# HP Prime Advanced Workshop

## Introduction

The HP Prime Advanced Workshop Series provide in-depth understanding of and experience with the advanced features of the HP Prime Graphing Calculator. This workshop focuses on the HP Prime Wireless Classroom Network, the new version 1.1 of the Geometry App, and programming the HP Prime using HP PPL (HP Prime Programming Language). In this workshop you will learn about custom HP Prime content, especially programs and customized HP Apps. As you learn about custom HP Prime content, you will also become familiar with the HP Prime Wireless Classroom Network, which will be used to distribute and share example content. The HP Classroom Network will also be used to survey the participants and share their thoughts, display a participant's Prime display for discussion purposes, and cover various other tasks.

## HP Prime Content

HP Prime was designed to be a customizable and extensible system for solving problems and learning mathematics. You customize and extend your HP Prime as you use it by saving data in variables, creating programs, and customizing the HP apps in the App Library.

### Programs

An HP Prime program is a set of instructions in HP PPL designed to complete a task. A program can be as simple as a user-defined function or much more complicated. Like notes, programs can stand alone in the Program Catalog or they can be attached to custom apps. Creating and sharing programs is one major focus of this workshop.

### Custom HP Apps

You customize any HP App in the Apps Library simply by using it, storing data in it, and then saving it under a new name. You can then attach a note to your new app, customizing it further. Finally, each app has a program attached to it. Initially, this program is empty. But you can use this program to extend your custom app even further. An app program can contain sub-routines; if it contains a sub-routine named "Plot" for example, then this sub-routine is called when the user presses Plot Setup while in the custom app. In fact, the app program can even take the user to another app and then return. Using programs to customize HP apps is the second major focus of this workshop.

### Notes

You can create as many notes as you like, subject to memory restrictions. You can keep your notes in the Note Catalog or you can attach them to custom apps as you create them. Notes have formatting capability to give them structure and make them easier to read.

# The Geometry App, Version 1.1

## Introduction

The Geometry app has a new version 1.1. Improvements of note to version 1.1 include tests on geometric objects, slider bars, animations, the ability to quickly select and move objects, and a whole set of plot types.

1. Press Apps Info and tap on the icon for the Geometry app

The app opens in Plot view. Press Plot Setup at any time to return to this view. There are shortcuts for basic objects and operations, as shown in the table below:

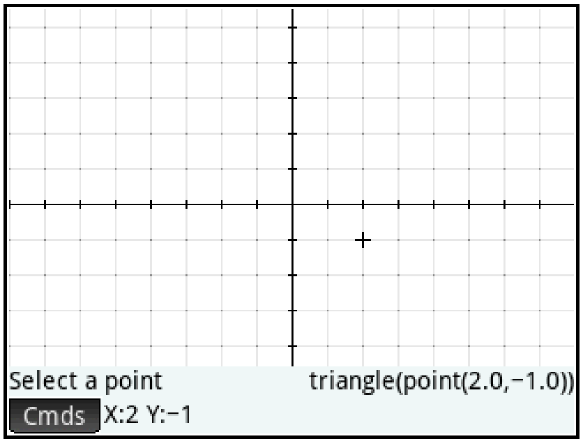
| Key | Tool |
|---|---|
| A | Toggle axes off and on |
| C | Circle |
| I | Find the intersection(s) of two objects |
| L | Line |
| P | Point |
| S | Segment |
| T | Triangle |

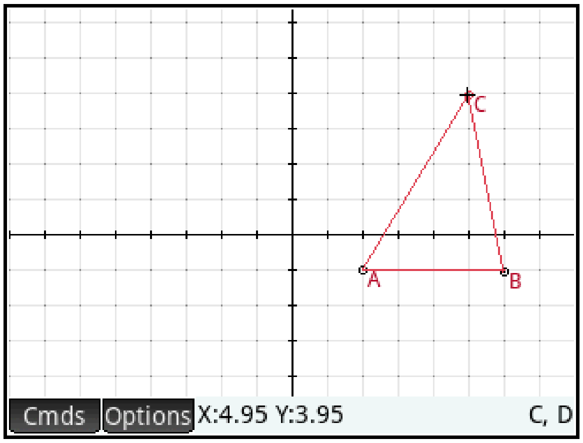
2. Press ÷, the shortcut for the Triangle command. The bottom of the display now has two lines: the top line shows help for the current command and the bottom line contains the Commands menu (Cmds) as well as the current cursor coordinates.
3. Press Enter to select the current cursor coordinates as the first vertex of the triangle.
4. Tap at another location and press Enter again.
5. Repeat Step 4 to create your triangle

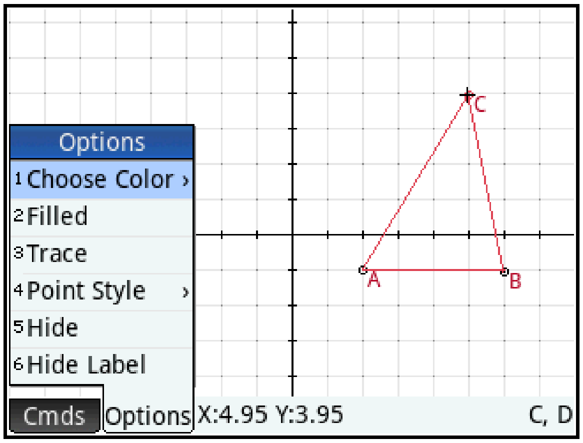
The Triangle command remains in effect. Press Esc Clear to clear it.

Note that the bottom right corner of the screen displays the names of all objects near the cursor; in this case, C and D. The cursor is near Point C and △ABC, which is the geometric object named D. These objects also are colored red to show they are currently selectable or moveable.

There is a new menu key: Options. This menu contains options for the selected object. In the figure to the right, `Hide label` was selected to hide the name of the triangle (D). Note the other options; these are available as well by pressing Menu Paste.


Select a point triangle(point(2.0,−1.0))



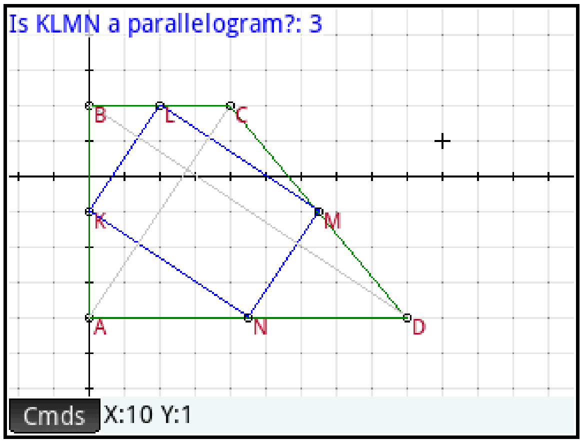


## Geometric Tests: The Midpoint Quadrilateral App

To get started and examine geometric tests, we will use the HP Prime Wireless Classroom Network to send a customized version of the Geometry app called the MidpointQuad app.

1. Press Apps and tap on the icon for the `MidpointQuad` app

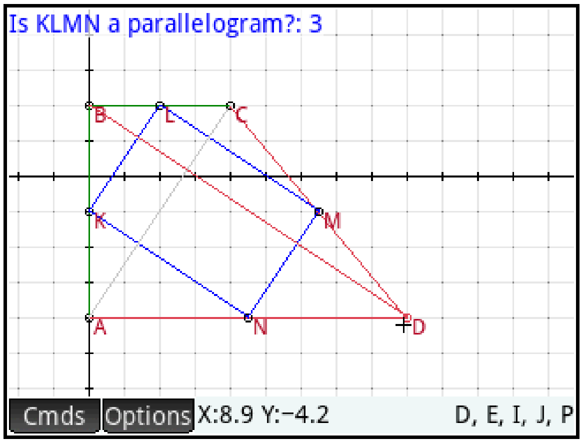
The app opens in Plot view. Press Plot at any time to return to this view. The green quadrilateral ABCD is the basis of the construction. The blue quadrilateral KLMN is formed by connecting the midpoints of the sides of ABCD. The diagonals AC and BD are also drawn to aid in conjecture and proof. At the top of the view is a test for whether or not KLMN is a parallelogram.


Is KLMN a parallelogram?: 3

### Two ways to move objects

2. Tap at point D to move the cursor there.

Once you see the cursor near point D, it will be shown among the list of objects near the cursor; this list is displayed at the bottom right of the screen, and the objects named are colored red for identification.


Is KLMN a parallelogram?: 3

3. Now tap at point D and drag it.

Point D will move according to the direction and length of your drag. This is a quick move.

4. Press Enter to select point D. It will turn light blue to show that it has been selected.

Now you can drag point D around OR use the arrow keys to move it one pixel at a time. Press Enter again to de-select point D.

Selected objects stay selected until you deselect them. You can select multiple objects as well, either individually or by tap-and-hold, then drag to create a selection rectangle. Everything touching the selection rectangle is selected. Press Esc to deselect everything.

### Geometric Tests

At the top of the screen is a parallelogram test for KLMN. This test consists of the test itself and its label. The label was edited to make it more readable for students. The Geometry app has a number of tests available to support student conjecture and proofs. These geometric tests can be created in either the Plot or Numeric views.

The parallelogram test can return any of the following five values:

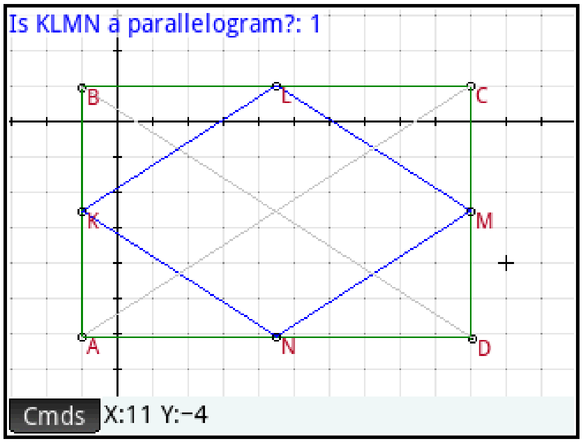
- 0: KLMN is not a parallelogram
- 1: KLMN is a parallelogram
- 2: KLMN is a rhombus
- 3: KLMN is a rectangle
- 4: KLMN is a square

Take a moment and experiment with the construction by moving points A, B, C, or D. You will see that the test never returns zero. Even if you make ABCD concave, the test will continue to return a value of at least one. This leads us to our first conjecture, usually known as Varignon's Theorem:

The quadrilateral formed by the midpoints of the sides of any quadrilateral is a parallelogram.

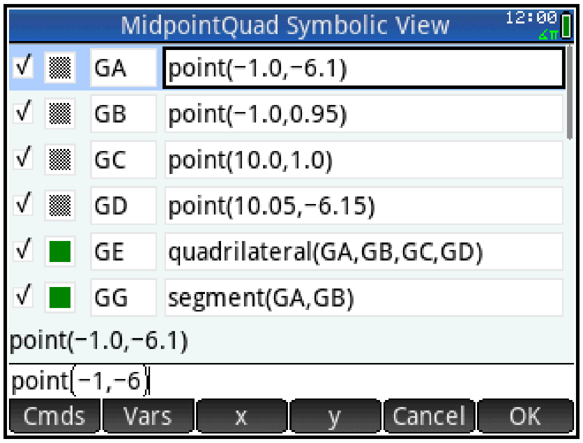
## The Technique

Let's make ABCD a rectangle.


Is KLMN a parallelogram?: 1

X:11 Y:−4

5. Move points A, B, C, and D so that they appear to form something close to a rectangle. In the figure to the right, the opposite sides of ABCD appear to be roughly equal in length and consecutive sides appear perpendicular.
6. Press Symb to open the Symbolic view of the app and observe the definitions of points A, B, C, and D.


MidpointQuad Symbolic View 12:00

| | | | |
|---|---|---|---|
| √ | ▩ | GA | point(−1.0,−6.1) |
| √ | ▩ | GB | point(−1.0,0.95) |
| √ | ▩ | GC | point(10.0,1.0) |
| √ | ▩ | GD | point(10.05,−6.15) |
| √ | 🟩 | GE | quadrilateral(GA,GB,GC,GD) |
| √ | 🟩 | GG | segment(GA,GB) |

point(−1.0,−6.1)

point(−1,−6)

We shall edit the coordinates of these points so that they are more exactly the coordinates of a parallelogram. Specifically, in our case (but not necessarily yours), we will make the following changes:

- GA: point(-1, -6)
- GB: point(-1, 1)
- GC: point(10, 1)
- GD: point(10, -6)

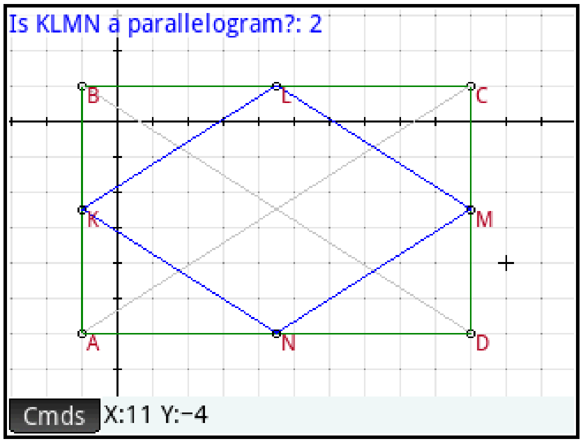
7. Tap on the definition of GA to select it.
8. Tap Edit to edit the definition
9. Delete the current coordinates and enter your own. Press Enter or tap OK when you are done.

In the figure to the right, the coordinates of point A are being changed to (-1, -6).

10. Repeat Steps 7-9 with points B, C, and D

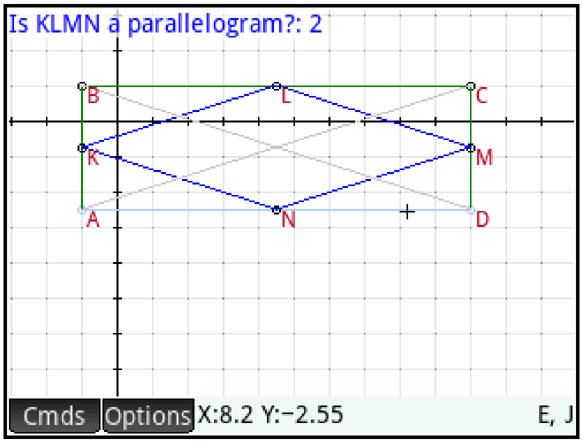
Return to Plot view after editing the coordinates of points A, B, C, and D.

11. Press Plot to return to Plot view to see what has changed.


Is KLMN a parallelogram?: 2

In this case, the test result has changed to 2, indicating that KLMN is a rhombus. Making ABCD a rectangle seems to make KLMN a rhombus. We will now change ABCD, but keep it a rectangle and see how the test behaves.

12. Tap on segment AD, between point N and point D, and press Enter to select it. A pop-up box will appear; make sure segment AD (object J) is selected and any others de-selected; then tap OK.
13. You can now use the up- or down-direction keys to move the segment, maintaining ABCD as a rectangle.

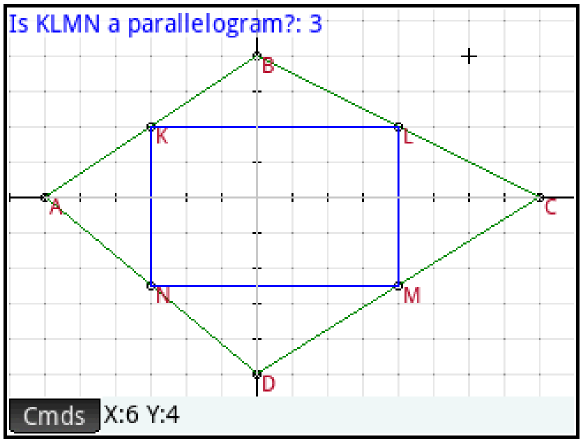

Is KLMN a parallelogram?: 2

You will note that the test continues to return a value of 2, indicating that if ABCD is a rectangle, then KLMN is a rhombus. This example was used to illustrate the technique involved:

- In Plot view, move A, B, C, or D to form what appears to be a special quadrilateral
- Go to Symbolic view and edit the definitions to make the coordinates exact
- Return to Plot view to see the effects on the test results
- Manipulate ABCD, keeping its properties intact and see if the test result changes

We can make ABCD a kite, a rhombus, a rectangle, a square, or a trapezoid. For each quadrilateral type, we can make a conjecture about KLMN.

## Try it yourself

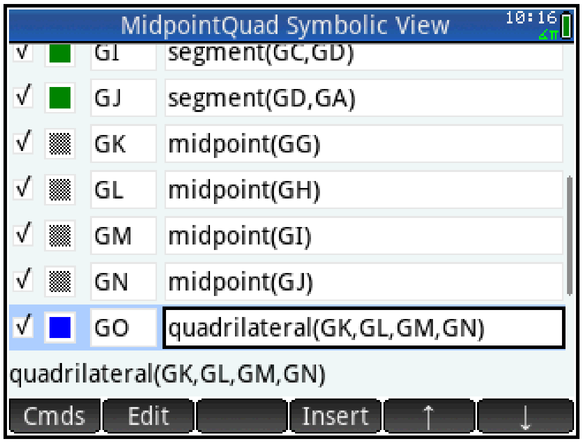

Is KLMN a parallelogram?: 3

Make ABCD a kite, following the procedure outlined above. Once it is a kite, select one of the vertices and press Enter to select it. Now you can move the vertex carefully using the directional pad, one pixel at a time. Move the vertex so that ABCD continues to be a kite. In the figure to the right, you could move A or C horizontally. Keep your eye on the test results as you move A or C.

## Creating a geometric test

We will now recreate the parallelogram test to illustrate the technique.

1. Press Symb to go to Symbolic view and identify the name of the quadrilateral KLMN. The figure to the right shows that KLMN is the geometric object named GO.


MidpointQuad Symbolic View 10:16

| | | | |
|---|---|---|---|
| ✓ | 🟩 | GI | segment(GC,GD) |
| ✓ | 🟩 | GJ | segment(GD,GA) |
| ✓ | ▨ | GK | midpoint(GG) |
| ✓ | ▨ | GL | midpoint(GH) |
| ✓ | ▨ | GM | midpoint(GI) |
| ✓ | ▨ | GN | midpoint(GJ) |
| ✓ | 🟦 | GO | quadrilateral(GK,GL,GM,GN) |

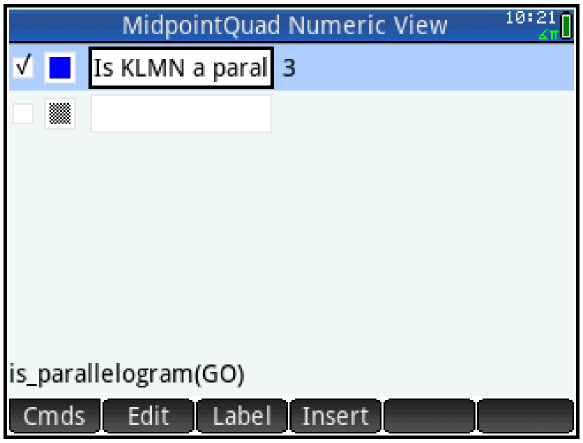
quadrilateral(GK,GL,GM,GN)

The Symbolic view shows the definition of every geometric object in Plot view, whether visible or not. Those that are checked are visible; those that are not checked are hidden. You can easily see how any construction was created and the order in which the objects were created. You can also choose the color of each object in Symbolic view.

2. Press Num to go to Numeric view. The first line contains our current parallelogram test. Select this entry to see the definition of the test at the bottom of the screen:

   is_parallelogram(GO)

Tap Edit to edit the definition or Label to edit the label.

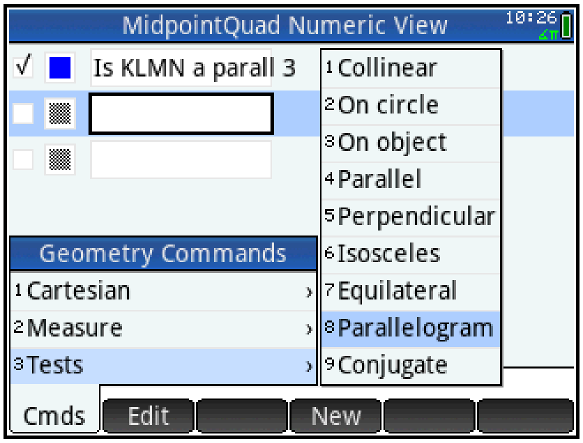

MidpointQuad Numeric View 10:21

| | | | |
|---|---|---|---|
| ✓ | 🟦 | Is KLMN a paral | 3 |
| ☐ | ▨ | | |

is_parallelogram(GO)

We will now recreate the test.

3. Tap on the second entry line to select it and tap New
4. Tap Cmds, select `Tests`, and choose `Parallelogram`
5. The command is pasted to the command line. Tap Vars, scroll down and select `GO` (or type the name using the Alpha shift).
6. Tap OK to enter the new test.
7. Tap Label to edit the test label. Enter a more descriptive text label. We used "Is KLMN a parallelogram?".
8. Tap OK when your label is done.


MidpointQuad Numeric View 10:26

| | | | |
|---|---|---|---|
| ✓ | 🟦 | Is KLMN a parall | 3 |
| ☐ | ▨ | | |
| ☐ | ▨ | | |

Objects in Numeric view that are checked appear docked to the top left in Plot view; objects that are not checked in Numeric view are not visible in Plot view.

Return to Plot view.

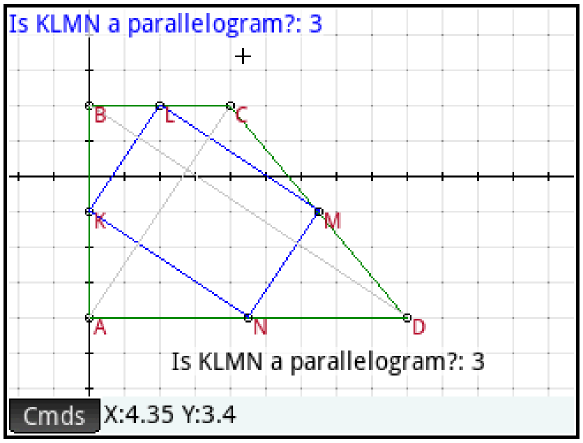
9. Press Plot to see our duplicate test in Plot view.
10. Tap on the duplicate. Now drag it to the bottom of the screen, as shown in the figure to the right. We now have the original test docked at the top left and our duplicate undocked at the bottom of the screen.
11. Pinch to zoom out. You will see that the undocked test changes both size and position with the zoom, while the docked test remains the same.
12. Tap and hold on the undocked test. The menu keys change and the label edit box opens.

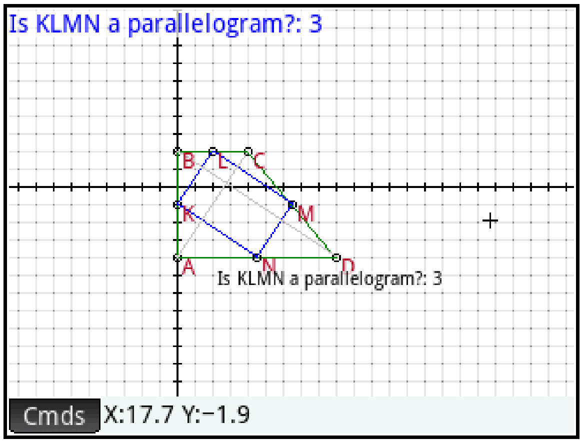
The cursor appears at the end of the current label. Delete or edit the text as you see fit.

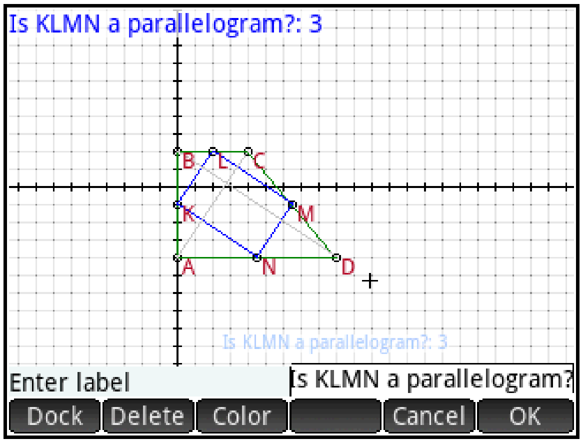
The table below lists the other operations you can perform with the new menu keys.

| Menu Key | Purpose |
|---|---|
| Dock | Re-dock the measurement or test |
| Delete | Delete the measurement or test |
| Color | Select a color for the test and its label |
| Cancel | Cancel label editing and return to the main menu |
| OK | Accept label editing and return to the main menu |

Since we do not need it anymore, tap Delete to delete the duplicate test.







The Geometry app has an entire set of tests like the parallelogram test. Tap Cmds and select `Tests` to see the whole list. There are an additional 7 tests in the Catalog.

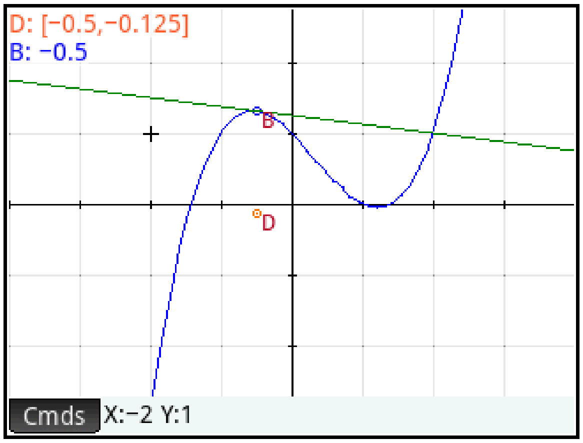
Testing geometric objects for their properties allows students to make conjectures and pursue proofs. Having dedicated Numeric and Symbolic views helps students and their teachers see how the app was created and how it works.

## Elements, animations and function plots: the SlopeFunction App

We will now use the HP Prime Wireless Classroom Network to send you an app called SlopeFunction. We will use this app to investigate the element command and creating animations. Along the way, we will discover one of many plot types.

1. Press Apps and tap on the icon for the `SlopeFunction` app

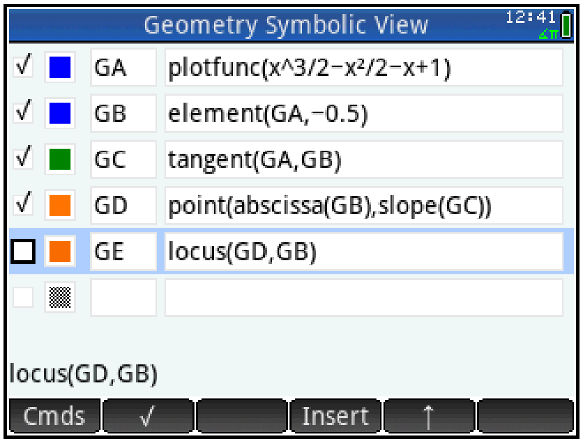
The app opens in Plot view. Press Plot at any time to return to this view. The plot of a cubic function is shown in blue, with a green tangent line through a point B on the function. There is an orange point D as well. At the top of the display are the coordinates of point D as well as a measurement named B that shows a current value of -0.5.



D: [−0.5,−0.125]
B: −0.5

2. Press Symb to enter Symbolic view. There are 5 objects defined here:
   - GA: the plot of a cubic function
     - Cmds, `Plot`, `Function`
   - GB: a point on the function
     - Cmds, `Point`, `Point On`
   - GC: the tangent to the function through the point B
     - Cmds, `Line`, `Tangent`
   - GD: a point whose abscissa is the same as the abscissa of point B and whose ordinate is the same as the slope of the tangent line
     - Cmds, `Point`, `Point`
   - GE: the locus of point D as point GB moves along the function GA
     - Cmds, `Curve`, `Locus`

The Plot view only showed us 5 of the 6 objects. Note that GE is not checked in Symbolic view, so it is not visible in Plot view.



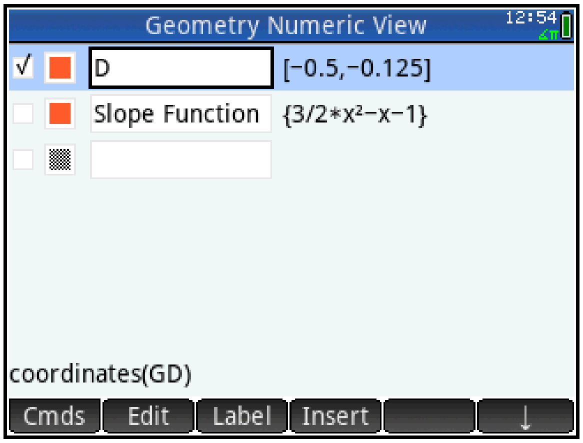
Geometry Symbolic View 12:41

| | | | |
|---|---|---|---|
| √ | 🟦 | GA | plotfunc(x^3/2−x²/2−x+1) |
| √ | 🟦 | GB | element(GA,−0.5) |
| √ | 🟩 | GC | tangent(GA,GB) |
| √ | 🟧 | GD | point(abscissa(GB),slope(GC)) |
| □ | 🟧 | GE | locus(GD,GB) |
| | ▨ | | |

locus(GD,GB)

3. Press Num to enter Numeric view. There are two measurements defined here:
   - The coordinates of point D
     - Cmds, `Cartesian`, `Coordinates`
   - The equation of the locus GE
     - Cmds, `Cartesian`, `Equation of`

Only the coordinates of point D are checked to be visible in Plot view. The equation of the slope function is hidden.

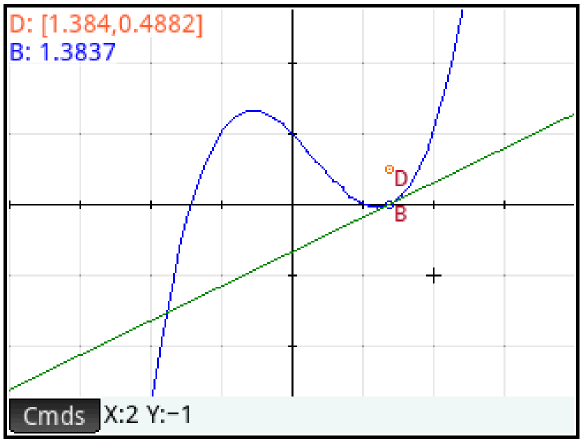


Geometry Numeric View 12:54

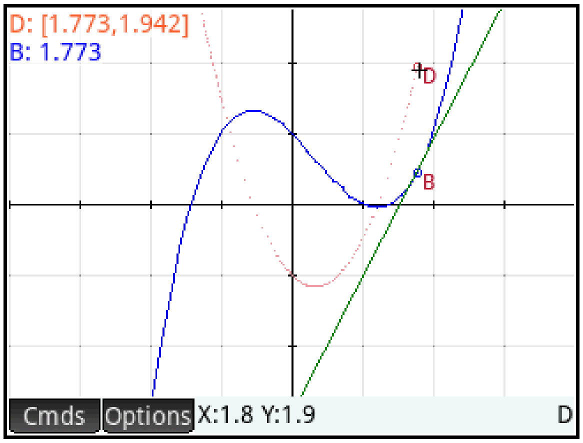
| | | | |
|---|---|---|---|
| √ | 🟧 | D | [−0.5,−0.125] |
| | 🟧 | Slope Function | {3/2∗x²−x−1} |
| | ▨ | | |

coordinates(GD)

The definition of point D places it on the slope function for the function defined in GA. Also, since its x-coordinate is the same as that of point B, you can control point D by moving point B.



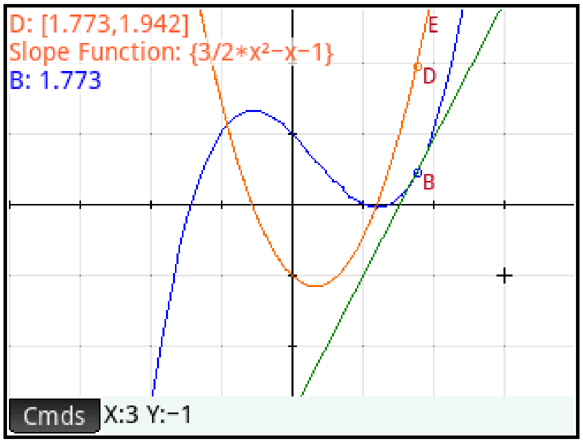
4. Tap on point B and move it anywhere along the function plot. You will see point D move and the coordinates of point D will update accordingly.
5. Tap on point D. The menu key Options appears. Tap on it and select `Trace`. Now move point B again. As point D moves, it traces its path.



The path of point D appears to be parabolic. We can now make the locus and its equation visible.

6. Press Symb to return to Symbolic view and check GE, the locus, so it is visible in Plot view.
7. Press Num to return to Numeric view and check the equation so it, too, is visible in Plot view.
8. Press Plot to return to Plot view to see our new additions.

The locus is shown in orange and its formula $\frac{3}{2}x^2 - x - 1$ is displayed as well.



You can now edit the function GA in Symbolic view and the Plot view will change accordingly. In other words, you now have a way to visualize the slope function of any function, either one point at a time, or via its trace or locus. The choice is yours.

## Elements and Animations

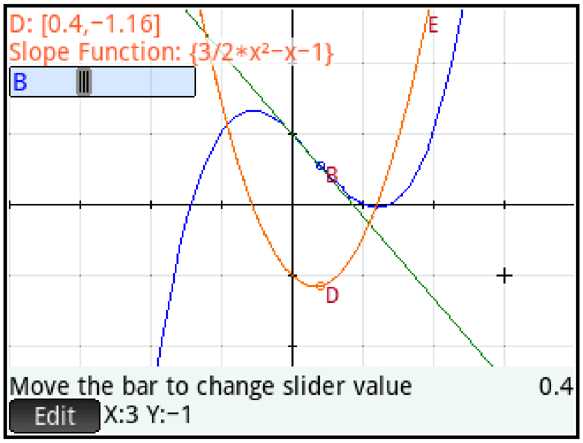
A word about the elements command is in order. In Symbolic view, the definition of GB is:

GB:=element(GA, -0.5)

This amounts to saying that point B is a point on GA (the function plot) whose initial x-coordinate is -0.5. When you create an element (a point on an object), the x-coordinate is shown in Plot view as a measurement. This measurement is actually a slider bar, and the slider bar can be used to control an animation.

9. Tap and hold the measurement labeled B. A slider bar appears. You can now move the slider left and right, either with your finger or with the left- and right-direction keys.

Notice the new menu key: Edit. Press this menu key to define the slider bar and any animation associated with it precisely.



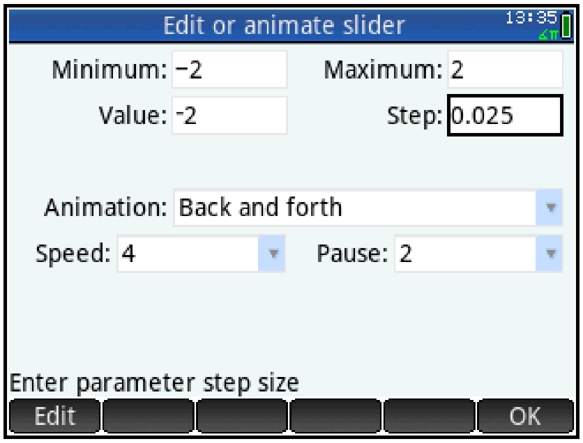
D: [0.4,−1.16]

Slope Function: $\{3/2*x^2-x-1\}$

B

Move the bar to change slider value 0.4

10. Press Edit. In the figure to the right, the minimum and maximum values have been set to correspond roughly with the Plot view visible domain of GA, with a step of 0.025 (the current pixel width in Plot view).
11. Tap on the `Animation` field. You can choose either a `Loop` or a `Back and Forth` animation. Select the `Back and forth` option.
12. Tap OK to create the animation.



Edit or animate slider

Minimum: −2 Maximum: 2

Value: -2 Step: 0.025

Animation: Back and forth

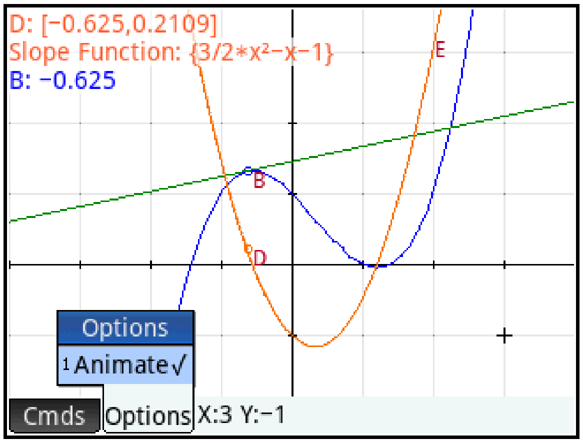
Speed: 4 Pause: 2

Enter parameter step size

Note that you can control the relative speed of an animation as well as the length of the pause before the animation repeats. Here we kept the default values.

Since the slider was selected when we edited it, the slider is still selected when we tap OK.

13. Press Esc Clear to deselect the slider bar and the animation will start.
14. In general, you start, stop, and restart an animation by pressing Options and tapping `Animation`. It is a toggle. A check mark will show that it is active and will be cleared when it is inactive.



D: [−0.625,0.2109]

Slope Function: $\{3/2*x^2-x-1\}$

B: −0.625

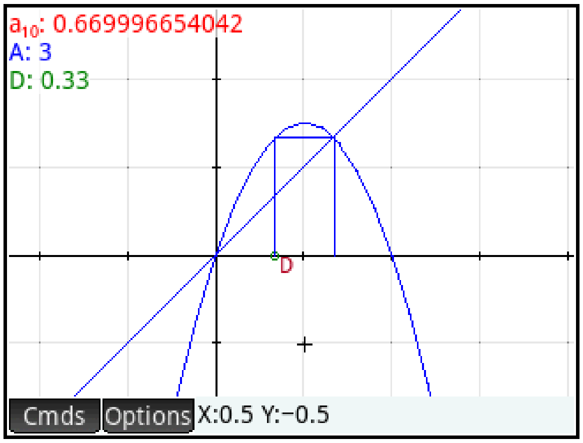
With slider bars and animations, you can create any number of interesting simulations and explorations. The next app we study will feature another use of slider bars as well.

## More on slider bars and plot types: the Iterations App

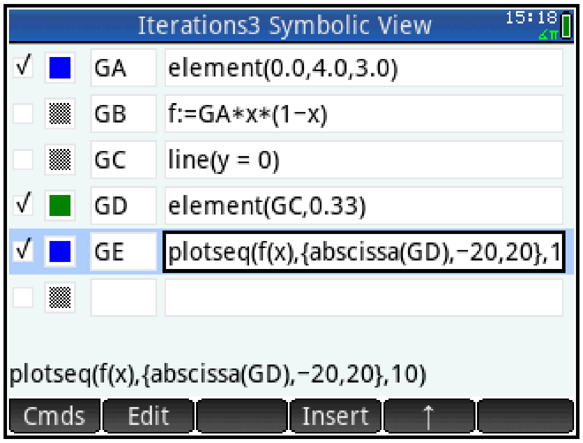
We will now use the HP Prime Wireless Classroom Network to send you an app called Iterations. We will use this app to learn more about the element command. Along the way, we will discover one more plot type.

1. Press Apps and tap on the icon for the `Iterations` app

The app opens in Plot view. Press Plot at any time to return to this view. The plot of a quadratic function and the line y=x are shown in blue, with a green point D as well. At the top of the display are three measurements or calculations, two of which are slider bars.



2. Press Symb to enter Symbolic view. There are 5 objects defined here:
   - GA: a slider bar for a parameter value
     - Cmds, `Curve`, `Slider`
   - GB: a variable named f that contains a quadratic function defined in terms of the slider GA
     - Just type it in
   - GC: a line defined by y=0
     - Cmds, `Line`, `Line`
   - GD: a point on the line y=0
     - Cmds, `Point`, `Point On`
   - GE: a sequence plot of GB, using the abscissa of point D as the initial value
     - Cmds, `Plot`, `Sequence`



| | | | |
|---|---|---|---|
| √ | 🟦 | GA | element(0.0,4.0,3.0) |
| | ▨ | GB | f:=GA*x*(1−x) |
| | ▨ | GC | line(y = 0) |
| √ | 🟩 | GD | element(GC,0.33) |
| √ | 🟦 | GE | plotseq(f(x),{abscissa(GD),−20,20},1 |
| | ▨ | | |

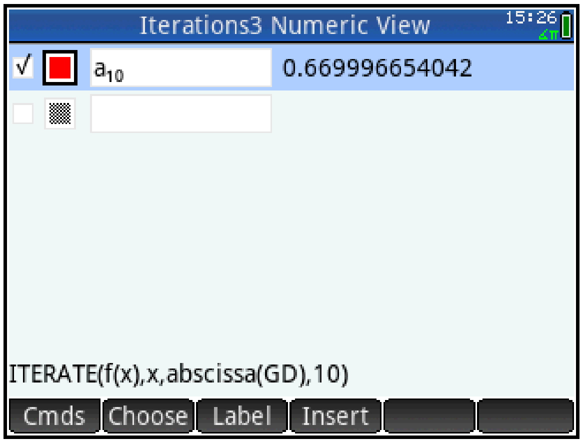
plotseq(f(x),{abscissa(GD),−20,20},10)

The plotseq command has the following syntax:

Plotseq(function in x, {starting value of x, xmin, xmax}, number of iterations)

The Symbolic view only showed us 5 of the 6 objects.

3. Press Num to enter Numeric view. There is one calculation defined here:
   - The 10th iteration on the quadratic function, using the abscissa of point D as initial value
     - Mem, Catlg, `ITERATE`



| | | | |
|---|---|---|---|
| √ | 🟥 | $a_{10}$ | 0.669996654042 |
| | ▨ | | |

ITERATE(f(x),x,abscissa(GD),10)

The ITERATE command has the following syntax:

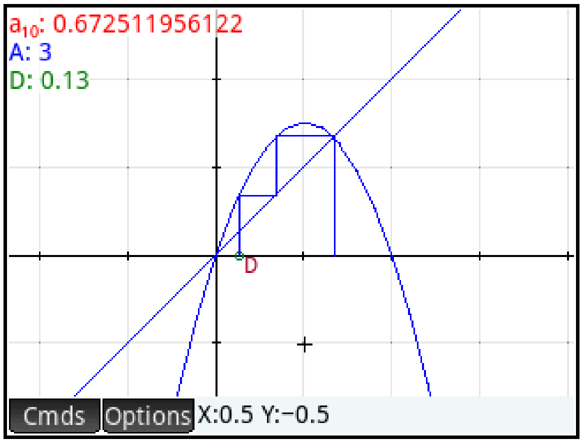
ITERATE(function, variable, starting value, number of iterations)

Note that GA uses the same element command that we used for a point on an object, but with a different syntax. The command element(object, value) creates a point on an object at an initial value as well as a slider bar to change that value. The command element(min, max, initial) create a slider bar with a given domain and a given initial value, without creating a point on an object.

4. Press P to return to Plot view. Tap and drag point D along the x-axis. The 10th iteration will change along with the value shown.

As you know from the previous app, you can tap and hold on the slider bar for D to open it and control the position of point D that way as well.

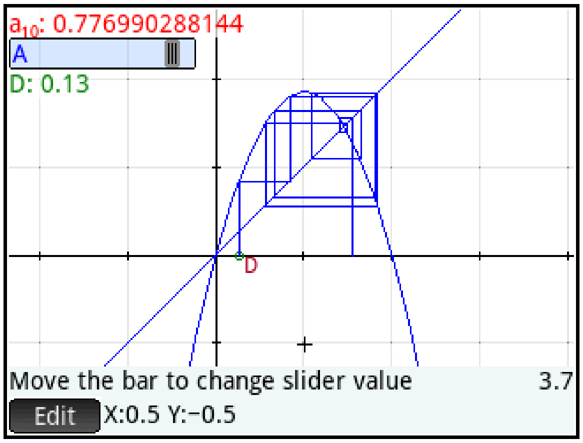
5. Now tap and hold on A to open its slider bar
6. Use the left- and right-direction keys to change the parameter value and observe its effect on the sequence plot.


$a_{10}$: 0.672511956122
A: 3
D: 0.13

X:0.5 Y:−0.5

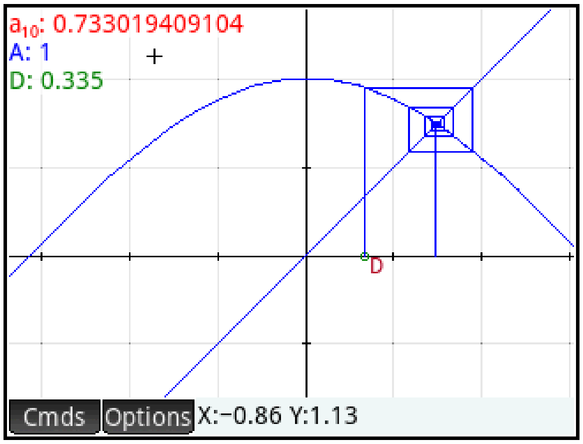
The value of the slider is shown at the bottom right of the screen.

Of course, you can create an animation based on the value of A, just as we did for the point D in the previous app. You can also create an animation for point D the same way. You can even run both animations at the same time; though in this case, the pedagogical value is dubious.


$a_{10}$: 0.776990288144

D: 0.13
Move the bar to change slider value 3.7
X:0.5 Y:−0.5

Edit the definition of the function f(x) in GB and you can iterate on any function you please. In the figure to the right, we edited GB so that f(x):=GA*cos(x).


$a_{10}$: 0.733019409104
A: 1
D: 0.335

X:−0.86 Y:1.13

With slider bars via the element command, and a whole set of plot types, the Geometry app is really about much more than just Geometry.

# Programming in HP PPL

The HP Prime Programming Language (HP PPL) was designed for general use. It is simple to get started with, but powerful enough for a wide range of applications. HP PPL supports basic programming structures, such as loops and branches of various sorts. It has an extensive set of graphical, input/output, string, matrix and other types of commands. Within any program, you also have access to all the mathematical functions of your HP Prime. Finally, you have access to any App Function from within a program.

## Programming Options

You can create an HP PPL program using a number of methods, as shown in the table below. Each method has its benefits and drawbacks. No matter which method you use, the HP Prime Connectivity Kit assures that you can send your program to your HP Prime and your HP Prime Virtual Calculator.

| Method | Benefits | Drawbacks |
|---|---|---|
| HP Prime | • Mobility: work on your program anytime, any place<br>• Debug: use the built-in check and debugger tools to check your program | • Must use menus to find and enter commands-or use the alpha shift to enter a command letter by letter |
| HP Prime Virtual Calculator | • Fast: use your PC keyboard to just type in your program<br>• Debug: use the built-in check and debugger tools to check your program | • Must have access to your PC |
| HP Prime Connectivity Kit | • Fast: use your PC keyboard to just type in your program<br>• Commands, strings, and comments are color-coded for easy identification and reading | • No debugging tools<br>• Must have access to your PC |
| Your Favorite Word Processor | • Fast: use your PC keyboard to just type in your program | • Must copy and paste the program into the virtual calculator or connectivity kit<br>• No debugging tools<br>• Must have access to your PC |

# Getting started: user-defined functions

The simplest way of extending the capabilities of your HP Prime via programming is to create user-defined functions. This can be done using two different methods, each with its own variiations. In the following examples, we will explore creating, modifying, and deleting user-defined functions.

In the setting of a geometric random variable, each trial is independent and has probability of success *p*; the probability density function for the distribution of such a variable computes the probability of the first success occurring on the *k*th trial. The formula for the geometric probability density function is thus $P(X=k)=(1-p)^{k-1}\cdot p$. HP Prime has no built-in geometric probability density or cumulative density functions. We shall create a user-defined geometric probability density function in two variables.

## User-defined functions: geometric_pdf(p, k)

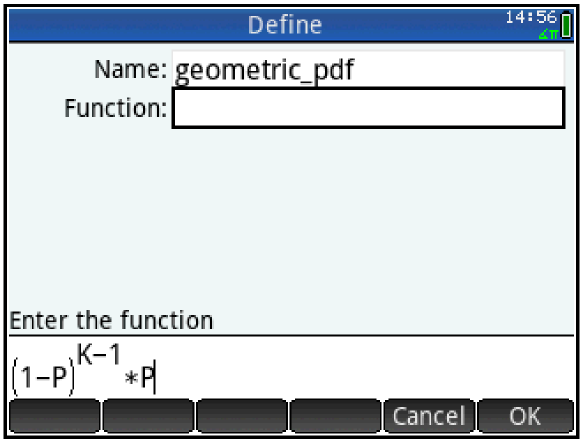
In this example, we use the Define dialog box to create a function named geometric_pdf.

1. Press Settings to go to the Home view
2. Press Shift xtθn Define (Define) to open the *Define* dialog box
3. The first field in the *Define* dialog box is for the name of the function:
   - Press ALPHA Shift ALPHA to lock the lower-case alpha shift
   - Enter "geometric_pdf" and tap OK

Note: press _ to get the underscore character

4. Highlight the *Function* field and enter the formula for the geometric probability density function
   - Enter the formula as shown to the right
   - Tap OK when you are finished

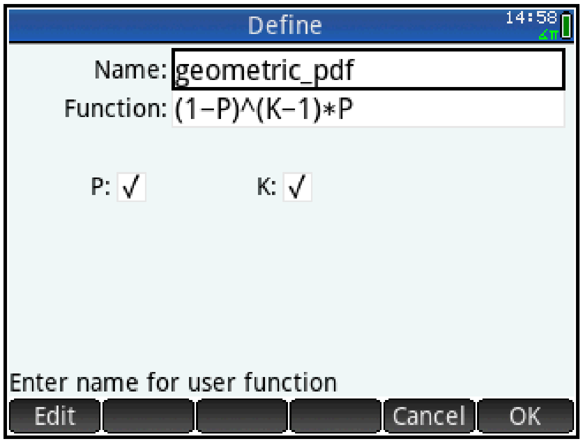
   Note: We use upper-case letters for our variables here



Name: geometric_pdf
Function:

Enter the function

$(1-P)^{K-1}*P$



Name: geometric_pdf
Function: (1−P)^(K−1)*P

P: ✓ K: ✓

Enter name for user function

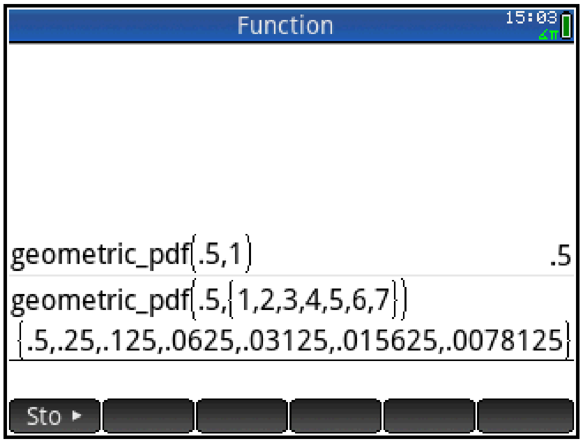
You will now see the two variables (P and K), with check boxes beside each one. The left-to-right order of the variables matches their order in the function; that is our geometric probability density function is actually geometric_pdf(P, K).

The check box beside each variable determines whether or not the variable is an input variable. If checked, a variable is an input variable and a value for it must be supplied each time the function is used. If unchecked, the variable is not an input variable; instead of being required as input when the function is used, the current value of the corresponding home real number variable will be used.

If a coin is has equal probabilities for heads and tails, then the probability of heads on the first toss is 0.5. That is geometric_pdf(0.5, 1) should return 0.5.

1. To access our new function, press Mem to open the Toolbox menus, tap USER for the *User* menu, tap *User Functions*, and select *geometric_pdf*
2. Complete the command geometric_pdf(0.5, 1) and press Enter

The result is as expected. The screen shot also shows what happens when you use a list for K.

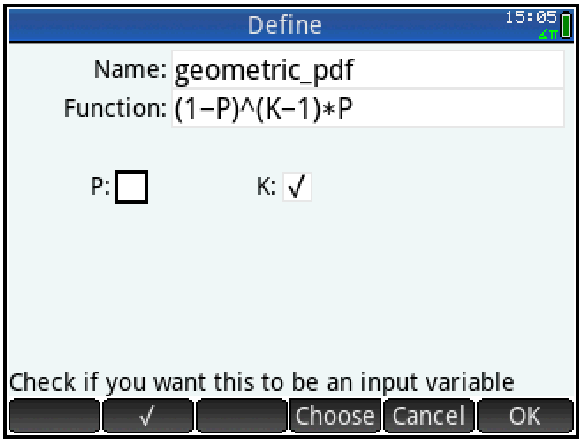


Function 15:03

```
geometric_pdf(.5,1)                              .5
geometric_pdf(.5,{1,2,3,4,5,6,7})
{.5,.25,.125,.0625,.03125,.015625,.0078125}
```

Suppose that, in practice, we find that we rarely change the value of *P* and do many calculations all using the same *P*-value. We can edit the definition of our function so that it takes its *P*-value from the current value of the home variable P instead of supplying that value each time we use the function.

1. Press Shift Define (Define) to open the *Define* dialog box
2. With the Name field highlighted, tap Choose and select *geometrix_pdf*; the current definition populates the *Define* dialog box
3. Highlight the field for the variable P, tap √ to uncheck it and tap OK
4. In the Home view, store the value 0.5 in P:
   - Type 0.5, tap Sto ▸, press ALPHA EEX (for P) and press Enter



Define 15:05

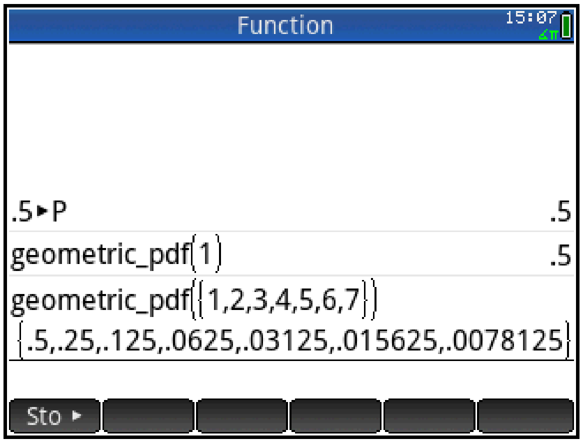
Name: geometric_pdf

Function: (1−P)^(K−1)*P

P: □ K: √

Check if you want this to be an input variable

Now our function is geometric_pdf( K) and takes its probability from the value of the Home real variable P. The figure to the right reprises the earlier examples.



Function 15:07

```
.5▸P                                             .5
geometric_pdf(1)                                 .5
geometric_pdf({1,2,3,4,5,6,7})
{.5,.25,.125,.0625,.03125,.015625,.0078125}
```

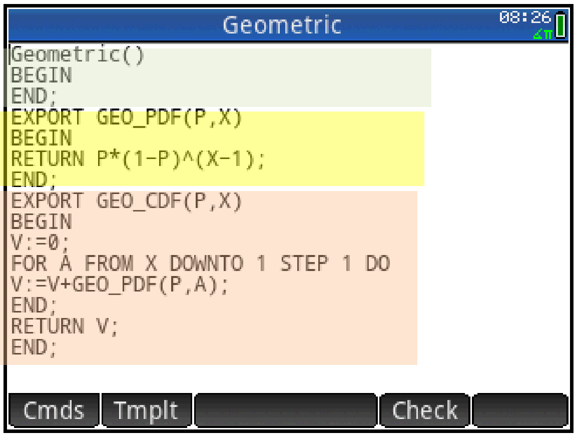
To delete a user-defined function, press Mem to open the Toolbox menus, tap USER for the *User* menu, and use the cursor keys to navigate to the function you want to delete. With the function highlighted, press Del.

In the next section, we will continue to examine user-defined functions, but this time we will use the Program Catalog and Program Editor.

## Exported functions in a program: Geometric probability functions

The geometric probability cumulative density function calculates the probability that a success is generated by the $K^{th}$ trial, where each trial is independent and has an equal probability of success *P*. It is thus a sum whose terms are each given by our geometric_pdf function. In this example, we will use the network to send you a program named **Geometric** that contains both the probability density and cumulative density functions.

1. Press Shift 1 (Program) to open the Program Catalog
2. Tap on the program name `Geometric`; the Program Editor opens to allow you to edit your new program



```
Geometric()
BEGIN
END;
EXPORT GEO_PDF(P,X)
BEGIN
RETURN P*(1-P)^(X-1);
END;
EXPORT GEO_CDF(P,X)
BEGIN
V:=0;
FOR A FROM X DOWNTO 1 STEP 1 DO
V:=V+GEO_PDF(P,A);
END;
RETURN V;
END;
```

The program is shown in the figure to the right. It has three parts:

- The program name, followed by an empty BEGIN…END block structure (highlighted in green)
- The exported GEO_PDF function (in yellow)
- The exported GEO_CDF function (in orange)

Let's take a look at each of these three parts in detail.

```
Geometric()
BEGIN
END;
```

This part is created automatically. We left the BEGIN…END block structure blank for the sake of simplicity.

```
EXPORT GEO_PDF(P,X)
BEGIN
RETURN P*(1-P)^(X-1);
END;
```

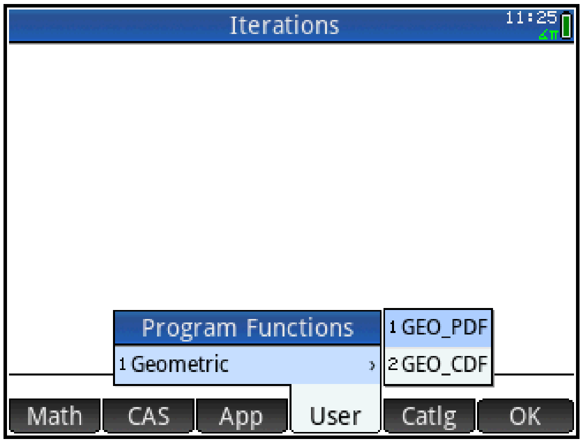
The EXPORT command makes the geometric probability density function GEO_PDF(P,X) visible outside of the program. The RETURN statement within the BEGIN…END block defines the function.

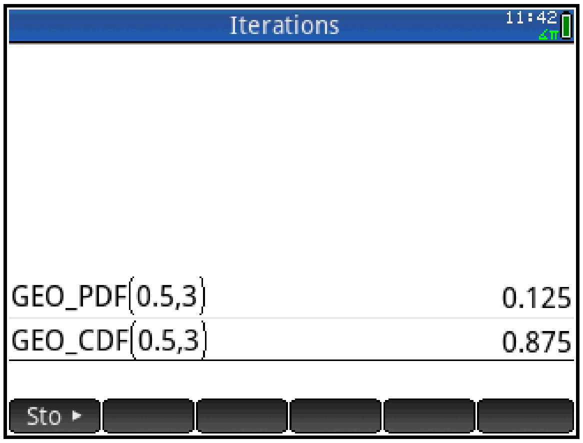
```
EXPORT GEO_CDF(P,X)
BEGIN
V:=0;
FOR A FROM X DOWNTO 1 STEP 1 DO
V:=V+GEO_PDF(P,A);
END;
RETURN V;
END;
```

The cumulative probability function GEO_CDF(P, X) initializes the variable V with a value of zero. The FOR…END loop computes the GEO_PDF value for each possible X-value and adds it to V to get a partial cumulative sum. Once the loop finishes, the RETURN command returns the final sum in V.

Return to Home view to use our new functions.

1. Press Settings to open Home view
2. Press Mem B to open the Toolbox menus and tap USER
3. Tap `Geometric` to see our two new functions, as shown in the figure to the right
4. Tap `GEO_PDF`; the command is pasted into the Home view command line
5. Complete the command GEO_PDF(0.5, 3) and press Enter; it returns the correct result: 0.125
6. Repeat Steps 2 and 3, but this time select `GEO_CDF`
7. Complete the command GEO_CDF(0.5, 3) and press Enter; it returns the correct result: 0.875



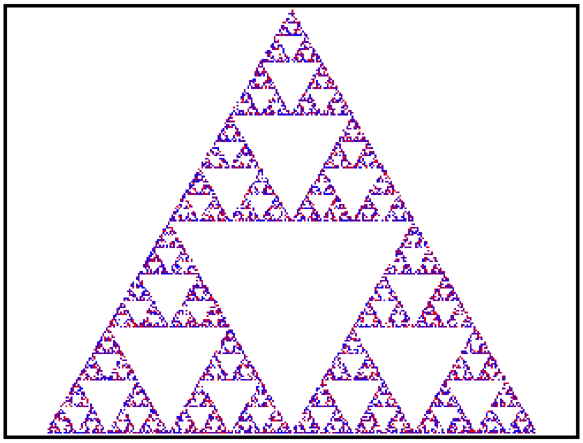


```
GEO_PDF(0.5,3)                 0.125
GEO_CDF(0.5,3)                 0.875
```

You have now learned two ways to create user-defined functions to extend the capabilities of your HP Prime. Along the way, you have learned how to use the Program Catalog and Editor to create new programs to extend your HP Prime. In the next example, we look at programming in a more general context.

## Programming: The Sierpinski Triangle

The program Sierpinski uses the Chaos Game approach to approximating the Sierpinski Triangle. In the Chaos Game, one starts with the three vertices of a triangle (named in order 1, 2, and 3) and a point as initial condition. An integer from 1 to 3 is chosen at random. If 1 is chosen then a point is plotted halfway between the initial condition and Vertex 1. Similarly, you plot the point halfway from the initial condition to Vertex 2 or Vertex 3 if a 2 or 3 were chosen. The new point is then made the initial condition and the Chaos Gam continues. In our program, we draw the first 10,000 points in a matter of seconds. The earliest points are red and they decrease in red and increase in blue as the game proceeds.



We will now send you the Sierpinski program using the HP Connectivity Kit.

## Program Listing

```
local x1, x2, x3, y1, y2, y3, xn, yn;
local a, b, color;

EXPORT Sierpinski()
BEGIN
x1:=160; x2:=299; x3:=21; xn:=160;
y1:=0; y2:=240; y3:=240; yn:=0;
RECT();
FOR a FROM 1 TO 10000 STEP 1 DO
b:=RANDINT(2)+1;
CASE
IF b=1 THEN xn:=(xn+x1)/2; yn:=(yn+y1)/2;END;
IF b=2 THEN xn:=(xn+x2)/2; yn:=(yn+y2)/2;END;
IF b=3 THEN xn:=(xn+x3)/2; yn:=(yn+y3)/2;END;
END;
color:=RGB(255-a/40,0,a/40);
PIXON_P(IP(xn),IP(yn),color);
END;
WAIT;
END;
```

## Comments

Declare the coordinates of the triangle and the initial condition as local variables. Also declare *a*, *b*, and *color* as local variables (more on these later).

Export the program so it appears in the Toolbox User menu.

Declare values for the vertices and initial condition.

RECT() with no argument clears the graphics display.

Local variable *a* is used in a FOR…DO…END loop

Local variable *b* contains the Chaos Game random integer

The CASE…END structure defines the point to plot, based on the random integer value.

The local variable *color* uses the RGB command to set a color for the point.

Turn on the pixel for that point, using the value of *color*.

Pause when finished until a key is pressed.

## Notes

We used physical screen coordinates rather than Plot view coordinates. The top left of the display is the point (0, 0) and the bottom right of the display is (320, 240). The point (x1, y1) is at the top middle of the display. The other two points were chosen to be at the bottom of the display and to form a triangle that is close to equilateral. The initial condition was chosen as the top vertex of the triangle.

The RANDINT(N) command generates integers from 0 to N, so we used RANDINT(2)+1 to generate 1, 2, or 3 and store it in the local variable *b*.

The RGB command creates a single hexadecimal integer from three decimal integers from 0 to 255. The first decimal integer is for the red component. Here we used 255-*a*/40 so that the red component would be strong at first and gradually die off. We kept the green component zero. The blue component is the opposite of the red component; it starts off close to zero and increases as the game continues. So early points are red, middle points are purple, and later points are blue.

The command PIXON_P(x, y, color) uses the same physical coordinates that we used for the points. If you prefer to use the Plot view coordinate system, use the command PIXON(x, y, color).

You can change the values of xn and yn to see that the initial condition really does not matter.

## Custom HP Apps with programs and notes: Sicherman Dice

Like regular dice, Sicherman dice are cubical in shape and numbered with positive integers. Unlike regular dice, the first Sicherman dices is numbered 1, 2, 2, 3, 3, and 4, while the second Sicherman dice is numbered 1, 3, 4, 5, 6, 8. If 2 Sicherman dice are rolled repeatedly, what does the distribution of the sum of each roll look like?

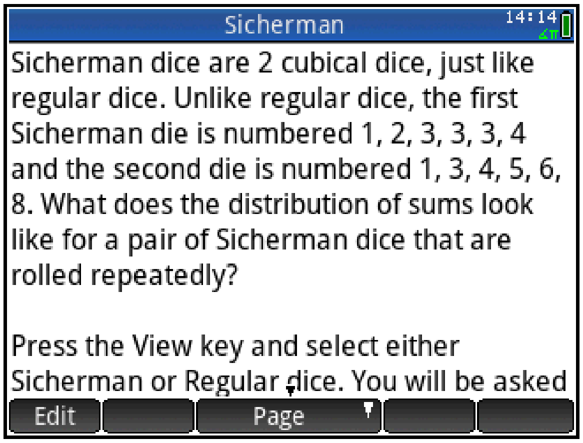
To simulate rolling regular and Sicherman dice, we created an app called Sicherman that has a program attached to it. Here is the process we followed to create the app:

1. Decide which HP App will be the basis of your new app
    a. We chose the Statistics 1Var app because we wanted to display a histogram of the distribution of the sums
2. Save the HP App under a new name and start it
    a. We chose the name "Sicherman"
3. Open the Program Catalog. The Program Catalog always lists a program with the same name as the current app. Open this program and edit it to add the functionality you need.

We will now send you the Sicherman App via the Prime Classroom Network. Then we will examine how it works and what the program attached to it contains.

1. Press Apps Info and tap on the icon for the `Sicherman` app

The app is programmed to open in its Info view (more on this later). Press Shift Apps Info (Info) at any time to return to this view. This view describes the app in general terms and directs the student to press View Copy to get started.

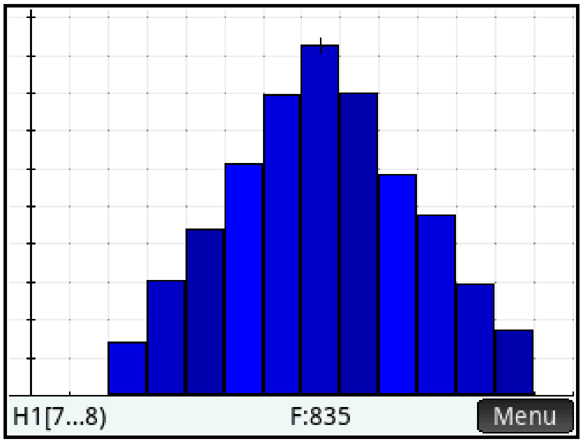


Sicherman dice are 2 cubical dice, just like regular dice. Unlike regular dice, the first Sicherman die is numbered 1, 2, 3, 3, 3, 4 and the second die is numbered 1, 3, 4, 5, 6, 8. What does the distribution of sums look like for a pair of Sicherman dice that are rolled repeatedly?

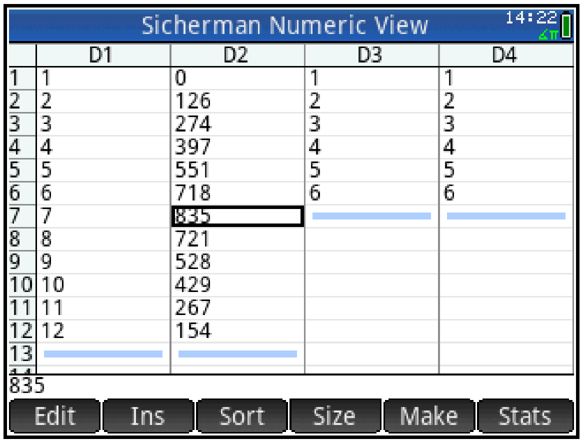
Press the View key and select either Sicherman or Regular dice. You will be asked

1. Press View Copy and select `Regular Dice`. You will be prompted to enter the number of rolls. Type 5000 and press Enter ≈.
2. The simulation will run and then the histogram of the sums of the rolls will be displayed.

As shown in the figure to the right, the distribution is symmetric and centered at 7, as expected.



3. Press Num to see the data.


Sicherman Numeric View

|  | D1 | D2 | D3 | D4 |
|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 1 |
| 2 | 2 | 126 | 2 | 2 |
| 3 | 3 | 274 | 3 | 3 |
| 4 | 4 | 397 | 4 | 4 |
| 5 | 5 | 551 | 5 | 5 |
| 6 | 6 | 718 | 6 | 6 |
| 7 | 7 | 835 |  |  |
| 8 | 8 | 721 |  |  |
| 9 | 9 | 528 |  |  |
| 10 | 10 | 429 |  |  |
| 11 | 11 | 267 |  |  |
| 12 | 12 | 154 |  |  |
| 13 |  |  |  |  |

835

Column D1 contains the possible sums (with 1 retained to make things easier). Column D2 contains the frequencies corresponding to each sum in D1. Columns D3 and D4 represent the two dice used in the simulation.

4. Now press View and select `Sicherman Dice`. Run a simulation of 500 rolls with these dice.

Are you surprised at the results? Look at Numeric view and compare the data to what is shown for the regular dice in the figure to the right.

We will now look at the Sicherman app program to see how it was constructed and how it behaves.

5. Press Shift 1 (Program) to open the Program Catalog
6. Tap on the program name `Sicherman (App)`
7. The Program Editor opens with the Sicherman program listing, shown on the next page with notes on each section.

## Program Listing

```
DOIT();
Plot();
EXPORT n, b, a;
VIEW "Sicherman Dice" Sich()
BEGIN
D3:={1,2,2,3,3,4};
D4:={1,3,4,5,6,8};
DOIT();
END;
VIEW "Regular Dice" Regu()
BEGIN
D3:={1,2,3,4,5,6}; D4:={1,2,3,4,5,6};
DOIT();
END;
```

## Comments

Declare two subroutines: DOIT() and Plot().

Export the variables *n*, *b*, and *a* outside of the program.

Add an option "Sicherman Dice" to the Views key menu

The program Sich() runs when this option is chosen. The program defines columns D3 and D4 to be the two Sicherman dice and then runs the program DOIT().

Add an option "Regular Dice" to the Views key menu

The program Regu() runs when this option is chosen. The program defines columns D3 and D4 to be two regular dice and then runs the subroutine DOIT().

## Program Listing

```
DOIT()
BEGIN
D1:={1,2,3,4,5,6,7,8,9,10,11,12};
D2:={0,0,0,0,0,0,0,0,0,0,0,0};
INPUT(n,"Number of Rolls","Rolls","Enter
number of rolls",50,50);
FOR a FROM 1 TO n DO
b:=D3(RANDINT(5)+1)+D4(RANDINT(5)+1);
1+D2(b)▶D2(b);
END;
Plot();
END;
Plot()
BEGIN
SetSample(H1,D1);
SetFreq(H1,D2);
H1Type:=1;
Hwidth:=1;
Hmin:=0;
Hmax:=13;
Xmin:=-0.5;
Xmax:=14;
Ymin:=-0.5;
Ymax:=1.1*MAX(D2);
Xtick:=1;
IF n<50 THEN Ytick:=1 ELSE
Ytick:=IP(Ymax/100)*10;
END;
CHECK(1);
STARTVIEW(1,1);
STARTVIEW(1,1);
END;
START()
BEGIN
STARTVIEW(6, 1);
END;
```

## Comments

The subroutine DOIT() defines D1 to be the possible sums (including 1 for ease) and D2 to be the frequencies of the corresponding sums in D1. Then an input form is created for n, the number of rolls. The FOR…DO…End loop defines *b* to be the sum of two numbers chosen randomly, one from each die. D2(b) is then incremented by one.

When the loop finishes, the Plot() subroutine runs.

The subroutine Plot() defines H1 in Symbolic view. It makes D1 the sample data and D2 the frequencies.

It sets the plot type to histogram and the bin width to 1.

It sets the Plot view window and makes a guess about where to put the ticks on the y-axis.

It makes sure H1 is checked (active) for graphing and then it displays the histogram in Plot view.

The subroutine START() runs whenever the app is started (by tapping its icon in the App Library or selecting it and tapping Start). It displays the Info view.

## Notes

The program Plot() actually runs not only whenever it is called as a subroutine, but also whenever you press Plot while this app is running. Likewise, the program START() runs when the app is selected in the App Library and you press Start. The other app control keys ( Symb, Num, Reset , etc.) can be tied to subroutines in a similar fashion. This gives you complete control over your custom app.

The VIEW command changes the options available when View is pressed. Each instance of the VIEW command adds exactly one option to the Views menu, in the order in which they are encountered in the program listing. You can reinstate standard Views options (Autoscale, etc.) as well, depending on the base HP app used.

The STARTVIEW command is used to present any view of the current app. View 6 is the Info view.

You noticed that the Sicherman app had a note attached to it, which is displayed when the app starts. The note is in the Info view of the app and is sent whenever the app is transmitted to another HP Prime.

When the Sicherman app is sent to an HP Prime, the Sicherman app program is sent along with it. You must open the Sicherman program on the receiving HP Prime to activate the program.

Thus, all custom HP apps have three files associated with them:

- The app file itself: Sicherman.hpapp
- The app program file: Sicherman.hpappprgrm
- The app note file: Sicherman.hpappnote

Usually, these three files are bundled in a single zip file that you drop into the Content Pane of HP Connectivity Kit. Right-click on this zip file and select `Send` to send the app, along with its note and program, to all connected HP Primes (or select a subset of the connected HP Primes in the Monitor pane and send to just those devices). Be sure to open the zip file to see if there are instructions for the teacher or student, or if there is a short introductory video.

## Program Listings

You can use the following program listings to copy and paste the programs into your HP Prime Virtual Calculator.

## Program: Geometric

```
Geometric()
BEGIN
END;
EXPORT GEO_PDF(P,X)
BEGIN
RETURN P*(1-P)^(X-1);
END;
EXPORT GEO_CDF(P,X)
BEGIN
V:=0;
FOR A FROM X DOWNTO 1 STEP 1 DO
V:=V+GEO_PDF(P,A);
END;
RETURN V;
END;
```

## Program: Sierpinski

```
local x1, x2, x3, y1, y2, y3, xn, yn;
local a, b, color;
EXPORT Sierpinski()
BEGIN
x1:=160; x2:=299; x3:=21; xn:=160;
y1:=0; y2:=240; y3:=240; yn:=0;
RECT();
FOR a FROM 1 TO 10000 STEP 1 DO
b:=RANDINT(2)+1;
CASE
IF b=1 THEN xn:=(xn+x1)/2; yn:=(yn+y1)/2;END;
IF b=2 THEN xn:=(xn+x2)/2; yn:=(yn+y2)/2;END;
IF b=3 THEN xn:=(xn+x3)/2; yn:=(yn+y3)/2;END;
END;
color:=RGB(255-a/40,0,a/40);
PIXON_P(IP(xn),IP(yn),color);
END;
WAIT;
END;
```

## App Program: InferencePlus (based on the Inference App)

```
local p,k;
GEO_PDF(p,k);
GEO_CDF(p,k);
EXPORT InferencePlus()
BEGIN
END;
VIEW "Geometric PDF" GEO_PDF()
BEGIN
INPUT({p,k},"Geometric PDF",{"P", "K"},{"Enter the probability of success, P","Enter the number of trials,
K"},{0.5,1},{0.5,1});
MSGBOX("GEO_CDF("+p+","+k+")="+p*(1-p)^(k-1));
END;
VIEW "Geometric CDF" GEO_CDF()
BEGIN
V:=0;
INPUT({p, k},"Geometric CDF",{"P", "K"},{"Enter the probability of success, P","Enter the number of the successful
trial, K"},{0.5,1},{0.5,1});
FOR A FROM k DOWNTO 1 STEP 1 DO
V:=V+p*(1-p)^(A-1);
END;
MSGBOX("GEO_CDF("+p+","+k+")="+V);
END;
```

## App Program: Sicherman (based on the Statistics 1Var App)

```
DOIT();
Plot();
EXPORT n, b, a;
VIEW "Sicherman Dice" Sich()
BEGIN
D3:={1,2,2,3,3,4};
D4:={1,3,4,5,6,8};
DOIT();
END;
VIEW "Regular Dice" Regu()
BEGIN
D3:={1,2,3,4,5,6};
D4:={1,2,3,4,5,6};
DOIT();
END;

DOIT()
BEGIN
D1:={1,2,3,4,5,6,7,8,9,10,11,12};
D2:={0,0,0,0,0,0,0,0,0,0,0,0};
INPUT(n,"Number of Rolls","Rolls","Enter number of rolls",50,50);
FOR a FROM 1 TO n DO
b:=D3(RANDINT(5)+1)+D4(RANDINT(5)+1);
1+D2(b)▶D2(b);
END;
Plot();
END;

Plot()
BEGIN
SetSample(H1,D1);
SetFreq(H1,D2);
H1Type:=1;
Hwidth:=1;
Xmin:=-0.5;
Xmax:=14;
Ymin:=-0.5;
Ymax:=1.1*MAX(D2);
Xtick:=1;
IF n<50 THEN Ytick:=1 ELSE
Ytick:=IP(Ymax/100)*10;
END;
CHECK(1);
STARTVIEW(1,1);
STARTVIEW(1,1);
END;
```