Unit 5: Rover's Sensors **Skill Builder 2: Ranger and Movement**

In the first lesson of Unit 5, you tested the Rover's Ranger to see how to read the sensor and display a value. This lesson advances that testing to control the Rover's motion.

Objectives:

- Use the READ RV.RANGER command to determine the distance to an obstacle
- Control the Rover's movement when the Rover gets too close to an obstacle.
- Manage the timing of the Rover's movement within the handheld program.

Let's write a program to make the Rover move back and forth between two 'walls'. We'll start the Rover moving FORWARD, read the Ranger sensor and, when the Rover gets too close to the wall, make the Rover stop, turn around, and move FORWARD.

The Big Idea

In a For loop (which will eventually terminate)

Start the Rover moving FORWARD

While the distance is greater than about 25 cm

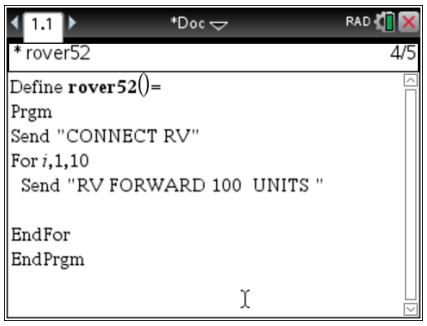
continue to monitor the Ranger sensor

End the While loop

STOP, turn RIGHT 180

End the For loop

1. Begin the program in the usual manner.
2. Add a **For** loop that will iterate ten times.
3. Add a **FORWARD 100 UNITS** command to move ahead 10 meters (100 * 0.1m per unit)

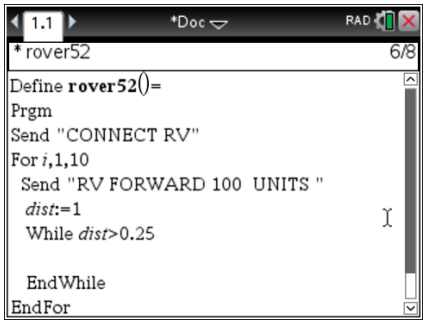


Teacher Tip: There's more code to go into the For loop body. At this point in the program, it appears that the Rover will move FORWARD 100 ten times. This is not the case in the final program because the Ranger will cause the Rover to stop and turn around. The FORWARD command goes into the 'drive queue' in the TI-Innovator™ Hub and the TI-Innovator Hub begins processing this queue immediately.

As in the previous lesson, we'll use the variable ***dist*** to represent ***dist***ance from the Ranger to an obstacle.

4. Initialize ***dist*** to 1, and insert a **While *dist* > 0.25** loop.

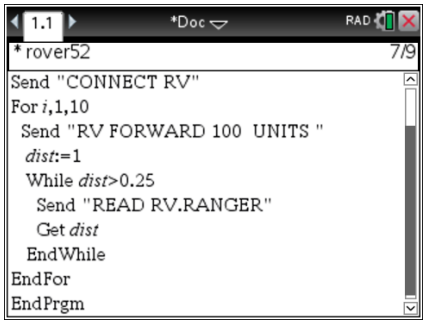
Note the two **End**s in the code: an **EndFor** and an **EndWhile**. This is called 'nested loops'.



5. We next insert the code into the body of the **While** loop.

6. Add the **Send "READ RV.RANGER"** command.

7. Use the **Get** command to store the value into the variable ***dist***.



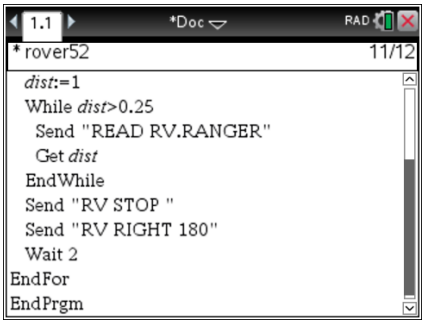
That completes the **While** loop. The Rover is moving FORWARD 10 meters and the **While** loop is monitoring the distance. If desired, add a **DispAt** statement to this **While** loop to display the current distance to make sure things are working properly.

When the **While** loop ends, this indicates that the Rover is too close to an obstacle. We'll tell the Rover to STOP moving FORWARD and turn around. However, we do not need to tell Rover to move FORWARD again.

8. *After* the **End** of the **While** loop but *before* the **End** of the **For** loop, add the RV **STOP** command and the **RV RIGHT 180** command.

9. Add the **Wait 2** statement to give the Rover time to turn around before moving **FORWARD** again. (Remember that the **FORWARD** command is at the beginning of the **For** loop.)

10. Test your program. When the Rover gets close to an obstacle, it should turn around and go in the opposite direction. Adjust values (minimum distance and **Wait** time), as needed, for your surface situation. If the Rover gets too close to the wall, then the back of the Rover could bump the wall when turning around.



Teacher Tip: A complete program (indenting for clarity only):

```
Send "CONNECT RV"
For i,1,10
  Send("RV FORWARD 100")
  1→d
  While d>0.25
     Send "READ RV.RANGER"
     Get d
  EndWhile
  Send "RV STOP "
  Send "RV RIGHT 180"
  Wait 2
EndFor
```