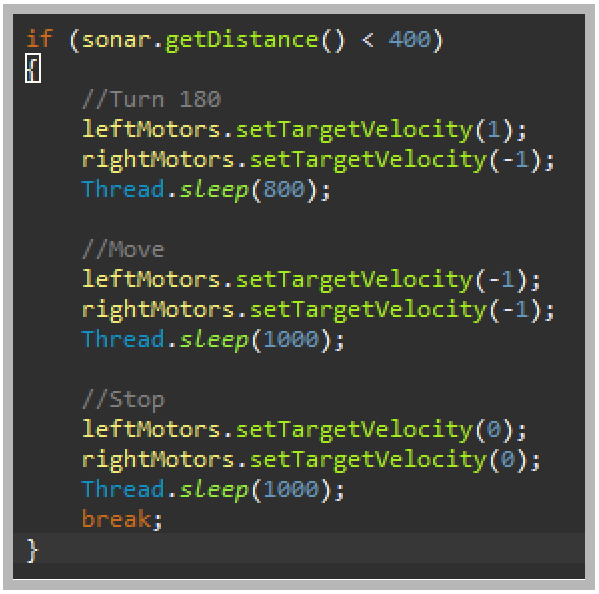
```
for (int i = 0; i < 4; i++)
{
    //Move
    leftMotors.setTargetVelocity(-1);
    rightMotors.setTargetVelocity(-1);
    Thread.sleep(2000);
    //Stop
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
    Thread.sleep(2000);
    //Turn
    leftMotors.setTargetVelocity(-1);
    rightMotors.setTargetVelocity(1);
    Thread.sleep(400);
    //Stop
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
    Thread.sleep(2000);
}
```

I decided to create a for loop instead of having multiple sections of the same code because it was easier to edit. I also added a pause at the end of the turn to make the turning more precise.



```
for (int i = 0; i < 4; i++)
{
    //Move
    leftMotors.setTargetVelocity(-1);
    rightMotors.setTargetVelocity(-1);
    Thread.sleep(1600);
    //Stop
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
    Thread.sleep(2000);
    //Turn
    leftMotors.setTargetVelocity(1);
    rightMotors.setTargetVelocity(-1);
    Thread.sleep(650);
    //Stop
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
    Thread.sleep(2000);
}
```

After some more testing I ended up with these values. It was able to somewhat consistently move in a square with the errors that the dust floor brang.



```
if (sonar.getDistance() < 400)
{
    //Turn 180
    leftMotors.setTargetVelocity(1);
    rightMotors.setTargetVelocity(-1);
    Thread.sleep(800);

    //Move
    leftMotors.setTargetVelocity(-1);
    rightMotors.setTargetVelocity(-1);
    Thread.sleep(1000);

    //Stop
    leftMotors.setTargetVelocity(0);
    rightMotors.setTargetVelocity(0);
    Thread.sleep(1000);
    break;
}
```

Finally I added the code that would turn the bot and make it move backwards if it sensed an object and I completed the challenge.