Skill Builder 3: Turtle Drive

UNIT 1: MOTION CONTROL
TI-84 PLUS CE PYTHON

THE ON-RAMP TO ROBOTICS

Overview:

Students write three programs to accomplish challenges requiring turtle drive. The challenges range from driving regular polygons to driving their Rovers around Olympus Mons on Mars.

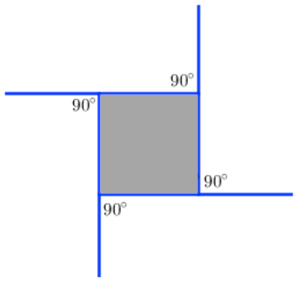
Goals:

Students will:

1. write a TI Python program that uses turtle drive to navigate a path.
2. discover basic properties of regular polygons.
3. explore the physical geography of Mars.

Background:

In computer science and computer graphics, Turtle commands are those where the present position, the “turtle”, is commanded to go to the next position by first turning to the appropriate direction and then moving forward a defined distance to reach the next point. In mathematics, this concept is known as a vector, something that has direction and size. In this activity, students will use their skills of turning and moving forward from the previous activities to accomplish three challenges. All three challenges are accomplished using turtle drive on the Rover. The first challenge is to drive a square with an edge length of .5 meters. This is accomplished by turning 90 degrees and then driving forward .5 meters. Repeat these steps three more times. Notice that the angle the Rover needs to turn is the EXTERIOR angle of the polygon. It is necessary to turn a total of 360° in order to be facing the original direction after the Rover finishes driving. This is a general statement for all regular polygons and leads to the formula for the exterior angles of a regular polygon of 360/n-sides. This formula is used in challenge 2 where students are challenged to make a regular polygon with the number of sides of their choosing (Note – Although there is no limit on the number of sides, advise students to keep it under 30 with a side length no smaller than 1 cm to ensure best results). In the final challenge, students are required to create a turtle drive path around a floor mat of Olympus Mons and avoid craters and boulders.



Exterior Angles of a Square add up to 360°

Olympus Mons on Mars is the tallest mountain in the solar system

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. The import ti_rover as rv statement is available from the Fns>Modul>ti_rover>Drive menu.
rv.forward(distance, “unit”)	`rv.forward(1.2, “m”)`	Rover drives forward 1.2 M at default speed of .20 M/S ** The rv.forward(distance,”unit”) function is found on the Rover Drive with Options menu. Unit options are “grid units” (10 centimeters is the default), “m” (meters) and “revs” (wheel revolutions). rv.forward(distance, “unit”) is found toward the bottom of the Fns>Modul>ti_rover>Drive menu.
rv.right (angle) rv.left (angle)	`rv.right(90)`	Rover spins to the right 90 degrees. Turn functions are found on the Rover drive menu: Fns>Modul>ti_rover>Drive menu

for i in range(size):
  block
	`for i in range(4):`
	`  rv.left(90)`
	`  rv.forward(.25,”m”)`
		Repeats the statements in the block four times. The index variable, i, starts at 0 and increases by 1 with each loop. If i is less than the size value, 4, the loop continues to repeat. The block starts with a colon and includes the indented lines that follow.
		
		The for loop statements are found on the Fns>Ctl menu.

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 programming information.

* The speed of the Rover will vary from the stated values depending on the floor surface. Some surfaces cause Rover to move more slowly. If accuracy is important, the speed should be measured by a method similar to the one in this activity
** The LEFT and RIGHT turns are made with a frame of reference from Rover’s driver’s seat.

Setup:

Students may work in groups of two or three. Choose an area to work that has at least 2 meters of clear uniform floor space. Carpeted flooring is less desirable than tile. If needed, driving mats may be used as a driving surface.

Materials:

- Dry erase marker and pen holder
- Drive mat or large hard, flat surface
- Large format print of Olympus Mons for drive mat overlay (optional but recommended)
- Various obstacles to negotiate on the driving mat
- TI-84 Plus CE graphing calculator with On-Ramp to Robotics Unit 1 Student TI-84 Plus CE Python program files 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.
- Student “Challenge” cards have been provided and can be printed/ cut into individual task cards, and distributed to students (optional)

Student Activity:	Teacher Activity:
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 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: Write a program that drives a square with a side length of .5M.	**Guidance during challenge 1:** Review the Rover commands needed for this activity. • Every time a program needs to talk to the Rover, use the `import ti_rover as rv` statement to bring in the Python TI Rover module capabilities. • When Rover needs to go forward some distance, use: `rv.forward(distance, "unit")` found toward the bottom of the Fns>Modul>ti_rover>Drive menu. • When Rover needs to turn a specific angle, use: `rv.right(angle_degrees)` found on the Rover Drive menu. • Review the for loop control statement found on the Fns>Ctl menu. • See Unit 1 Skill Builders 1 & 2 Example Programs • Give students the opportunity to explore making a square. Instead of giving them the turn angle, allow them to explore various angles to achieve their goal. • Allow them to choose to use a for loop or to explicitly list the many turtle drive functions one after the other. Foster a discussion with your students regarding which method is better or which is the most understandable. • Ask them to observe the beginning direction and ending direction of the Rover after they successfully drive a square. • Ask the students how many degrees Rover turned to finish facing in the same direction that it started. • For those who finish their squares quickly, challenge them to make an equilateral triangle. • Help them observe the exterior angles of their square and triangle. They can use the dry erase marker to draw on a board or large paper to illustrate the exterior angles.

Program: ORSB3C1T

```
import ti_rover as rv
for i in range(4):
  rv.right(90)
  rv.forward(.5,”m”)
```

Challenge 2: Write a program that drives a polygon with as many sides as you like, up to 30, with a side length of .3M.

Guidance during challenge 2:

- Help students to discover the formula for the exterior angles of a regular polygon, 360/n-sides.
- Point out the efficiency, compactness and readability of using a For loop in their programs.

Program: ORSB3C2T

```
import ti_rover as rv
for i in range(8):
  rv.right(45)
  rv.forward(.3,”m”)
```

Challenge 3: Write a program that navigates around Olympus Mons or class-created substitute without hitting any boulders.

Your team may use a meter stick and protractor to measure the course.

Guidance during challenge 3:
If you are able to print the large poster of Olympus Mons, please proceed to the files named, “On Ramp to Robotics Unit 1 Challenge Drive Around Olympus Mons”. Specifications to print this poster can be found in the file titled, “readme for printing drive mat”. There is also a student handout, titled, “On-Ramp to Robotics Unit 1 Challenge Drive Around Olympus Mons_Student”, which challenges students to make their measurements on a smaller scale and then do the math to scale them up for their program. A version of the file for teachers offers sample calculations and should be very close to what the students are aiming for. The file is titled, “On-Ramp to Robotics Unit 1 Challenge Drive Around Olympus Mons_Teacher”. This challenge can be modified to be used without the Olympus Mons poster. Details on this alternative approach are below.

In a large, open space (at least 1m x 1m), a challenging course should be designed using obstacles placed strategically to block an easy path around the center of the space. You could put a construction cone, stack of books, or other object(s) in the middle of the space to represent Olympus Mons, the Martian volcano that is the tallest known in the solar system. Once you establish an obstacle course around the “volcano”, allow students to use meter sticks and protractors to determine an efficient path. They will require these measurements to code their turtle drive program to negotiate the course. Students should follow the directions on the “On Ramp to Robotics Unit 1 Challenge Drive Around Olympus Mons_Student” regardless of using the printed Olympus Mons mat or a classroom-designed course.

Program: ORSB3C3T

```
import ti_rover as rv
rv.left(0)
rv.forward(.226,”m”)
rv.left(32)
rv.forward(.299,”m”)
rv.left(66)
rv.forward(.260,”m”)
rv.left(60)
rv.forward(.255,”m”)
rv.left(74)
rv.forward(.270,”m”)
rv.left(32)
rv.forward(.307,”m”)
```