Unit 2: Assigning Values to Variables | **Skill Builder 2: Local variables**

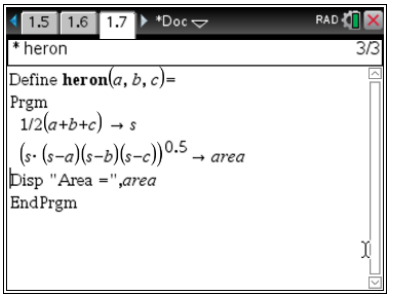
In this second lesson for Unit 2, you will learn about declaring **local** variables in a program

Objectives:

- Understand the need for **local** variables
- Use **local** variables in programs

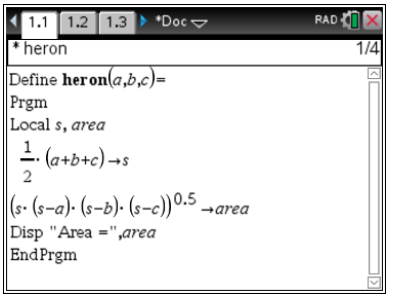
---

1. Begin by opening the document containing the **heron** program you wrote in Skill Builder 1 as seen in the image to the right.



Recall that this program creates the variables **s** and **area** in the current problem of the document. This is undesirable, and we'll learn how to prevent this issue now.

2. Below the **Prgm** keyword, add the statement **Local s, area**.



Note: The **Local** statement is located in **menu > Define Variables**.

3. When this line of code has been entered, use the shortcut **ctrl+B** to check and store the program.

4. In a Calculator app, before running the program, use the command **DelVar s, area** to delete the two variables from the problem.

Delete Variable is located in **menu > Actions** in a Calculator app.

5. Now, run the **heron** program.

6. When the program is done, select **var,** and notice that the only variable listed is the program **heron** itself.

What happened?

When you state or *declare* that a variable is '**Local**,' it tells the TI-Nspire™ CX to create the variable while executing the program and to delete the variable when the program ends so that it does not exist anywhere else in the current problem. This eliminates the issue of variables being created in the problem where they are not needed. If the problem already uses a variable that the program refers to, then declaring it **Local** within the program will not change the problem variable's value since the program makes its own (temporary) variable instead.

Teacher Tip: Local can declare several variables at once. **Local *s,t,u,v*** will declare all four variables to be local.

Even though a variable is declared as Local, it cannot be used until it is assigned a value.

Local *s*
Disp *s*

causes an error because ***s*** is not yet defined. It must be assigned a value first.

Summary
The 'scope' of a variable is the location(s) where the variable exists. In the TI-Nspire CX, variables live in the *current problem*. Adding a problem to a document gives you a 'clean slate' of variables. Problem 2 in a document knows nothing about the variables in Problem 1 and vice-versa.

If programs in a problem use the same variables as the problem, then the programs can inadvertently (or intentionally) create or use those variables. Sometimes it might make sense to link the program variables and the problem variables but keep that in mind when creating programs.

Declaring variables as **Local** stops the program from creating or affecting those variables in the problem.

Functions, on the other hand, deal with the 'scope' principle in a different way as we'll see in the next lesson.