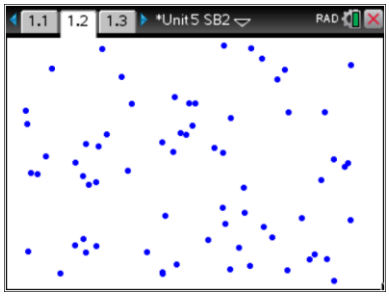
Unit 5: Lists, Graphics, and Dynamic Programs | **Skill Builder 1: Programming with Lists**

In this lesson, you will learn about using **Lists** in programs to produce interesting, interactive data graphs.

Objectives:

- Describe the basics of **lists** in programs
- Write programs that use **lists** to create scatter plots



TI-Basic does not provide direct graphical statements. A program can define functions for graphing and can create lists to produce scatterplots but cannot plot points directly.

In this unit, we will develop three big ideas:

1. working with lists in a program;
2. setting up a Graphs app to display a scatter plot of those lists; and
3. using 'dynamic' programs that can run 'on demand.'

Defining Lists

Lists are defined using the curly braces, { }. To create an empty list, use a statement such as **mylist:= { }** with nothing between the braces.

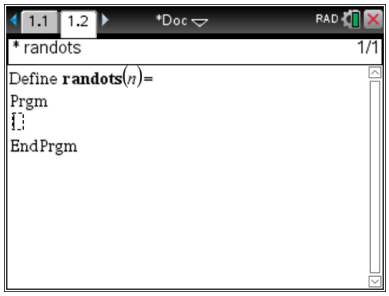
The list *elements* (values between the braces) are addressed using square braces after the list name, such as **mylist[3]**, which refers to the third element in **mylist**.

Add elements to a list by storing a value in the position immediately after the last value of the list. For example, if a list contains three elements such as {12, 7, 2} then you can add an element to this list by storing a value in list element number [4]. Entering **mylist[4]:= 17**, results in **mylist** = {12, 7, 2, 17}. Knowing how many elements are in a list is very helpful. **dim(***listname***)** tells how many elements are in the list (the dimension of the list). This command is often used to add an element to the end of a list. For example:

mylist [dim(mylist) + 1] := *<some value>*

Programming Random Dots

We'll begin by writing a program that will create a random dot pattern in a Graphs app.



1. Create a new program (we called it **randots**), and use the argument *n*.
   - *n* represents the number of dots the program will create.
   - The program will create (or revise) two 'global' lists of random numbers for use in a scatter plot in a Graphs app.

Random Number Generators

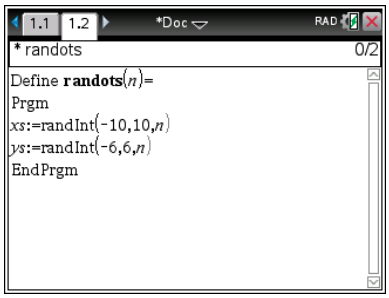
There are several random number functions in the TI-Nspire. The two most common are **rand( )** and **randInt( )**.

- **rand()** creates a random decimal between 0 and 1.
- **randInt(*a, b*)** creates a random integer between *a* and *b*, inclusive. Example: **randInt**(1,6) creates a random number from 1 through 6, inclusive. Try this command in a Calculator app.

©2018 Texas Instruments Incorporated

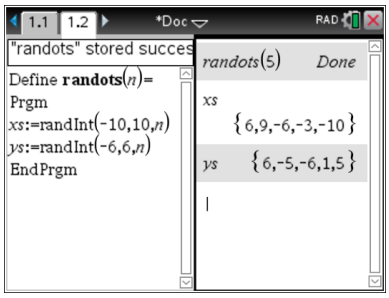
Our first lists of random numbers:

xs:= randInt(-10, 10, n)
ys:= randInt(-6, 6, n)



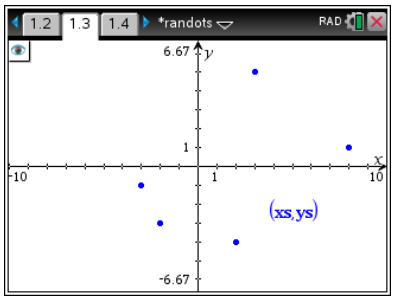
2. The third argument *n* in the two variable assignments above cause the **randInt( )** function to create a list of *n* random integers in the specified interval rather than a single random number.

3. Select **ctrl+R** to store and prepare to run the program. In the Calculator app, run the program with a small argument such as 5, and observe the values of *xs* and *ys*. Your values will probably differ from the ones to the right.



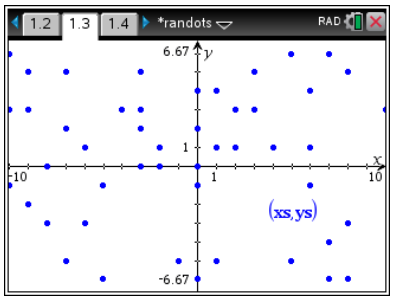
Note: To get a split screen with the Calculator app on the right from the Program Editor, select **doc > Page Layout > Select Layout,** and choose the vertical split icon (**Layout 2**). Add the Calculator app to the new window on the right.

4. Add a Graphs app to this problem, and set up a scatter plot of (xs,*ys*) by selecting **menu > Graph Entry/Edit > Scatterplot**.
5. Enter ***xs*** for the x← field, use the down arrow to move to the y← field, and enter ***ys***.
6. Press **enter** to plot the points.



Do you see why we chose those intervals for the random number functions? Study the Standard window settings using **menu > Window/Zoom > Window Settings**.

7. Now that we've tested the program, go back to the Calculator app, and rerun the program with a larger (but not too large) value for *n*.
8. Return to the graph to see the results.



randots(50)