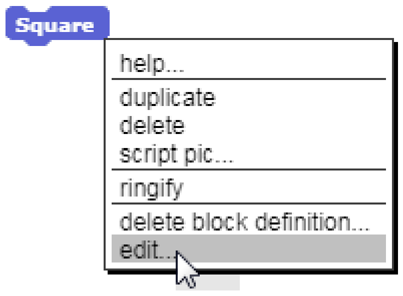
Variables

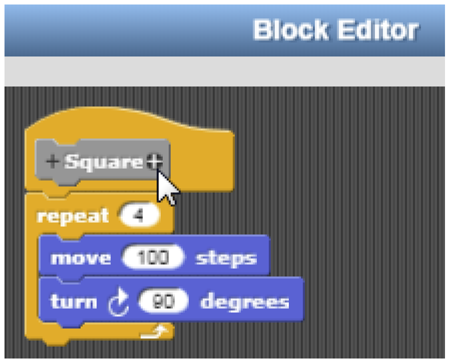
Editing the Square Procedure

The initial **Square** command is somewhat limited. It can only draw a square that is 100 steps on a side. Edit the **Square** command by right-clicking the name of the procedure. This produces a drop-down menu. Click the *Edit* option to edit the procedure.



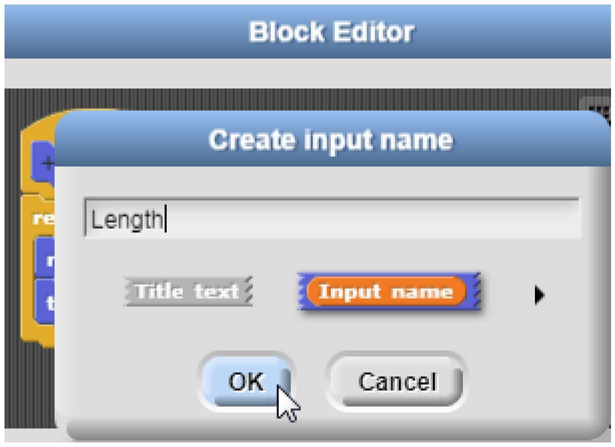
Adding an Input to the Procedure

Click the “Plus” sign (+) to the right of the title ***Square*** in the *Block Editor* to access another option, *Create Input Name*.



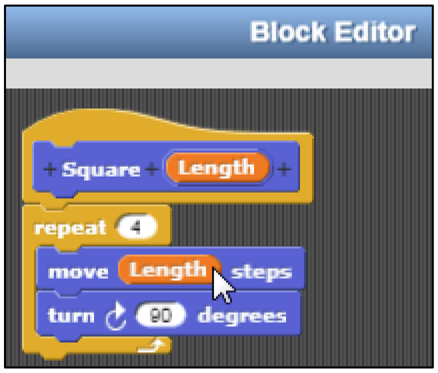
Naming the Variable - Size

Enter the name *Length* as the name of an input to the command. This name was chosen because we are going to use this variable to vary the length that the turtle travels as it completes each side of the square. Click OK to confirm this choice.

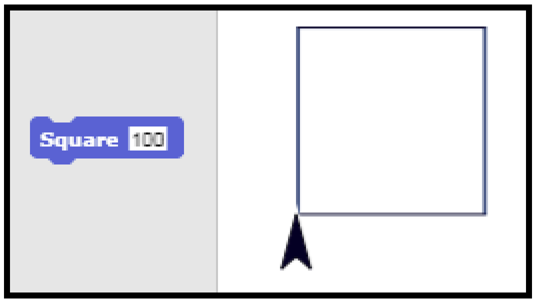


Using the Size Variable in the Square Procedure

Drag the orange oval labeled ***Length*** into the the *step* field of the **Move** command replacing the number.

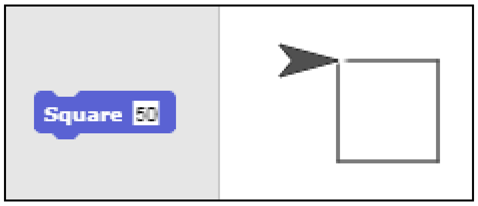


The input to the command **Square** can then be used to specify the length of each side of the square that will be drawn.



Drawing Squares of Varying Size

If 100 is entered as the input, a square that is 100 turtle steps on each side will be drawn. If 50 is entered as the input, a square that is 50 turtle steps on each side will be drawn.



Variables – A Key Computing Concept

The input to the command **Square** is known as a *variable* because it makes it possible to vary the size of the square drawn. Different values can be assigned to the variable known as *Length*. *Variables* are the second big idea in computing. Variables allow the actions of a procedure to be adjusted according to the needs of the situation.

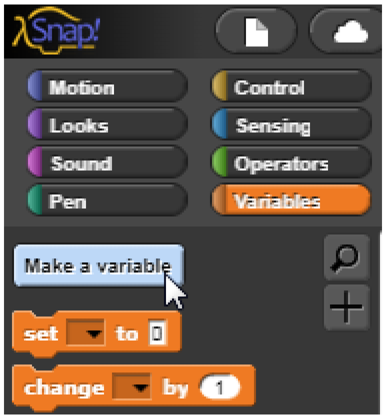
Together, the ability to define new commands and create variables give computing languages much of their power. They enhance the ability of the programmer to create complex applications.

Global Variables

The Make a Variable Option

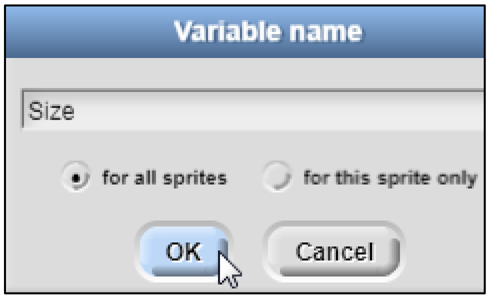
The variable ***Length*** only affects the block of code within the **Square** procedure. It is called a local variable because it *only* affects the local code within the **Square** procedure.

Global Variables can be used *across man*y procedures and can also be created. To create a global variable, select the *Make a Variable* option found in the *Variable* command palette (orange). Variables created in this way can be used across multiple blocks of code. For example, a global variable could be used to specify the size of a square and a triangle. It could even be used to control creation of a series of squares of varying size.

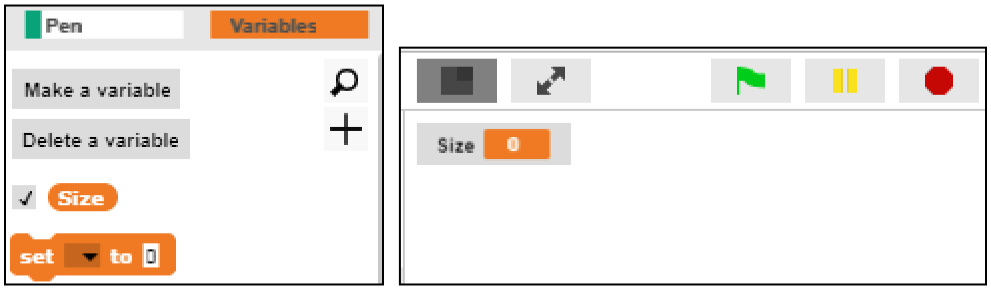


The Variable Name Editor

Select the *Make a Variable* to access the *Variable Name* editor. Enter the name ***Size*** as the name of the new global variable. This variable will be used to vary the size of different polygons such as triangles and squares.



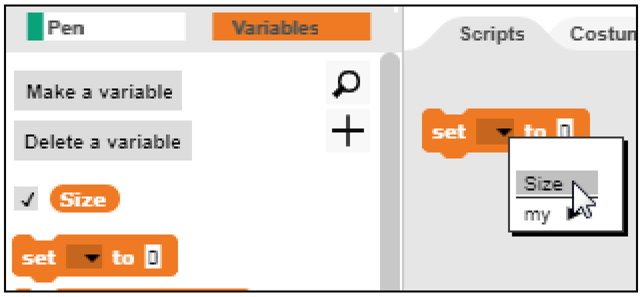
The new global variable, ***Size***, will appear at the top of the *Variables* palette



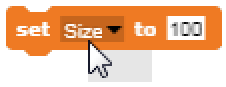
When the box beside the variable is checked, the value of the variable will be displayed on the stage (on the right-hand side of the screen). A default value of zero is assigned to new variables.

Assigning a Value to a Variable

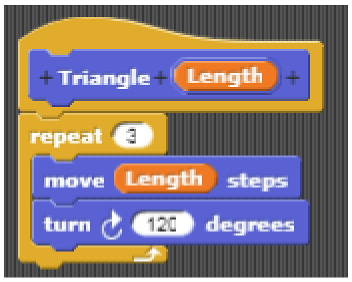
Use the **Set** command to assign a value to a global variable. The **Set** command is found under the *Variables* palette.



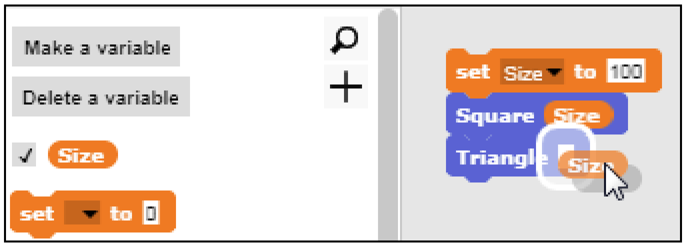
Enter a value of 100 for the global variable size.



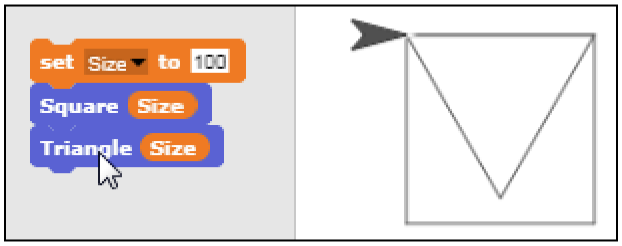
Define a **Triangle** procedure. Enter a value of 3 in the **Repeat** statement and a value of 120 degrees for the angle of each turn. Why does 120 degrees achieve the desired result?


```
+ Triangle + Length +
repeat 3
    move Length steps
    turn ↻ 120 degrees
```

Use the global variable *Size* to control the size of both a square and the triangle. Drag the orange oval labeled Size into the inputs of the **Square** and the **Triangle** procedures.

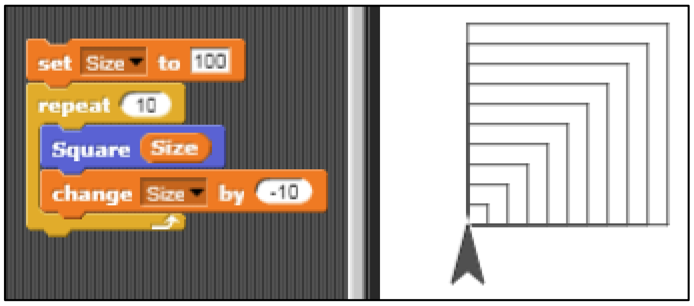


Then click the code block to draw a square and a triangle that are each 100 steps on a side. Experiment with other values for *Size*.


```
set Size to 100
Square Size
Triangle Size
```

Repeating a Series of Squares

Combine the *Size* variable with a **Repeat** command and the **Square** procedure to create a series of squares. After each square is drawn, the value of the variable is decreased by 10 using the **Change** command. A series of squares decreasing in size results.



```
set Size to 100
repeat 10
    Square Size
    change Size by -10
```

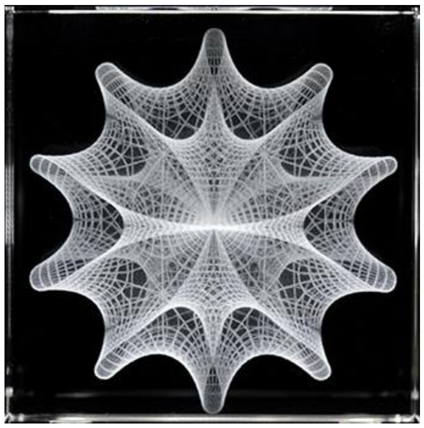
Algorithmic Thinking – A Key Computing Concept

Algorithmic thinking is the process of automating solutions through a series of ordered steps (sequencing) and conditional logic (flow of control), and is an essential characteristic of computing. Developing complex algorithms, such as the one above to draw a series of shapes, is an important skill to develop. *Snap!* provides a simple platform for constructing such algorithms by connecting and sequencing the command blocks. In a later section, we will explore conditional logic and controlling the flow of an algorithm in more depth.

Patterns in Art

Digital Artists

Artists like Bathsheba Grossman use mathematical patterns to create art. Their designs are etched in acrylic or glass using digital fabrication tools such as laser cutters.



Bathsheba Sculpture Website

Spinning the Square to Create a Pattern