Move the Cone Challenge

TI-84 PLUS CE PYTHON AND THE TI-INNOVATOR™ ROVER

Overview:

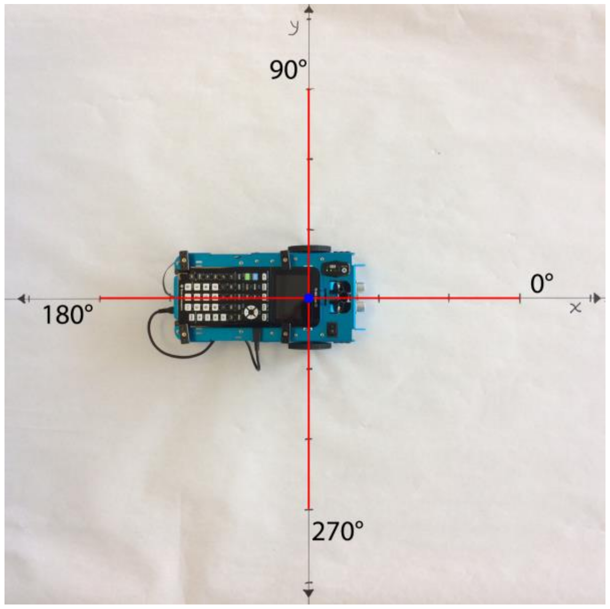
The goal of this activity is to have students exercise their knowledge of polygons, the coordinate plane, and equations of lines to write code to complete a set of challenges that introduce students to a set of basic commands to control the TI-Innovator Rover. Students will progress through a series of challenges to build skills, and then attempt a more complex final challenge. Note: These challenges can be addressed at several math levels, see the note at the end of the document for other grade level connections.

Goals:

Students will:

- create and edit TI-84 Plus CE Python programs on the calculator.
- write programs that include several commonly used Rover and calculator commands.
- use their knowledge of polygons, the coordinate plane, and equations of lines in the context of challenge problems.

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. 

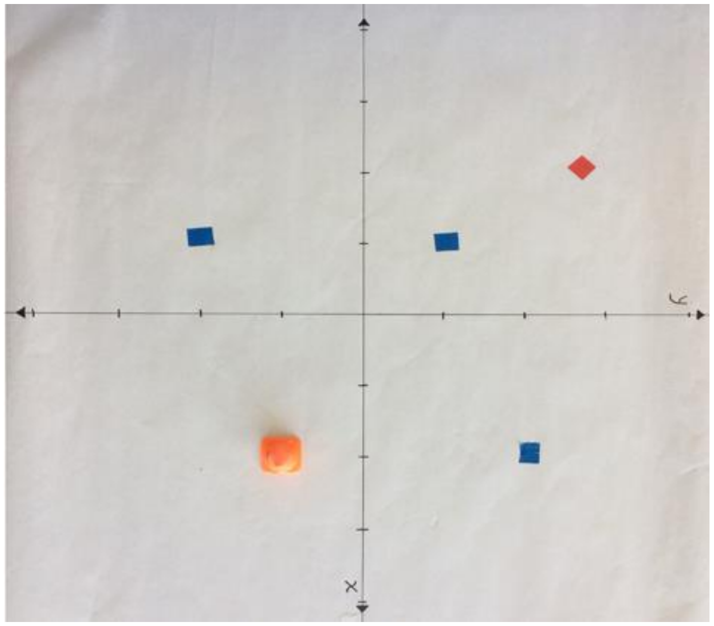
Python Syntax Reference (continued):

Statement	Example	Behavior
rv.forward(distance)	rv.forward(1)	Rover drives 1 unit forward. The default unit is 10 cm.
rv.backward(distance)	rv.backward(2)	Rover drives 2 units backward.
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.

See the Rover module section beginning on page 26 of the Python Programming for the TI-84 Plus CE Python Graphing Calculator Guidebook for more information.

Setup:

Students may work in groups of two or three. An example final challenge map is shown below.



Materials:

- TI-84 Plus CE graphing calculator with Move the Cone Student TI-84 Plus CE Python program loaded for use in the Python app.
- Calculator unit-to-unit cable (USB mini A to USB mini B cable)
- TI-Innovator Hub
- TI-Innovator Rover
- Challenge surfaces
  - 3'x3' Butcher paper
  - 3 colors of sticky dots
  - Axes labeled with 10 cm tickmarks
- Miniature traffic cones (or some other object to move)

Student Activity:

Sit in small groups with your calculator and supplies for this activity. Practice the guidance modeled by your teacher.

Teacher Notes:

Review and introduce the calculator, Hub, and Rover commands needed for this activity. In preparation for the coding on this activity, refer to Meet the Rover with Geometry Challenges (download the TI-84 Plus CE Python PDF), Unit 1 Getting Started with Python Skill Builder 1 of the 10 minutes of code activities and Unit 4 Rover's Driving Features Skill Builder 1 of the 10 minutes of code with TI-Innovator activities.

- Download the student and teacher files from the project web page.
- Use CE Connect to download the student Python files to the handhelds. The student files include the necessary module import statements and some additional scaffolding of the activity.
- Open the student Python program file for each challenge in the Python Editor. Have the students enter the necessary statements to drive the challenge.
- Attach Rover.

Challenge 1: Have Rover drive a square using the rv.forward(), rv.left() and rv.right() functions.

Teacher Guidance during Challenge 1:

- **import ti_rover as rv** sets the origin and heading of the rover. When this statement is executed, the current position of the rover is reset to (0,0) and the heading is toward the positive x-axis.
- rv.forward() accepts options of distance, time in seconds, and speed in m/s. In this activity focus on students using only distance in the default rover units (10 cm).
- 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() or rv.backward() statement will be needed.
- Note that program names must start with an alpha character and may not include spaces.

- Example Program: **MTC1T**

```
import ti_rover as rv
rv.forward(5)
rv.right(90)
rv.forward(5)
rv.right(90)
rv.forward(5)
rv.right(90)
rv.forward(5)
rv.right(90)
```

- *Challenge 1 Extension: Make a triangle*
    - 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 1:
        - What did you have to change to make the triangle?
        - What if we had decided to do an irregular triangle, what would be true about the angles turned?
        - What if you wanted to do an n-gon, what would have to be true about the angles turned?

Challenge 2: Have Rover drive a square using the rv.to_xy() function.

Teacher Guidance during Challenge 2:

- Rover tracks its current position, heading, and distance traveled in relation to the origin of its internal coordinate plane. The initial position and orientation are set when the **import ti_rover as rv** statement is executed.
- The rv.to_xy() function takes two values, one for the x-coordinate and one for the y-coordinate. Rover will turn and drive to the given point along a direct path.
- Example program: **MTC2T**

```
import ti_rover as rv
rv.to_xy(3,0)
rv.to_xy(3,3)
rv.to_xy(0,3)
rv.to_xy(0,0)
```

- *Challenge 2 Extension: Make a triangle*
    - Have your students write a new program to drive a triangle. Note: this challenge will be much less difficult using coordinates since the inputs are only 3 coordinates. To increase the rigor of this extension, challenge them to make a specific type of triangle (Right, Isosceles, Obtuse, etc.) or ask them to explore transformations of their triangle.

- *Discussion Starters*
  - The following are suggested discussion starters to engage your students with the mathematics inherent in Challenge 2:
    - How would the coordinates of a triangle in the first quadrant have to change so that rover drove the image of that triangle reflected over the y-axis?
    - Dilate your triangle about the origin by a factor 2. What are the new coordinates? Change your program, what do you think has happened to the area? Justify your responses.
    - For students familiar with trigonometry ask: What are the corresponding statements using rv.forward(), rv.left(), rv.right() that would make the same triangle?

Final Challenge: Have Rover circumnavigate the three blue dots, then push the traffic cone (or object) from the yellow dot to the red dot.

Teacher Guidance during Final Challenge:

- Students can use any commands they like in this challenge.
- Each team's challenge mat should be unique. You could even have students place the three blue dots, the yellow dot and the red dot to create their own challenge.
- Note students will need to account for the distance between the front of the rover and the reference point for rover. The reference point for rover is the center of the line between the axles of the motors. This makes for a nice conversation with students about translations. See background for orientation photo with the reference point of rover.

- Example program: **MTC3T**

```
import ti_rover as rv
rv.to_xy(3,3)
rv.to_xy(-3,3)
rv.to_xy(-2.5,-4)
rv.to_xy(3,-2)
rv.to_xy(-2,3)
```

- *Discussion Starters*
  - The following are suggested discussion starters and challenge extensions to engage your students with the mathematics inherent in the Final Challenge:
    - Have your students mark on the paper the point where they turned to start pushing the cone.
      - What do the last coordinate before you start pushing the cone, the initial position of the cone, and the final position of the cone have in common?
      - What is the equation of the line connecting the yellow dot (where the cone starts) and the red dot (where the cone stops)?
    - Challenge extension: make a second path that turns to push the cone further away or closer to the cone than their previous challenge solution.
      - What is true about the new "last point before pushing the cone" and the point from the previous solution?
    - Final challenge extension: push the cone to all three blue points prior to pushing to the red cone.
      - What new things to consider does this bring to the move the cone challenge?

Other Course Connections:

The above challenges can also be modified to address topics in more advanced math courses with small modifications to the prompts above:

- For Geometry students, switch the focus to coordinate geometry and trigonometric ratios.
  - In Challenge #1, have students change the triangle extension to have students drive a right triangle. Students will need to use special right triangles, trigonometric ratios, and/or Pythagorean Theorem to determine side and angle measures.
  - In Challenge #3, have students to complete the task using rv.forward() and rv.left() or rv.right() functions. This will force them to use trigonometry and the Pythagorean Theorem to compute the angles and distances to travel.
- For Precalculus students, vector addition, trigonometric ratios and polar coordinates can be the focus.
  - In Challenge #1, make the challenge about vectors that drive a square. Students will need the rv.to_angle() function.
  - In Challenge #2, have students use polar coordinates instead of Cartesian to draw the triangle and square. The rv.to_polar() function.
  - In Challenge #3, have students use polar coordinates, or vector addition to complete the task.

Command Support for other course connections:

Command	Example	Behavior
rv.to_angle (angle, “unit”)	rv.to_angle(270,”degrees”)	Rover rotates counter clockwise from its current heading to a heading of 270 degrees (parallel to the y-axis pointed toward negative infinity). This command takes a true angle measurement as set by the origin (see the picture in the description for **import ti_rover**)
rv.to_polar (angle, degrees)	rv.to_polar(5,30)	Sends rover to the coordinate point that is 5 units from the origin at an angle 30 degrees from the x-axis.