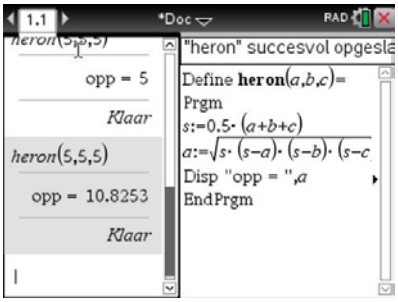
Unit 2: Waarden toekennen aan variabelen | **Oefenblad 2: Lokale variabelen**

In deze tweede les van Unit 2 leer je over het 'declareren' van lokale variabelen in een programma

Doelen:

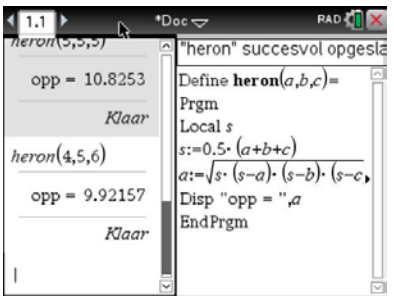
- De noodzaak van lokale variabelen begrijpen
- **Lokale** variabelen gebruiken in programma's

Begin met het openen van het document dat het programma **heron** bevat, dat je in oefenblad 1 schreef, zoals je kunt zien in het plaatje rechts.



Bedenk dat het probleem in dit programma is dat het onbedoeld de variabele **s** creëert in het document. Dit is onwenselijk en we zullen nu leren hoe we dit probleem kunnen voorkomen.

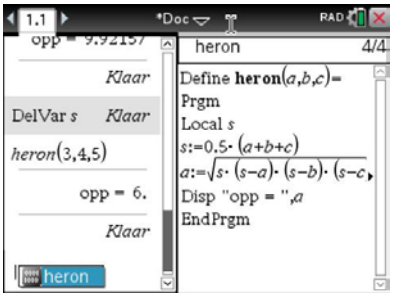
Voeg onder het sleutelwoord **Prgm** de volgende opdracht toe:



Local s je kunt **Local** vinden in **menu > Variabelen definiëren**

Wanneer deze opdracht is ingevoerd check dan de syntax en slag het programma op door ***menu > Syntax controleren en opslaan > Syntax controleren en opslaan*** te selecteren (of door de sneltoetscombinatie **ctrl-B** op de rekenmachine te gebruiken.)

Gebruik in de toepassing Rekenmachine de opdracht **DelVar s** om de variabele s te verwijderen.



(**Variabele wissen** is te vinden in **menu > Acties**.)

Voer het programma **heron** uit.

Druk op **var** en merk op dat de enige variabele het programma **heron** zelf is.

Wat is er gebeurd?

Wanneer je 'declareert' (of stelt) dat een variabele '**Local'** is, vertelt dit de TI-Nspire™ CX om de variabele te creëren terwijl het programma wordt uitgevoerd en de variabele te verwijderen wanneer het programma eindigt, zodat de variabele nergens in de huidige opgave meer bestaat. Dit neemt het probleem weg dat de variabele **s** wordt gecreëerd in de opgave en daar blijft. Ook is het zo, dat als de opgave deze variabele al gebruikt, het **Local** maken van de variabele binnen het programma de waarde van de variabele niet verandert omdat in plats daarvan het programma zijn eigen (tijdelijke) variabele maakt,

Docenten Tip: Met de opdracht **local** kunnen meerdere variabelen tegelijkertijd 'gedeclareerd' worden. Local *s,t,u,v* zal alle vier de variabelen als lokaal bestempelen.
Ook al is een variabele 'gedeclareerd' als lokaal (local) dan kan deze niet gebruikt worden totdat er een waarde aan is toegekend:

Local s
Disp s

Veroorzaakt een fout omdat s nog niet is gedefinieerd. Er moet eerst een waarde aan toegekend worden.

Samenvatting

De ‘speelveld’ van een variabele is de plaats waar de variabele bestaat. In de TI-Nspire™ CX, ‘leven’ variabelen in de *actuele TNS opgave*. Het toevoegen van een opgave aan een document geeft je een ‘schone lei’ voor de variabelen. Opgave 2 in een document weet niets van de variabelen in opgave 1 en omgekeerd.

Als programma’s in dezelfde opgave dezelfde variabelen gebruiken dan kan dat onbedoeld of bedoeld variabelen creëren of gebruiken in de actuele opgave. Soms kan het logisch zijn om de programmavariabelen en de variabelen in de opgave aan elkaar te verbinden, maar houd dat in gaten wanneer je programma’s schrijft.

Het declareren van variabelen als **Local** voorkomt dat het programma deze variabelen in de opgave creëert of beïnvloedt.

Functies, aan de andere kant, gaan op een andere manier om met het principe ‘speelveld’ zoals we zullen zien in de volgende les. Blijf erbij!