Name_____ Date_____

Algebra
Unit 3 Line Master 6a

Using Code to Transform Triangles

You will be creating new subprograms by writing code that will transform the vertices of a triangle.

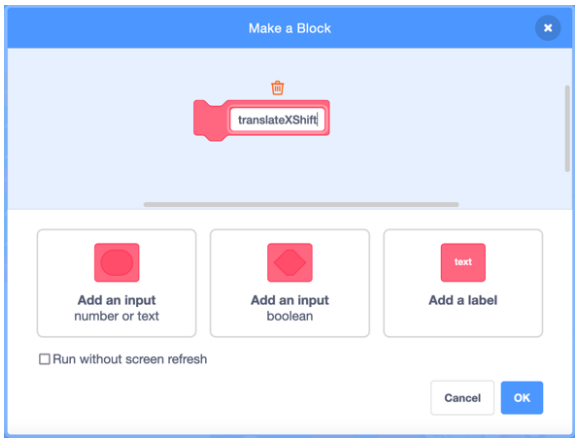
Click on the link to open the Scratch application with subprograms that you used in Master 5.

If you have a Scratch login, save the project in your Scratch account by selecting **Remix** at the top of the screen. A login is not required to work with the code, but you will not be able to save your changes without it.

https://scratch.mit.edu/projects/741053989/editor/

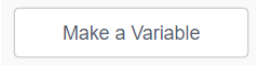
Part 1: Translating the Triangle Horizontally

1. Let's write code for a subprogram that will translate the 3 vertices horizontally by shifting the *x*-coordinate of each point.
   First, make a new block for the subprogram by selecting **My Blocks**.
   Name it **translateXShift.**
   It will appear in the code editing area.
   Drag it to an area where there is room to place blocks below it.

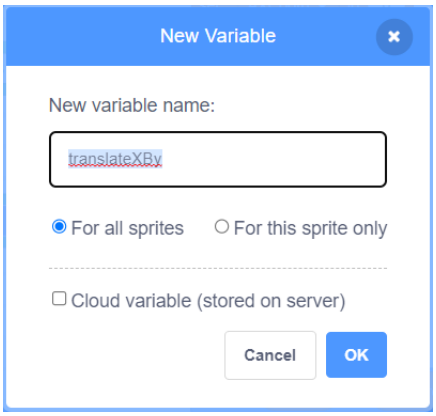


2. You will use **set** blocks to shift the *x*-coordinate of each point.
   To do this, you will need a variable to hold the value by which the *x*-coordinates will shift.
   - Select Variables.

   - Select **Make a Variable**.



   - Name the variable **translateXBy**.

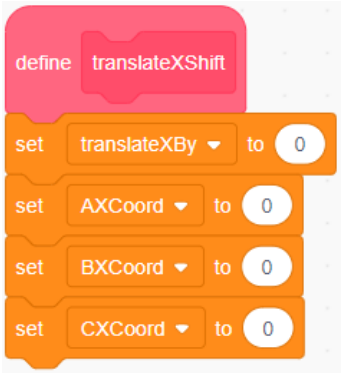


Name_____ Date_____

Algebra
Unit 3 Line Master 6b

Using Code to Transform Triangles (cont'd)

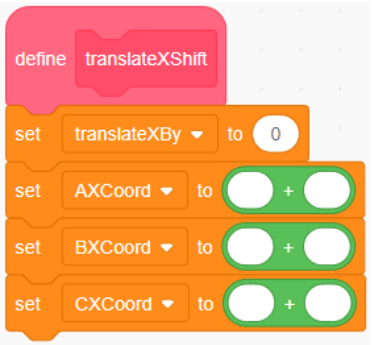
3. Drag four **set** blocks to beneath the subprogram called **translateXshift** and adjust the pulldown menus as shown on the right.



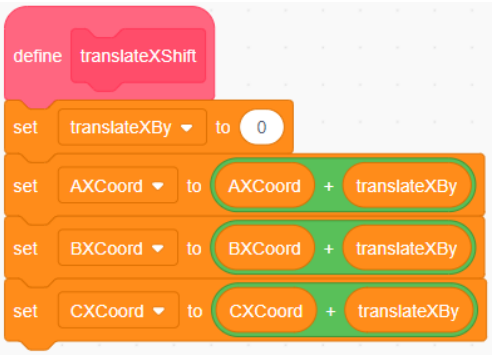
   Look carefully to make sure you have selected the *x*-coordinates of A, B, and C in the **set** blocks and not the *y*-coordinates.

4. You will use **operator** blocks to add the value by which the *x*-coordinates will shift to the original values of the *x*-coordinates.

   - From **Operators**, drag an **addition operator** to inside each of the **set** blocks, as shown on the right.



   - From **Variables**, drag the *x*-coordinate variables into the first part of each **addition operator**.
   - Drag the **translateXBy** variable to the second part of the ***addition operator*** for each ***set*** block.



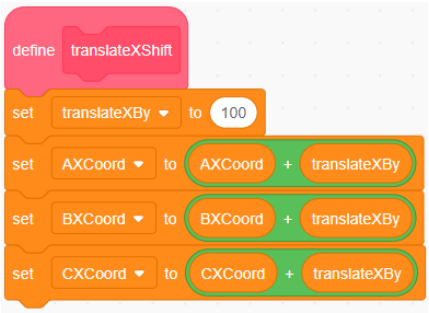
Note: This can be finicky—take your time. If things don't snap into place as you expect, just pull them apart and try again!

Name_____ Date_____

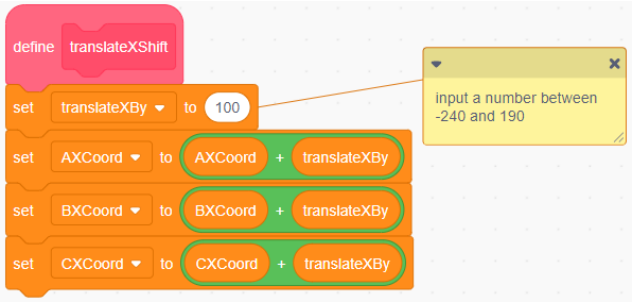
Algebra
Unit 3 Line Master 6c

Using Code to Transform Triangles (cont'd)

5. Let's start by shifting the *x*-coordinates by 100.
   To do this, enter 100 in the **set** block for the **translateXBy** variable, as shown on the right.

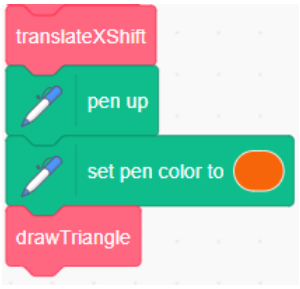


6. Add a comment to let anyone who uses your code know the range of values they can enter for the **translateXBy** variable.
   To do this, right-click on the **set translateXBy** block, select **Add Comment,** and type this statement:
   Input a number between -240 and 190.
   This will ensure that, based on the original triangle, the value the person enters will result in a translation that can be displayed on the stage. This range will need to be adjusted for different starting triangles.



7. Finally, you need to add code to the main program to call your new subprogram and draw a new triangle with vertices that have shifted horizontally 100 units from the original vertices.

   Adjust the code by adding the 4 blocks shown here to the end of the main program.



Name_____ Date_____

Algebra
Unit 3 Line Master 6d

Using Code to Transform Triangles (cont'd)

Here's what these 4 blocks of code do:

- After the main program draws the blue triangle, the **translateXShift** subprogram is called up to shift the *x*-coordinates of the vertices by 100 units.
- Next, the pen is lifted up to avoid drawing a line while the sprite moves to the new starting location.
- Then, the pen colour is changed to orange so that the translated triangle can be distinguished from the original triangle.
- Finally, the **drawTriangle** subprogram is called to action once again!

8. Click on the green flag to test out the revised application. Here's what the stage should look like when the program has executed. Describe how the triangle has moved.

9. Alter the code to translate the original triangle 100 units to the left instead of the right. What change did you make?

Name_____ Date_____

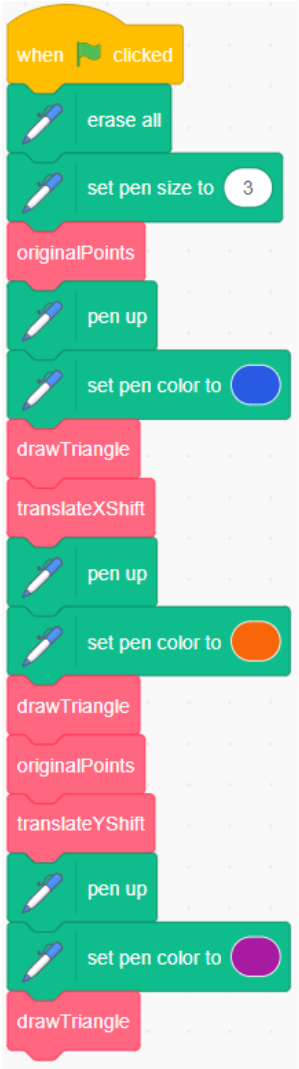
Algebra
Unit 3 Line Master 6e

Using Code to Transform Triangles (cont'd)

Part 2: Translating the Triangle Vertically

Next, you'll write code for a subprogram that translates the 3 vertices of the triangle up or down along the *y*-axis. Here are some tips:

- Start with your version of the application from Step 8 that includes the **translateXShift** subprogram.
- Write similar code to your **translateXShift** subprogram, but call it **translateYShift**.
- You will need a new variable called **translateYBy.** Start with a translation value of 100 units.
- Once you have written your subprogram, add the following comment to your **set** block for the variable **translateYBy**:
  Input a number between -180 and 130.



Here's what the main program might look like. Notice that it resets to the original points before translating vertically by calling the **originalPoints** subprogram. This means that the triangle will be shifted vertically from the original triangle and not from the triangle that you translated horizontally. It is possible to translate the second triangle instead. This is up to you as the programmer!

If you choose to do this, your program will be missing the block that calls the **originalPoints** subprogram. Otherwise, it should look the same as what is shown here.

If you think your **translateYShift** subprogram is not working properly or you get stuck, check with your teacher for a solution.

Name_____ Date_____

Algebra
Unit 3 Line Master 6f

Using Code to Transform Triangles (cont'd)

Additional Challenges

- Write code for a subprogram that reflects the triangle points in the *x*-axis. Call the subprogram **reflectXAxis**.
  Hint: You will need a multiply operator.
  Why will the subprogram shown here work?

- Write code for a subprogram that reflects the triangle points in the *y*-axis. Call the subprogram **reflectYAxis**.
  Think about how it will be similar to **reflectXAxis** and how it will be different.
- Prompt the user to enter values for the horizontal translation shift.
  Hint: Use the **Ask and wait** block under **Sensing**.