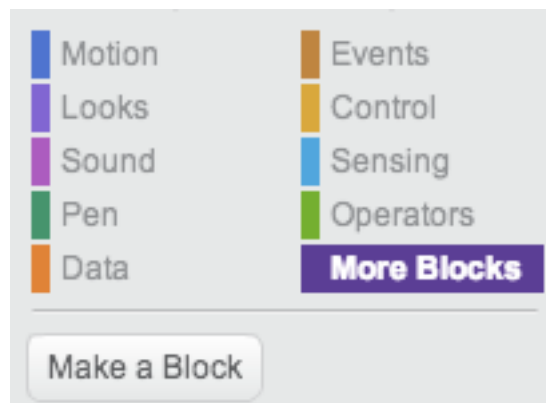


LESSON 6: March 2 due March 9

First Topic

We are going to learn about BYOB this week.
This acronym stands for Build Your Own Blocks.

If you look at the bottom right of the Scratch color choices, you will see it say MORE BLOCKS in Purple.



When you make a new block, you are adding a new word to the Scratch language or defining a new procedure where some work is going to happen.

HW 6.0A So I can make a block to build a square of any specified size.

In this **first required 4 min video** at <http://youtu.be/P05EMU9sqSk> I am building a block called SQUARE and using it to achieve Spinning Squares.

Or I can make a block to create a new command. Or I can make a block that handles the question/answer dialogue with the student in a quiz program.

Once again, the BYOB feature of Scratch makes it into a real computer language.

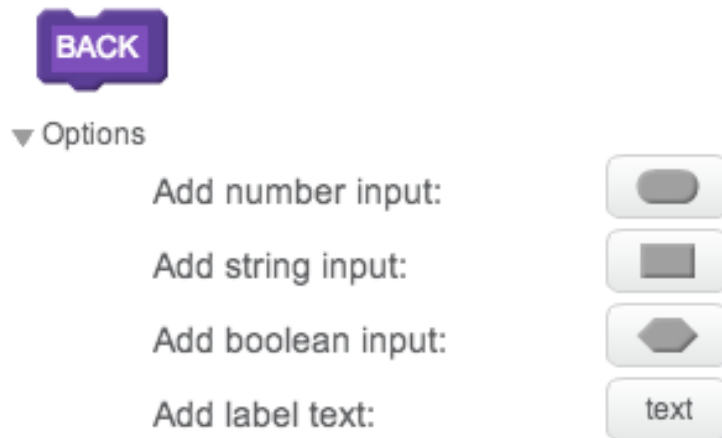
For example, we have the MOTION command



that we have used since our first lesson, but we do not have one that GOES BACKWARD 10 STEPS. So we use MORE BLOCKS to create one, even though we could put a negative number in front of "steps." When it says MAKE A BLOCK, we choose BACK and then notice there are OPTIONS below.



When we click on OPTIONS, we get several choices, each with a purpose



The first 3 are called INPUTS and they allow us to "pass info" to the new block called BACK. For our purposes now, we want the first type (numeric) so that we can issue commands like BACK 30 or BACK 140. Yes, I know we can MOVE -30 or -140 but I want to pretend we cannot and want to create this NEW command called BACK.

HW 6.OB This optional 7 minute video at <http://youtu.be/4pNpcn2R1Ck> takes you through building the new COMMAND called BACK. It also has a section on the BACKPACK feature for those of you whom I have not yet shown via screenshare.

Or we might one day create a block called REVERSE that would take a string and reverse the order when we type the command REVERSE (any word).

Boolean refers to True or False and that is yet another input, e.g. we could say RESPOND (X>Y) where X>Y is either true or false.

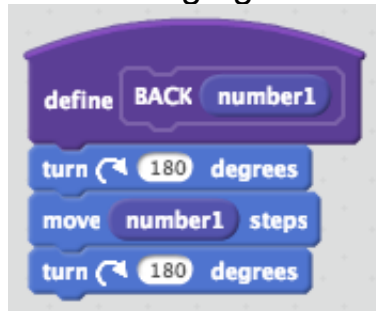
And LABEL TEXT allows you as programmer to put some comments or notes about this block for your future reference or that of someone else if you are on a programming team.

So for our BACK block, we choose NUMERIC and it would look like this:

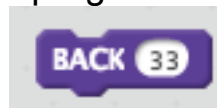




We now put in the script for this command -- turning 180 degrees, moving the specified number of steps and then turning again.

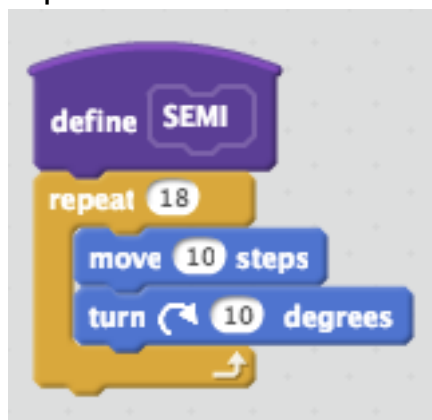


and we now can go back to our main program and issue commands like



and everything acts as if Scratch has a new command!

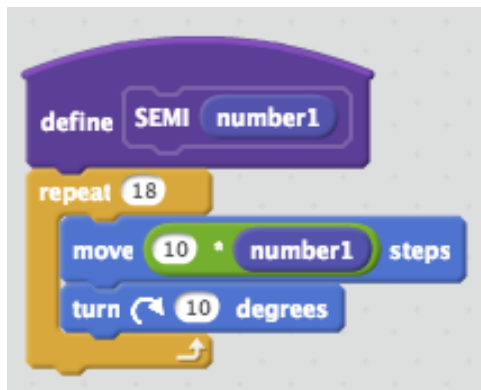
How about a block that draws a SEMICIRCLE which can be useful when drawing letters of the alphabet like "S" or "C" or "G"



which results in

Want to make the SEMI block fancier?

How about a numeric input that specifies whether the semicircle is bigger or smaller than one we just drew? We can EDIT the BLOCK called SEMI to have a numeric input and then use NUMBER1 as a multiplier for the number of steps.



so SEMI .5 is



and SEMI 1.5 is



The fact that both BACK and SEMI use the variable NUMBER1 which was made by the system is not a problem, since it is a "local variable" used internally when the programmer issues the command.

For another example of a BLOCK with a STRING INPUT, consider how useful it would be to have a BLOCK called MUSIC so that you could type in the first few notes of a song and it would play it!

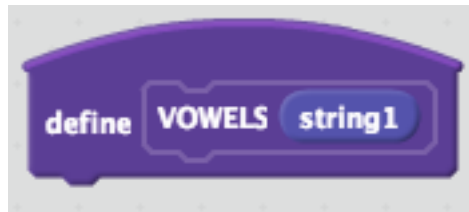


If you had this block working (not impossible to write) then you could put in your program

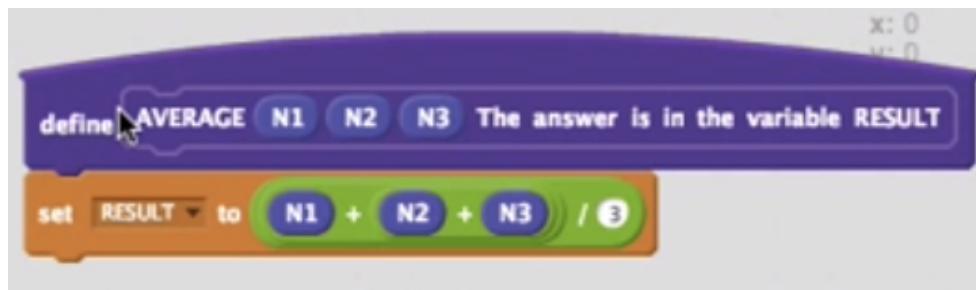


and it would play the first 7 notes of *Doe A Deer*, one of the few songs I remember from my piano playing days as a teenager ;-)

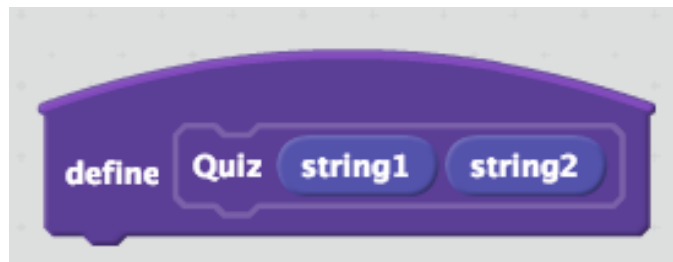
Or you could have a VOWELS block that would count the number of vowels in a string.



Or you could have an AVERAGE BLOCK that automatically finds the average of several numbers. HW 6.0C Here is a required 5.5. minute video on building the block called AVERAGE -- http://youtu.be/l4Qzw3j_TqE -- using three numeric parameters. Hopefully this video will solidify your awareness of BYOB.

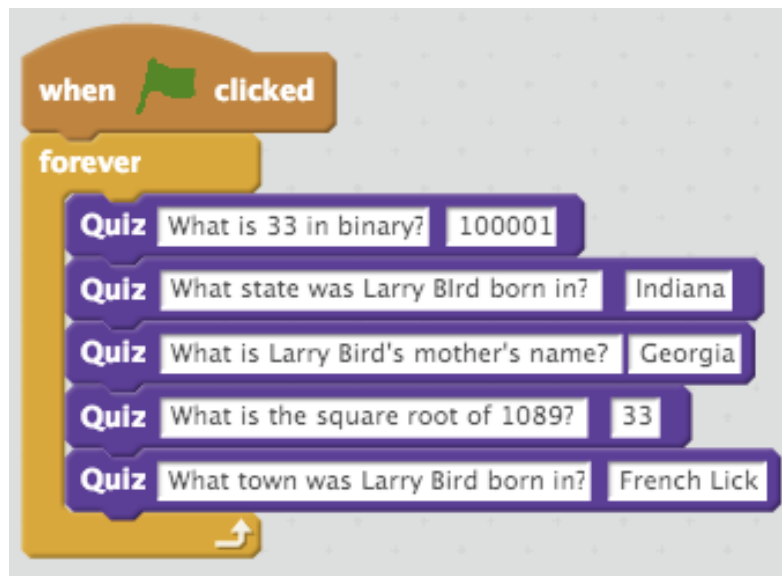


Here is one last example that you will weave into your 6.1 HW so it is the most important example. I am going to use the BYOB feature to create a quiz game that will ask a bunch of questions of the user.



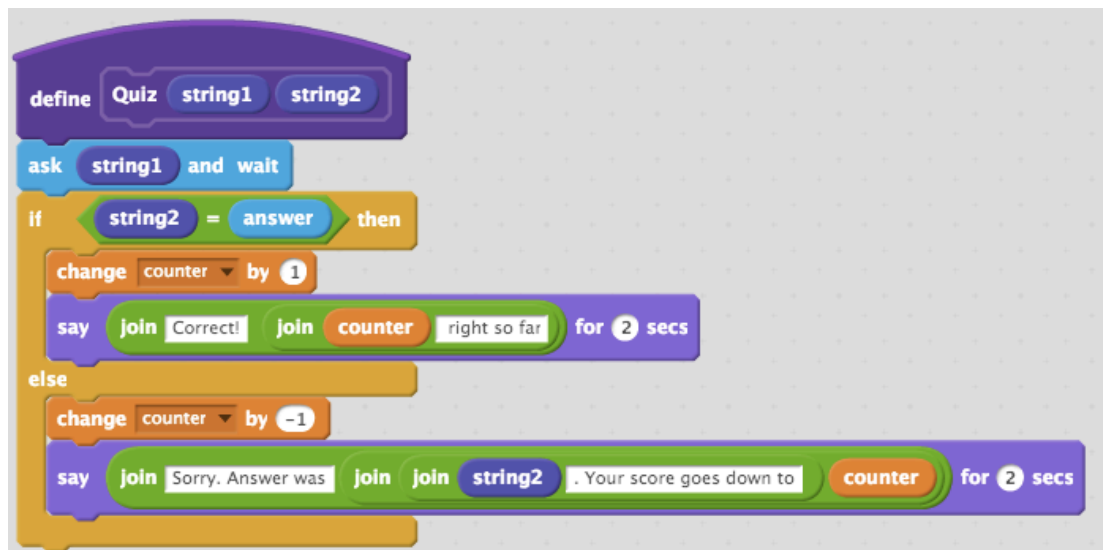
The two variables STRING1 and STRING2 are called PARAMETERS. We are passing these parameters from the main program to the BLOCK and saying "hey Block, please take charge of asking the question called STRING1 to the user and evaluate what he or she responds based on the answer I am giving you in STRING2."

One more point of advocacy regarding BLOCKS. It allows you as the programmer to keep your code shorter and more efficient. The code I have written for my quiz program would have to be replicated 5 times in my main program if I did not have blocks. Just imagine if you were charged \$1 per line of code! Efficiency in coding is a plus.



Instead, this is my main program where I have my five questions. The block called QUIZ takes care of all the work. In other computer languages, what we call blocks in Scratch might be called SUBROUTINES or PROCEDURES.

Here are the details of what goes on within the block called QUIZ.



Notice how the computer asks the question called STRING1 of the user and then evaluates whether STRING2 match ANSWER, either adding or subtracting a point from the user's score via the variable COUNTER. Now using the COUNTER, you



can do what you want on your screen. I have chosen to program a basketball to start bouncing and eventually reach Tom Brady on this screen.



The script for the ball is based on the script that one of Jeff's students (from Houston) used. Even though I can't say I understand it 100%, I can now use it effectively and want you to do the same. I have programmed the use of the COUNTER to make the ball bounce a little higher each time you get one quiz question correct. I want you to do something similar in the context of your 4.1 program, bouncing a fish, a volleyball or an object from Downton Abbey.

In my program called HW 6.1 Adding a Quiz to your 4.1 Screen Chase, you will see all of these elements. You can remix the program and use your backpack and move any scripts you want into your own 4.1 Creation (which now was enhanced several times in lesson 5). The URL is <http://scratch.mit.edu/projects/17847231>

So your HW 6.1 is an enhancement to your evolving 4.1 project

When done, I want you to have your 6.1 include

- a bouncing ball or bouncing anything
- a quiz program that keeps track of points using BLOCKS
- the reward of seeing your 4.1 Screen Chase animation happen after a certain number of points are scored using BROADCAST

Please send an email to your partner and me when you are ready for us to see your 6.1 creation.

On the next two pages – to help you – I am including the scripts of the bouncing ball and wizards hat from my 6.1. Please make this program your own by doing your own thing with these techniques in your own creative way!



Here is the script of the Bouncing Ball Sprite1 that animates more and more based on the counter.

Script 1 (Left): When clicked, switch backdrop to Quiz Backdrop, set counter to 0, go to x: 200 y: 50, show, then a forever loop. Inside the loop, if counter > 0, set jump to counter * 4, change y by 10, repeat until touching color (green), change jump by -1, and change y by jump.

Annotation 1: THESE SCRIPTS ARE FOR THE BOUNCING BALL WHICH HAS A GREEN PLATFORM THAT IT JUMPS FROM. YOU NEED THAT BUT CAN CHOOSE YOUR COLOR.

Annotation 2: THE 4 IS THE JUMP FACTOR THAT MAKES THE BALL JUMP HIGHER EACH TIME COUNTER INCREASES. YOU CAN ALTER THE 4 IF YOU WANT.

Script 2 (Top Right): When clicked, forever loop: repeat until touching color (green), change y by -5, if not touching color (green) then set jump to 0.

Annotation 3: THIS IS THE MAGIC SCRIPT FOUND BY JEFF'S STUDENT THAT MAKES THE BALL "LAND" ON THE GREEN PLATFORM.

Script 3 (Bottom Right): When clicked, forever loop: if counter > 5, hide, switch backdrop to Stadium, broadcast Winner.

Annotation 4: THIS IS THE KEY SCRIPT THAT GOES TO THE TOM BRADY 4.1 ANIMATION SCREEN ONCE THE USER GETS OVER 5 POINTS.

Try your best to understand the script on the left. Pretend that COUNTER = 2. This makes JUMP start out as 10. So in the bottom REPEAT loop, it changes Y by 9, then 8 then 7 until 1 which is why the ball GOES UP. Then JUMP is -1 so it descends 1 pixel in the first nanosecond, then descends 2 pixels in the second nanosecond, then 3, then 4 which means it speeds up until it touches "and lands" on the green. I am in awe of whomever was this clever to first program this!



This is the script of the Wizard Hat Wizard Hat that asks the questions

Script 1 (Left): When clicked, switch backdrop to backdrop1, show, then a forever loop with five Quiz blocks: "What is 33 in binary?" (100001), "What state was Larry Bird born in?" (Indiana), "What is Larry Bird's mother's name?" (Georgia), "What is the square root of 1089?" (33), and "What town was Larry Bird born in?" (French Lick).

Script 2 (Right): Define Quiz string1 string2, ask string1 and wait, if string2 = answer then change counter by 1, say join Correct! join counter right so far for 2 secs, else change counter by -1, say join Sorry. Answer was join join string2 . Your score goes down to counter for 2 secs.

Script 3 (Bottom Left): When I receive Winner, hide.

That's it folks!

Be well everyone and hope you stay in touch. Teaching you and getting to know all of you has been fun. I wish you the best of success with your computer usage and your careers. Even though our 30 minute phone sessions/teaching sessions end this Monday 3/9, if I can help you at some point in the next 33 years by email or a 10-15 minute phone call, I would be glad to!

Be well, everyone! Steve

p.s. in a week or so, I will send you an optional end of course survey along with an optional procedure for a certificate for PD hours



p.s. for any ambitious types, here is an OPTIONAL bonus topic for week 6. The topic -- FRACTALS -- makes for a great one-on-one lesson during our final half hour conversation this coming week. I love fractals and love the fact that I can make one using Scratch. You will feel the same way (I think) if you work at it!

Optional Bonus

Let us begin with an explanation of recursion -- an important part of computer science and computer programming theory. Please stay with me on this journey! This is a great way to end the course if you can create your own **Fisher-Price First Fractal** that uses variables, blocks and recursion. Stay on the journey! Persevere!

Recursion and Fractals -- Exciting & Empowering

Here is the easiest definition I could find from the website of

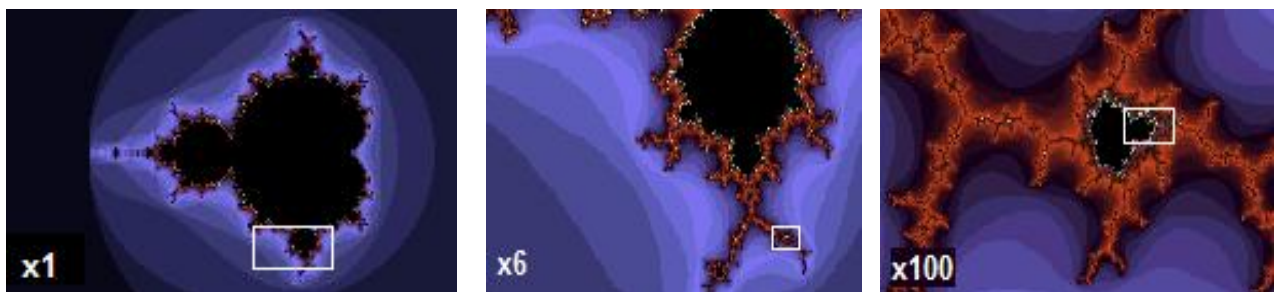
<http://www.techterms.com/definition/recursivefunction>

If you look up *Recursion Computer Programming* on Google, you get so much amazing complexity.

"A recursive function is a function that calls itself during its execution. This enables the function to repeat itself several times, outputting the result and the end of each iteration. Recursive functions are common in computer science because they allow programmers to write efficient programs using a minimal amount of code. The downside is that they can cause infinite loops and other unexpected results if not written properly. For example, the function may be terminated if the number is 0 or less or greater than 9. If proper cases are not included in the function to stop the execution, the recursion will repeat forever, causing the program to crash, or worse yet, hang the entire computer system."

Here is my 3 minute video -- <http://youtu.be/r40MHcsWxql> -- hopefully humorous but informative attempt to explain with my Larry Bird doll what recursion is and why it is different from repetition using the REPEAT command.

Starting in the 1970s, computer programmers were able to use the techniques of recursion to create fractals. You may have heard of the Mandelbrot fractal below on the next page. When you zoom in to this fractal on the left you get the graphic in the middle and then when you zoom in again, you get the graphic on the right. Each "zoom" yields a shape that is proportional (similar) to the original one.



This paragraph from <http://en.wikipedia.org/wiki/Fractal> may be of help:

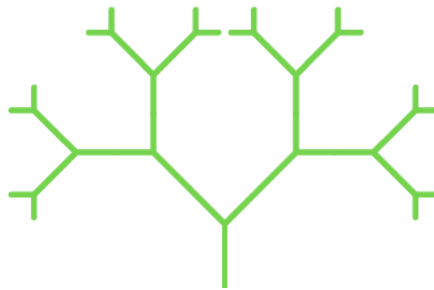
"The mathematical roots of the idea of fractals have been traced through a formal path of published works, starting in the 17th century with notions of recursion, then



moving through increasingly rigorous mathematical treatment of the concept to the study of continuous but not differentiable functions in the 19th century, and on to the coining of the word fractal in the 20th century with a subsequent burgeoning of interest in fractals and computer-based modelling in the 21st century. **The term "fractal" was first used by mathematician Benoît Mandelbrot in 1975.**

Mandelbrot based it on the Latin *frāctus* meaning "broken" or "fractured", and used it to extend the concept of theoretical fractional dimensions to geometric patterns in nature."

So our journey today is to make our own fractal.



Video Part 1 (5 min) <http://youtu.be/WQfEd5AcnVY>

This gives you the orientation to what a fractal is and an overview of the one we are going to code from scratch using Scratch.

VideoLesson Fractals Part 2 (18 min) <http://youtu.be/LKkVzmBB5Zs>

This takes you through the nitty gritty programming for creating a tree fractal. This is a tough challenging topic yet needs to be an important part of any coding course. I wish I could teach this to someone in 33 seconds but I can't!

The above two videos will show you how to make this primitive "tree fractal" which of course I have shared with you in my Scratch library/studio

<http://scratch.mit.edu/projects/13740603/>

program name: Recursion with Fractals (Make Your Own Fractal 2013)

Here is a better step-by-step presentation of creating your first fractal that has simpler and more efficient code that corresponds to below.

<http://scratch.mit.edu/projects/32791442>

program name: My First Fractal (step by step 2014)

STEP 1

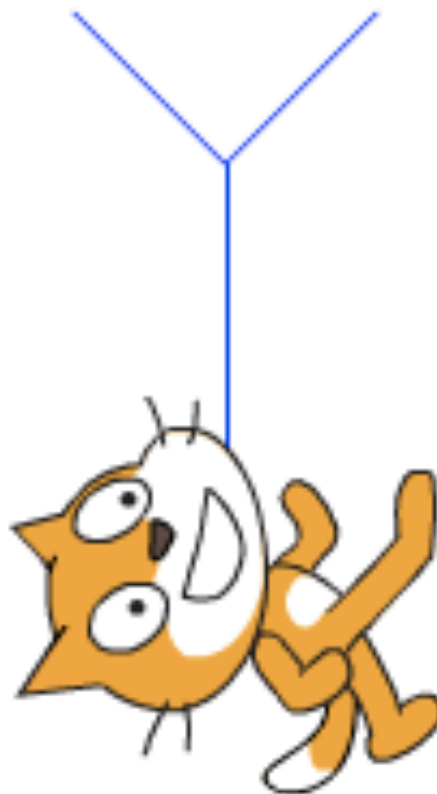
We begin by making an uppercase Y with each "limb" equal to half of the main trunk. Observe that the cat comes back to the starting spot (very important).

Coding

Online Lessons at <http://teachingcompany.com/jan15>



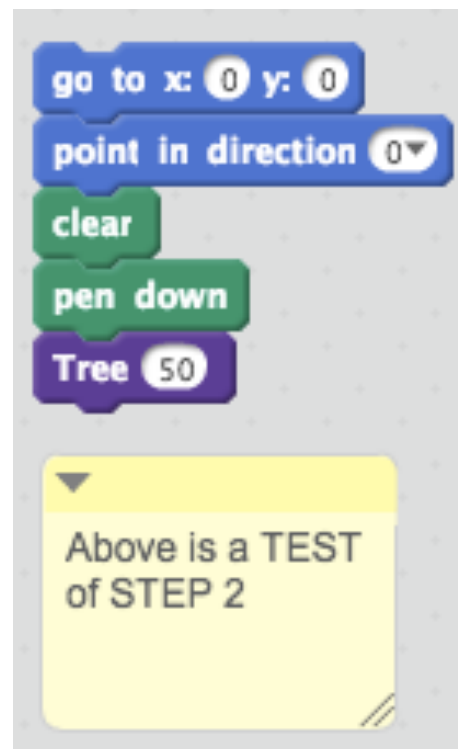
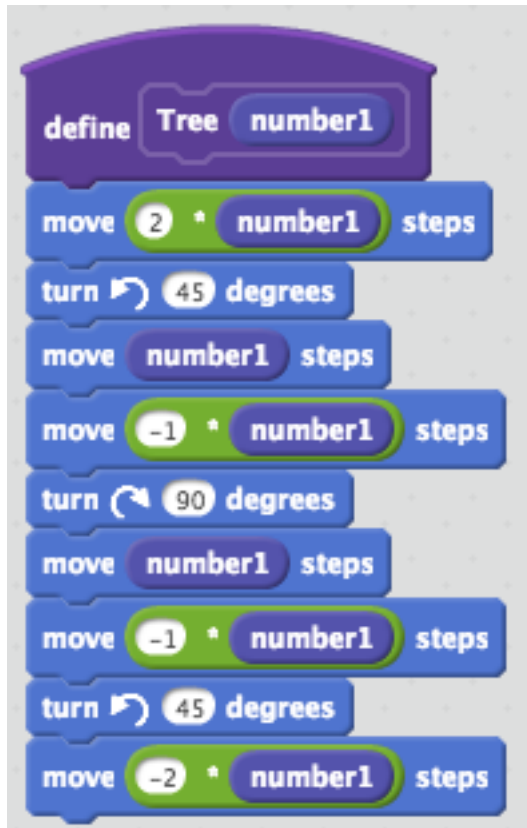
Lesson 6 page 12



STEP 2

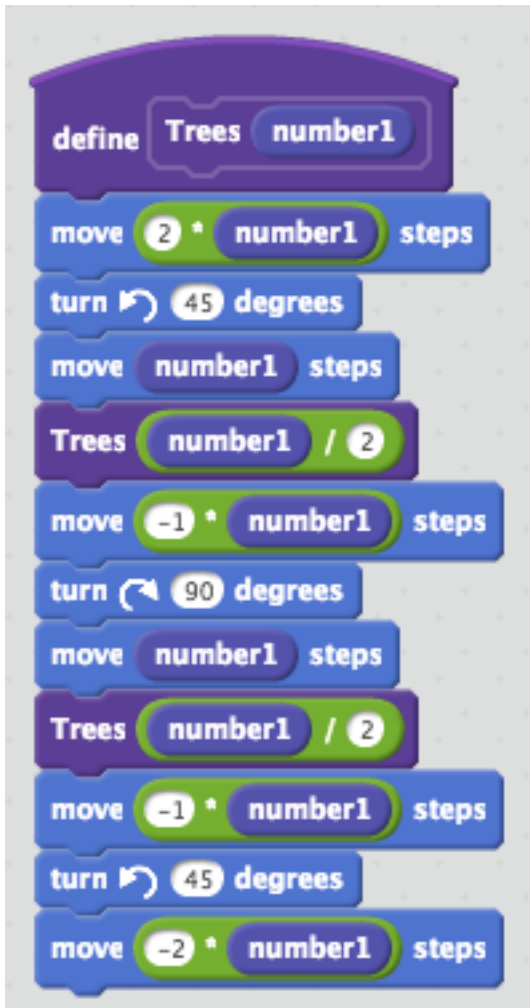
We now turn this into a new block called TREE that makes a variable sized "Y" where in this example NUMBER1 is taking on the value of 50 as in the STEP 1 example.

Make sure you test STEP 2 to get the same Y as you had in STEP 1.



STEP 3

Below is STEP 3 where in my demo I modified the TREE block, calling it TREES just so I can show you both STEP 2 and STEP 3. You do NOT need to have both the TREE and the TREES blocks. You can succeed with one block that is like TREE modified in step 3 and step 4.



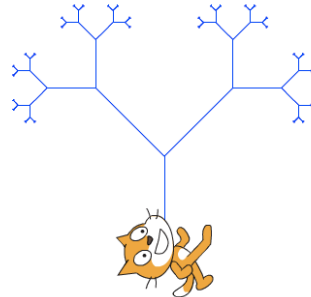
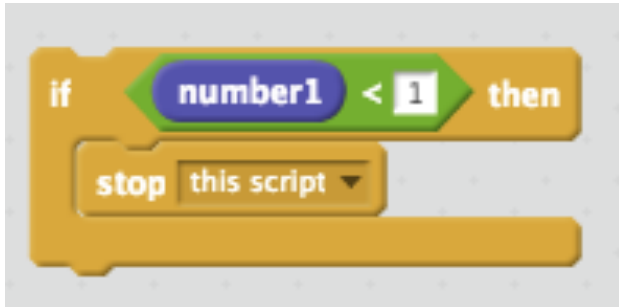
This STEP 3 is very important even though it is a failure. You can see that the computer is trying to make the "Y" but it never finishes. At the top LEFT of each subsequent "Y" it tries to make another "Y" that is smaller. But it never finishes so what we have in STEP 3 is a series of "left sides of each Y."



summercore

STEP 4: We are so close to success. We need just ONE IF STATEMENT to STOP the recursion when number 1 gets "too small" which you are welcome to define.

We add this one statement to our loop



and now we have it since now each subsequent and smaller "Y" is done until the variable is less than 1.

Below is STEP 3 where I modified the TREE block, calling it TREES just so I can show you both STEP 2 and STEP 3. You do NOT need to do this.

You can have just one block named TREE.

Below is a test of TREES for STEP 3 and STEP 4.

Note that we are starting at 0,-100 so that more of the screen can be used.

Optional HW 6.2

Study, study, study!

Practice, practice, practice!

Consider this your final exam.

Then wake up the next day and create the fractal from scratch without notes!

If you succeed, buy a drink and charge it to me!

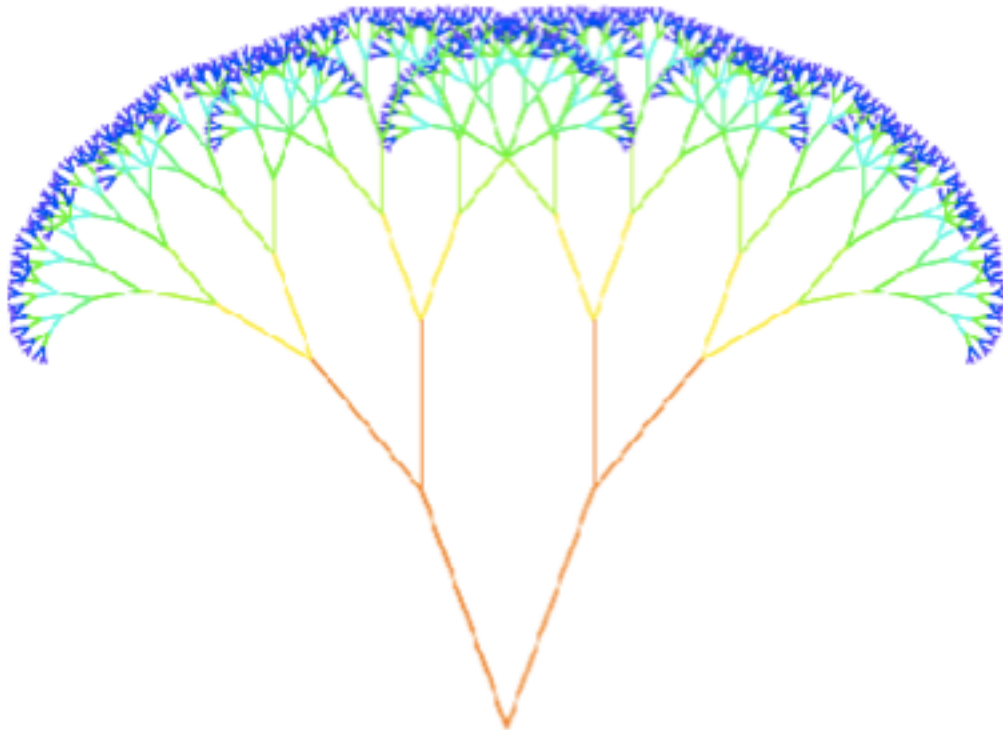
If you don't succeed and have to look at notes, then try it again the next day.

Modify your first fractal (or mine) to have colors. Modify your first fractal to have thicker lines. Modify your first fractal so that the left branches of the tree are not symmetric with the right branches in terms of thickness, angle or color.

A Better Tree Fractal by someone else on Scratch

This fractal tree from the library of NGMR is much nicer. I have remixed it for you and it is called Remix Fractal Tree by NGMR

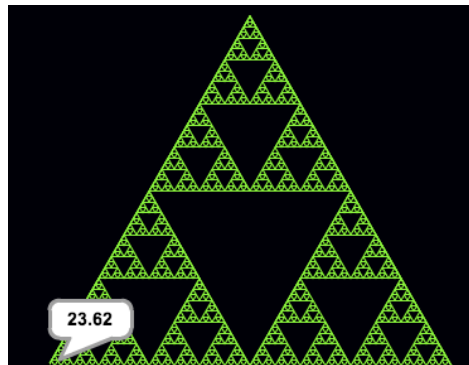
<http://scratch.mit.edu/projects/13941742/>



A Different Type of Fractal called The Sierpinski Triangle

I have also remixed for you a famous fractal called the Sierpinski Triangle and it is called Remix Sierpinski by S65 and Cyclone103

<http://scratch.mit.edu/projects/13941868/>



I hope you see the way that each of these graphics is a fractal in that the big picture is proportional and similar to any magnified portion.

Connections to Education and Our Kids

How does fit into education, particularly lower school? Kids can and should learn about these graphics and they can identify and see real world examples such as rivers, trees, leaves and snowflakes.

Creating connections between computers, mathematics and nature is part of developing in children an enthusiasm for what is now being packaged as STEM in our schools -- Science, Technology, Engineering and Mathematics.

See <http://fractalfoundation.org/2009/02/fractals-on-the-earth/>
Fractals on the Earth

or see <http://math.rice.edu/~lanius/frac/>
A Fractals Unit for Elementary and Middle School Students

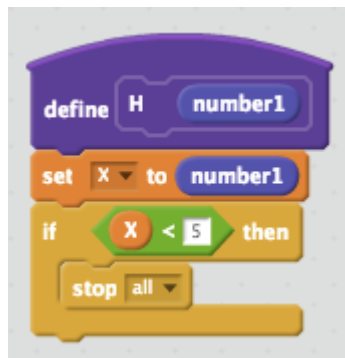
Next Optional Item

Make your own fractal using one of the block letters you created of variable size in a previous lesson.

Let's say you made a H of variable size.

Then you can make a fractal by adding the H command at each corner of the H.

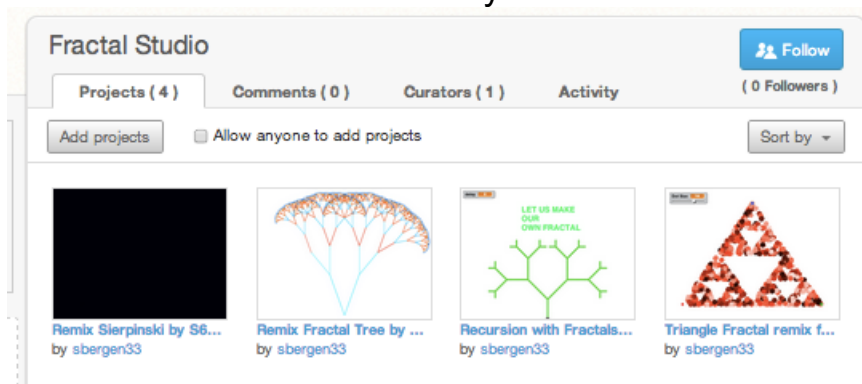
Here is the big picture:



then do the steps for an H of variable size
and at each corner of the H you include



Get the logic? It will draw an H of specified size NUMBER1 which then goes into X. As long as X stays above 5 then all is good. At each "corner" of the H, it will do a smaller H of one third size. If you succeed with your H fractal, you can then sing the song from West Side Story "Recursion, recursion, I just wrote a program with recursion." Let me know and I will add it to my new fractal studio.



URL of my fractal studio? <http://scratch.mit.edu/studios/358166/>

INDEX of HIGHLIGHTS can be found on the last page

- ☑ **HW 6.0A** Watch the required video on BYOB (p1)
- ☑ **HW 6.0B** Watch optional video on the BACK command (p3)
- ☑ **HW 6.0C** Watch the required video on BYOB with AVERAGE (p5)
- ☑ **HW 6.1** Enhance your evolving 4.1 project to include the bouncing ball, a quiz sequence on any topic and the reward of seeing the chance scene after the user achieves a number of points. Send to your partner and me. (p7)
- ☑ **HW 6.2 Optional Bonus:** Read about Fractals and then create your first fractal either following directions or with me at our final screenshare (pages 10-18)