Overview:

Students will use measurement of distance and angles to write code to explore a set of challenges. Students will apply their knowledge of the relationships among distance, rate and time to write equations to optimize routes through a city, “Math-hattan”, or other maze with obstacles. This activity is appropriate for students familiar with modeling and solving relationships involving distance, rate, and time.

Goals:

Students will:

- use their knowledge of angles, writing equations, and the relationship $\text{dist} = \text{rate} \cdot \text{time}$ in the context of challenge problems.
- create and edit TI-Nspire CXII Python programs that include several commonly used Rover and calculator commands.

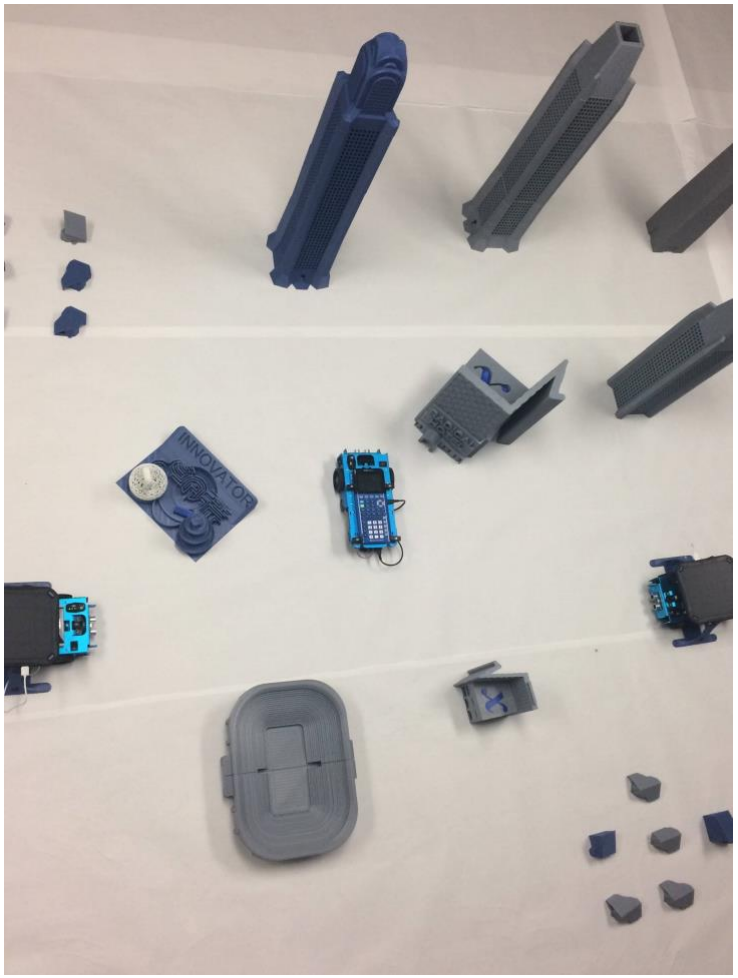
Python Syntax Reference:

Statement	Example	Behavior
import module_name as name_space	import ti_rover as rv	Required for all TI Rover Python programs. Imports the ti_rover module into the Python program. The module provides the methods for controlling the Rover. Sets the current position of the RV as the origin and the heading as 0 degrees measured from the x-axis.
rv.forward_time(time,speed,"unit")	rv.forward_time(3,.23,"m/s")	Rover drives forward at a rate of 23 cm/sec for a time of 3 seconds. Note that the rate at which Rover drives will vary for different surfaces which affects how far it travels. The SPEED specification accepts 0.14 M/S to 0.23 M/S. rv.forward_time() is available from the Rover> Drive> Drive with Options menu.
rv.backward(distance,"unit",speed,"unit")	rv.backward(1,"m",.15,"m/s")	Rover drives backward a distance of 1 meter at a rate of 0.15 m/s, or 15 cm/sec. rv.backward() with distance and time options is available from the Rover> Drive> Drive with Options menu.
rv.right (angle_degrees)	rv.right(60)	Rover turns 60 degrees right from its current heading.
rv.left (angle_degrees)	rv.left(45)	Rover turns 45 degrees left from its current heading.
rv.to_xy (x-coordinate, y-coordinate)	rv.to_xy(3,4)	Rover turns and drives along a straight line to the point (3, 4) as defined by Rover’s internal coordinate system.

For more on programming Rover with TI-Nspire CXII follow the links to the TI Rover Menu Map: TI-Nspire™ Python Programming > Python Menu Map > TI Rover Menu.

Setup:

Students may work in groups of two or three. An example “Math-hattan” city layout is shown below. Be sure roads, and predicted turns are at least 30 cm wide to allow for enough navigation room for the Rover on the course. Note: Ideal surface for this activity is a gym floor or cafeteria floor that will provide a smooth surface, as well as ample space for Rover driving. Surface type will affect how fast Rover drives and turns.



Materials::

- TI-Nspire CXII Graphing Calculator
- Calculator unit-to-unit cable (USB mini A to USB mini B cable)
- TI-Innovator Hub
- TI-Innovator Rover
- “City Scape”
    - Buildings: 3D printed buildings or boxes representing different buildings in a city. The files for printing some of the buildings are posted with the lesson. Masking tape of different colors can work as well- making it clear to students these are solid structures.
    - Roadways: Masking tape to mark out roadways.
    - Obstacles: Miniature traffic cones to temporarily block paths, and challenge students as needed, once the activity has begun.
- Meter sticks or measuring tapes and protractors

Student Activity:	Teacher Notes:
Sit in small groups with your calculator and supplies for this activity. Practice the guidance modeled by your teacher.	Review and introduce the calculator, Hub, and Rover commands needed for this activity. In preparation for the coding on this activity, refer to Unit 1 Skill Builder 1 of the 10 minutes of code activities and Unit 4 Skill Builder 1 of the 10 minutes of code with TI-Innovator activities. • Start a new program • Attach Rover

Challenge 1: Use the rv.forward(distance, "unit") function to determine the rate that Rover drives forward, in meters per second.	**Teacher Guidance during Challenge 1:**
	• Students may have many ideas on how to determine this rate. The only important thing is for them to approximate this rate reliably so they can use it in future challenges.
	• The default speed of the Rover is 0.2 meters per second, which will be affected by the surface. If the rate is not 0.2 meters per second, some adjustment to the later challenges might be made based upon this activity.
	• It is important not to use the SPEED option in this challenge. Students will be introduced to that option in Challenge 3.
	• Have students drive Rover forward at least 1 meter. The default distance unit is 10 cm. Meters can be specified using the rv.forward(distance, "unit") form of the function available from the Rover>Drive>Drive with Options menu.
	• Example Program: **driverate.py**
	`import ti_rover as rv`
	`rv.forward(1,"m")`
	• *Discussion Starters*
	○ The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 1:
	▪ Write and interpret an equation for determining Rover's average drive rate.

Challenge 2: Use the rv.left() or rv.right() function to determine the rate that Rover turns, in degrees per second.

Teacher Guidance during Challenge 2:

- Students may have many ideas on how to determine this rate. The only important thing is for them to approximate this rate reliably so they can use it in future challenges.
- There is no SPEED option for the rv.left and rv.right functions.

- Example Program: **turnrate.py**

```
import ti_rover as rv
rv.right(360)
```

- *Discussion Starters*
  - The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 2:
    - Does it appear that Rover turns the same rate for small angles as for large angles? Explain how you know.
    - Write and interpret an equation for determining Rover's average turn rate.

Challenge 3: Have Rover drive 1 meter forward in less than 5 seconds. Use the rv.forward(distance,"unit",speed,"unit") form of the function.

Teacher Guidance during Challenge 3:

- There are forms of the forward() and backward() functions that accept different combinations of distance, TIME in seconds, and SPEED in m/s.
- In this challenge focus on students using the SPEED option. Find rv.forward(distance,"unit",speed,"unit") on the Rover>Drive>Drive with Options menu.
- The default speed is 0.20 m/s. Speed can be between 0.15 m/s and 0.23 m/s.
- Surface type will affect how fast Rover drives and turns. This may require an adjustment to the challenge.
- Example Program: **distcheck.py**

```
import ti_rover as rv
rv.forward(1,"m",.22,"m/s")
```

- *Challenge 3 Extension: Make Rover drive backward to the starting position at a different rate.*
  - Have your students edit the distance check program to drive backward to the original starting position at a different rate. They will need to use rv.backward(distance,"unit",speed,"unit") to drive in reverse to the starting position.

- *Discussion Starters*
  - The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 3:
    - How long did your Rover take to drive 1 meter? From that time, write and solve an equation to determine the actual rate your Rover was moving?
    - Try your program on a different surface and compare the times to drive 1 meter at the same rate. What effect does the type of surface have on the speed of Rover?

Challenge 4: Have Rover drive a rectangle using rv.forward() with distance and speed options, rv.left() and rv.right() functions that can be driven in exactly 10 seconds, ignoring the time it takes to turn.

Teacher Guidance during Challenge 4:

- Have students use the speed and distance options to control the lengths and times of each side. rv.left() and rv.right() accept angles from 0 to 360 degrees. The angle is measured from the current heading. This controls only the turn, in order for Rover to move in the new direction, a new rv.forward(distance,"unit",speed,”unit") or rv.backward(distance,"unit",speed,”unit") function will be needed.

- Example Program: **timerectangle.py**

```
import ti_rover as rv
rv.forward(.4,"m",.2,"m/s")
rv.right(90)
rv.forward(.6,"m",.2,"m/s")
rv.right(90)
rv.forward(.4,"m",.2,"m/s")
rv.right(90)
rv.forward(.6,"m",.2,"m/s")
rv.right(90)
```

- *Challenge 4 Extension: Make a triangle that takes 10 seconds to drive, ignoring the time it takes to turn*
  - Have your students write a new program to drive a triangle. Note: students may struggle with the angle measurements on what angle to turn to make a triangle, but it gives a nice discussion about the sum of the exterior angles of a polygon.

- *Discussion Starters*
  - The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 4:
    - How did you split up the time so that you had exactly 10 seconds?
    - Show the equations you solved to determine the time for each side.
    - What did you have to think about when determining the speed to drive each side of the rectangle?
    - How are the lengths of the sides of the rectangle and the speed related for this challenge?

Challenge 5: Have Rover drive a square using rv.forward_time() with time and speed options, rv.left and rv.right() functions. At least two sides of the square should be driven at a different rate than the others.

Teacher Guidance during Challenge 5:

- Have students only use the SPEED and TIME options to determine the length of each side. Note that TIME is only in seconds so there is no unit required. The rv.forward_time(time,speed,"unit") is found on the Rover>Drive>Drive with Options menu.

- Example Program: **speedtimesquare.py**

```
import ti_rover as rv
rv.forward_time(1,.2,"m/s")
rv.right(90)
rv.forward_time(.87,.23,"m/s")
rv.right(90)
rv.forward_time(1.33,.15,"m/s")
rv.right(90)
rv.forward_time(1,.2,"m/s")
rv.right(90)
```

- *Challenge 5 Extension: Drive one side of the square in reverse.*
  - This challenge will get students thinking about the external angles.

- *Discussion Starters*
  - The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 5:
    - How was this challenge different that challenge 4? How was it similar?
    - What is the relationship between speed and time that you had to use in order to keep the distance constant for each side of the square?
    - Write equations and solve for the length of each side of your square.
    - How long is each side of your square?
    - What impact would doubling the time for each side have on the perimeter of the square? How would driving each side for half the time impact the perimeter?
    - How does driving double the time impact the area of the square?

Challenge 6: Navigate "Math-hattan"! Have Rover navigate between the two locations that your team has been assigned. Be sure to follow the posted speed limits. Choose the path that will allow for the shortest time.

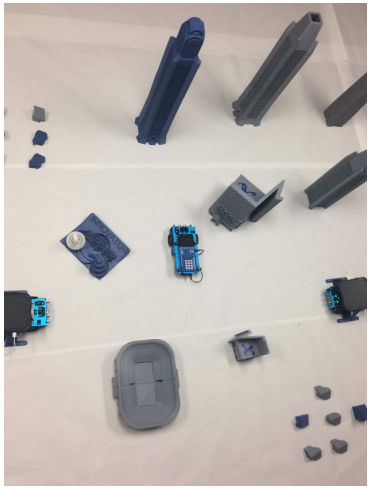


Image 1: City challenge suggested layout.

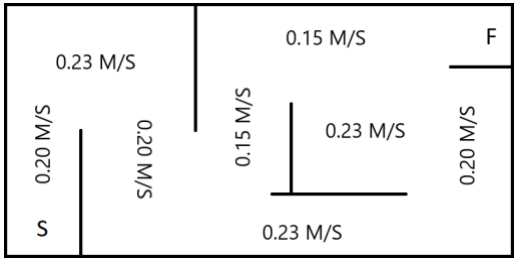


Image 2: Small practice maze suggested layout.

Teacher Guidance during Challenge 6:

- As students work on the route, remind them to consider the rate at which Rover turns in degrees per second. They will use that information to help them optimize their path through the city. There isn't a SPEED option for the turn so each group should have approximately the same answer for turn rate.
- Make sure students are using the SPEED option for each section where you've set up a speed limit.
- For the pictures, we've used 3D printed models; the files for these are posted with the activity. You can use your own buildings or just boxes or even layout the city with just masking or painters tape.
- Depending on your students, you may want to have them try a simpler maze similar to Image 2 first before navigating a larger city like in Image 1.
- Make sure students estimate the amount of time that their route will take before driving it.
- Give students the meter sticks and protractors to measure distance and angles for their route.
- Students will use the rv.forward_time() function available from the Rover> Drive> Drive with Options menu.

- Example program: **navigate.py**

```
import ti_rover as rv
rv.forward_time(3,.2,"m/s")
rv.left(90)
rv.forward_time(3.3,.15,"m/s")
rv.left(90)
rv.forward_time(3.5,.23,"m/s")
rv.right(90)
rv.forward_time(4,.23,"m/s")
```

- *Discussion Starters*
  - The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 6:
    - How did the rate at which Rover turns affect your planning of the route for fastest trip?
    - Explain how you used speed, distance, and number of turns to determine the fasted route.
    - Set up an equation that could be used to predict the amount of time it will take to drive your route. Drive your route and time it, how close was your prediction.