

Programs vs Functions



Unit 2 – Application – Teacher

7 8 9 10 11 12



TI-Nspire™



Coding



Language



Student



50 min

Introduction

The key difference between a function and a program on TI-Nspire is how they are used and what they return.

Functions:

Functions are designed to calculate and return a single value. A function keeps variables to itself. If you use 'x' in your function, it will not change the value of 'x' in your document. Functions built through the programming tool are designed to operate in a similar way as the calculator's inbuilt functions such as trigonometric functions: sine, cosine and tangent.

Programs:

Programs are more general-purpose and can perform a series of actions or calculations without necessarily returning a specific value. Programs are able to share or change variables currently in use, alternatively then can be made "Local" so as to avoid inadvertently changing a variable that has already been purposefully defined within the same problem.

This activity is designed to help understand the differences between functions and programs through a mathematical tool developed many centuries ago.



It is assumed that you have completed **Unit 2 Programming Basics**

You may return to the Skill Builder exercise at any time to review the instructions.

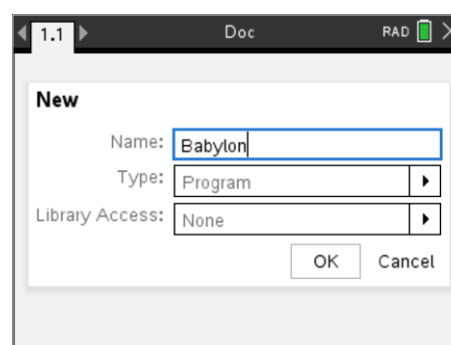


Program

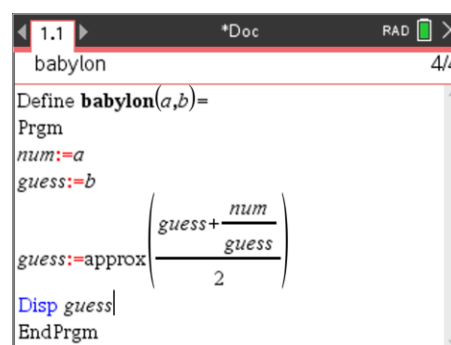
Start a new document, insert a Program titled:

Babylon

Note: Make sure the **Type** says Program.



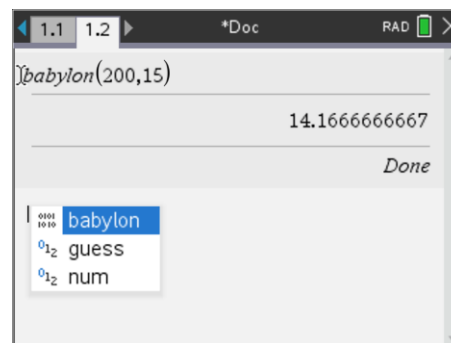
Complete the program shown opposite.



Insert a Calculator Application and run the program.

Use the values for a (number) and b (guess) as 200 and 15 respectively.

After running the program just once, press: **var**.



The variable list now includes:

$\begin{matrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{matrix}$ Babylon (Program)
 0_{12} Guess (Numerical value)
 0_{12} Num (Numerical value)

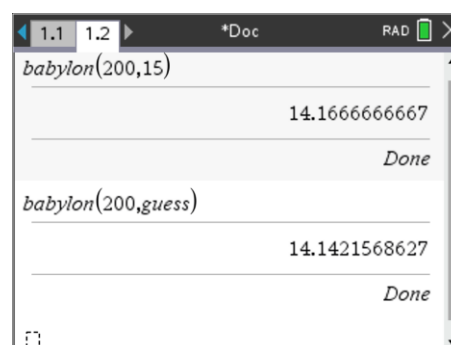
The 'guess' value can be inserted back into the program.

In this case the program acts upon:

Babylon(200,14.166...)

What happens if you press **enter** again? What number will it act upon?

Press **enter** a couple of times and observe what is happening.



Question: 1.

Run the program again, this time as: Babylon(2,1)

a) What value is returned?

Answer: 1.5

b) Repeat the process using Babylon(2,guess). What value is returned after multiple executions of the program?

Answer: After first 'guess': 1.41666666667 After multiple guesses: 1.41421356237

c) How does your answer relate to Babylon(200,#)

Answer: $\text{Babylon}(200,\#) = 10 \times \text{Babylon}(1,\#)$. The result is $1/10^{\text{th}}$ of the previous result.

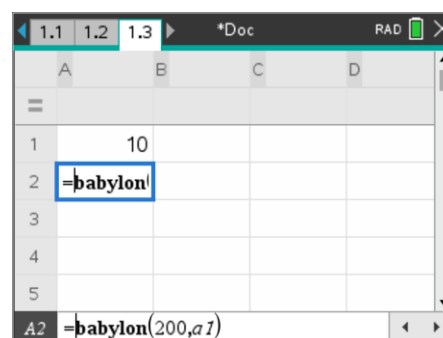
Teacher Notes: The Babylon program is an algorithm to determine the square-root of a number, (Babylonian square-root algorithm) hence the connection: $\sqrt{200} = 10\sqrt{2}$.

Insert a Spreadsheet Application.

In cell A1 enter the value: 10

In cell A2, insert an "=" sign and attempt to run the **Babylon** program using the number as 200 and guess entered in cell A1.

What happens when you do this?



Function

Insert a new problem in your current document:

doc > **Insert (4)** > **Problem (1)**

Select a **Calculator** Application.

Notice that the page numbering changes from 1.# to 2.#

Press **var** and check the list of variables in Problem 2.

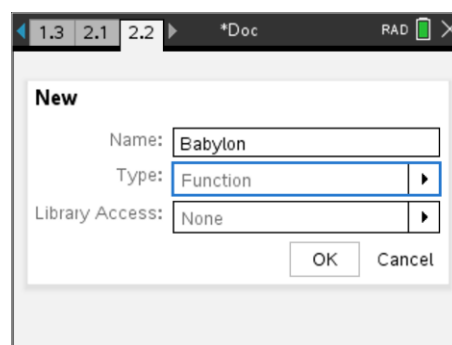
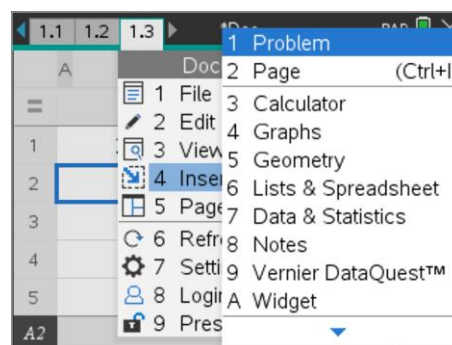
Inserting a problem creates a fresh space where all previously defined variables are not available. If you return to Problem 1, the variable list will return.

In Problem 2, press:

ctrl + **I** [Insert a Program Editor]

Call the new item: **Babylon**

In the Type drop-down list, select: **Function**

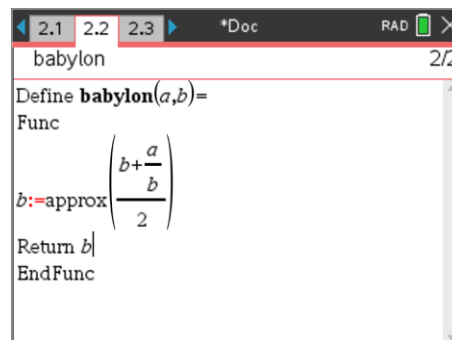


The first line now says: "Func" rather than: "Prgm".

The Babylon function involves the same calculations as before, we can use the same two variables a and b

The calculation can be returned directly, or stored and then "Return".

Enter the rule as: $\text{approx}((b + a/b)/2)$

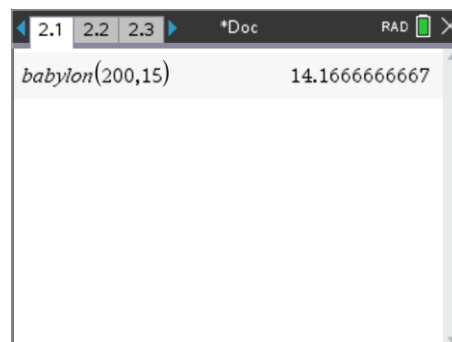


Save the function by pressing **Ctrl + B**.

Return to the Calculator Application and run the Babylon function using the same initial values as before: 200 and 15.

The process can easily be repeated by using: **Babylon(200,ans)**

The function and program are being used in the same way, to return a single value.



Insert a Spreadsheet Application.

In cell A1, enter the value: 10


In cell B1, enter the value: 200

In cell A2, enter the formula: =babilon(\$b\$1,a1)

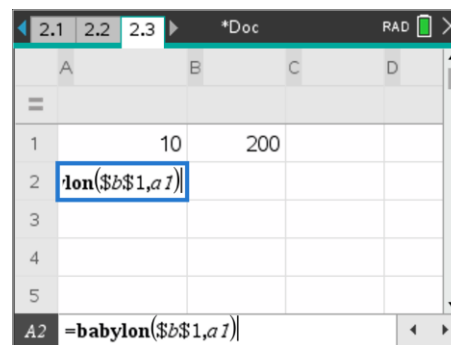
The “\$” in the cell referencing locks the “b” and the “1” and is referred to as an ‘absolute’ reference. (The same as Excel™)

Select cell A2 then press:

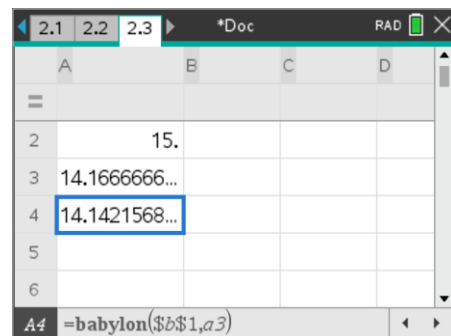
 > Data (3) > Fill (3)

Either use the mouse or tap the navigation pad/arrow down to cell A10 and then  to execute. A sample for the first cell is shown opposite.

Look at the formula bar in cell A4 (opposite). The formula is still pointing at cell B1, but the reference to cell A1 is now A3, this is called a relative reference. The formula in cell A2 points to A1, the formula in cell A3 points to A2 ... it points to the previous cell.



	A	B	C	D
1	10	200		
2	=babilon(\$b\$1,a1)			
3				
4				
5				



	A	B	C	D
2	15.			
3	14.1666666...			
4	14.1421568...			
5				
6				



The function command can be executed in the Spreadsheet application!

Question: 2.

Change the value in cell A1 to 1 and cell B1 to 2.


a) What value is returned in cell A2?

Answer: 1.5

b) What values are the cells progressively approaching?

Answer: 1.414213... (Square-root of 2)

Question: 3.

Return to the Calculator Application in Problem 2. Press  and record the variables now available.

Answer: Babylon() – Program only. Variables are not passed from functions to other applications.

Teacher Notes: This application covers the Babylonian algorithm for computing the square-root of a number using a simple recursive technique. The guess does not need to be particularly accurate. The algorithm essentially takes the average of the guess and the corrective feedback (original number ÷ guess). If the guess is close to the square-root, then

$$\text{Number} \div \text{guess} \approx \text{guess}$$

This new guess is fed back into the averaging algorithm so it continually gets better (closer).

Students need to understand the benefits of a function over program and program over function. Functions can be used in a spreadsheet in the same way that trigonometric functions can be. The same is true for other environments such as the Graphs, Geometry and Notes applications.

Just for fun(ction) you can also calculate: Babylon(200,Babylon(200,10)) ... a function of a function!