Unit 6: Coordinates with Rover

Skill Builder 3: Make the Shape

In this lesson, you will write a program to make a pre-designed two-dimensional shape. You will program with lists and use a loop to get Rover to draw (or just drive) the shape on paper.

Objectives:

- Create lists in a Lists & Spreadsheet app
- Use **recall_list** in Python
- Use a **for** loop to process elements in a list
- Use a **pause** statement to control processing

This project requires you to make two lists that represent the x- and y-coordinates of the vertices of a shape of your own design. In this lesson, we will use the design of the block letter **T** shown to the right. Your goal is to have the Rover make this design using a marker (or just follow the route if no marker is available).

1. Before writing the program, enter your shape's coordinates into two lists in a TI-Nspire CX II's Lists & Spreadsheet app. All the coordinates of the 'T' shape are given in the image to the right. Note that the names of the two lists are **xs** and **ys**. These names are important for your program.

   You can test your values by setting up a Scatter Plot on a Graphs app or use the Data & Statistics app (Quick Graph from the Lists & Spreadsheet app).

Teacher Tip: If students are not familiar with the Lists & Spreadsheet app, you can make the two lists within the Python program:

```
xs = [ ... ]
ys = [ ... ]
```

but the method used in this lesson illustrates the ability to transfer variables between the Python environment and the TI-Nspire CX II environment.

2. Begin a new Python Rover Coding project.

   Your first two *new* statements will get the lists from the TI-Nspire variables and store them in two Python variables. It's okay to use the same variable names, but they are really not the same variables.

   From **menu > More modules > TI System**, use the statement:

   recall_list("name")

   You need *two* of these statements, so just select, copy, and paste the statement (or get it from the menu again).

```
1.4 1.5 1.6 *Unit 6 Py...rds RAD
*u6sb3.py 12/31
import ti_rover as rv
from math import *
import ti_plotlib as plt
from ti_system import *
from time import *
#================================
list=recall_list("name")

list=recall_list("name")
```

3. In one of these statements use **xs** for list and name and in the other use **ys.**

   These two statements get the lists **"xs"** and **"ys"** from the TI-Nspire CX II and store them in the Python variables **xs** and **ys**, respectively (to the left of the = sign).

```
1.4 1.5 1.6 Unit 6 Pyt...rds RAD
*u6sb3.py 1/22
# Unit 6 SB3 – Make the Shape
#================================
import ti_rover as rv
from math import *
import ti_plotlib as plt
from ti_system import *
from time import *
#================================
xs=recall_list("xs")

ys=recall_list("ys")
```

Teacher Tip: The Python variables and the TI-Nspire CX II variables do not have to have the same name, but this makes sense, doesn't it?
The statement **store_var(list, "name")** on the TI System menu does the reverse: it transfers values from a Python list to a TI-Nspire CX II variable "name". The tool tips are especially helpful in this statement as they clearly tell you which is which.

4. You are now ready to program the Rover's route. Remember that Rover begins at the point (0,0). Your first point might not be the origin, so have Rover move to the first point before pausing to insert the marker. The first point is (xs[0], ys[0]), so use the statement:

   rv.to_xy(xs[0], ys[0])

   [ ] are found on the left parenthesis key to the left of the 0 key.

```
1.4 1.5 1.6 Unit 6 Pyt...rds RAD
*u6sb3.py 2/22
#================================
import ti_rover as rv
from math import *
import ti_plotlib as plt
from ti_system import *
from time import *
#================================
xs=recall_list("xs")

ys=recall_list("ys")
rv.to_xy(xs[0], ys[0])
```

Teacher Tip: Both brackets are displayed when selecting them on the handheld, but you can delete either one if you only need one of them.
There are several ways in Python to iterate two lists. The one presented here is good for beginners.

5. Add a statement that pauses processing while you insert a marker (if you have one).

input("Press enter to continue.")

6. Make a **for** loop to drive to the rest of the points:

for i in range(1, len(xs)):
block

len(xs) is the length (size) of the list **xs**. If the length is 12, then the loop ends with the value i = 11, the last element of the list.

Teacher Tip: xs and **ys** must have the same size.

7. Complete the **block** of the loop using the statement:

rv.to_xy(xs[i], ys[i])

After the loop (*unindent*), pause again to remove the marker, and then send Rover back to the origin (0,0) facing east.

8. Challenge: Use the **ti_plotlib** tools to also plot the shape's coordinates on the screen in sync with Rover. Use **plt.auto_window(xs,ys)** to set up the window for your points. At the end, you can use the function **plt.plot(xs,ys,"mark")** to draw a connected scatter plot.