**Recursion Project: Balloon Tree**

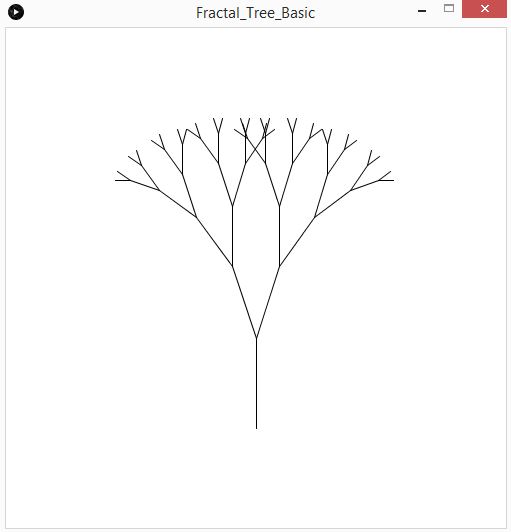
Trees can be generated through recursive functions. For this assignment you will be provided some preliminary code and will be required to extend it to achieve the project goals.

The Balloon Tree is a somewhat famous piece of Children's Literature by Phoebe Gilman. Some of you may have seen this book as a child.



Your project will be to modify the supplied code to create your own Balloon tree. (A tree with circular leaves of random color and size). The following program will be provided to you as sample code, which generates a 2-branch tree (copy-and-paste into a new Processing sketch:

Two Branch Sample Code.rtf  (file is included in the GitHub Repository)



Using the provided sampled code as a template, begin be reading through the program and try to understand it. Try experimenting by modifying the existing code and see how it impacts the execution. See if you can predict what your changes will do to the program.

Once you've familiarized yourself with the sample code, your assignment is to implement the following extensions into the program:

1. Make it a 3-branch Tree  
     
   The sample code creates a tree where 2 new branches start from the end of each branch of the previous level. Modify the program so that instead there are three branches occurring.
2. Add some Leaves (Balloons)  
     
   For this to be a Balloon Tree, it needs some balloons for leaves! Create a function called drawLeaf() and use it to create a circle of random color on the end of the branches. To make a good-looking tree, there are a few tasks involved in making the leaves:  
     
   **Random leaf color** - use random number generation to give the leaves random color. If this ends up being a little crazy (new colors each frame) - don't worry. That is acceptable for meeting the base requirements. Fixed random colors will be an extra for experts challenge.  
     
   **Draw leaves based on depth** - It doesn't make sense to have leaves growing right out of the trunk, so use a conditional statement to determine when leaves should be drawn. For the base requirements, leaves should be drawn on every branch that has a depth less than 5  
     
   **Random Leaf size based on depth** - Balloon should be create with a random size. However, the range of possible sizes should depend on the depth; the close to the outside of the tree, the smaller the leaves should be. Again, you may run into the issue of the leaf size changing each frame, but this is acceptable for the base requirements of the program.
3. Interactive Angles  
     
   The value of the angle between the branches drastically impacts the final appearance of the constructed tree. Make this program interactive so that the user can control the angle between the branches with the X-value of the mouse position. When the mouse is on the left side of the screen, the branch angle should be small. As the mouse moves across the screen to the right, the angle should widen. The map() function is definitely useful for this - make sure to see the video below.   
     
   <https://www.youtube.com/watch?v=7R2MUEHGqUE>
4. Varied Branch Thickness  
     
   Make the thickness (line weight) of the branches change based on the depth. Branches should be thicker the closer they are to the trunk. You can see this illustrated in the example videos below.

**Extra for Experts**

1. User choose leaf depth   
     
   Add functionality so that you user can use keyboard input to control at what levels leaves are drawn. 'z' should reduce the levels, down to the point where no Balloons are drawn at all, and 'x' should increase them, up to the point where every branch has a Balloon drawn on its end.
2. Fixed random colors and sizes  
     
   Creating the random balloon color and sizes in the base requirements is not complex, but an obvious problem presents itself; each time our draw loop repeats, new random values are chosen. This causes the size and color of the balloons to be changing every frame in a very distracting way. Fix this by devising a strategy to ensure that the colors and sizes are randomly selected, but that each from the exact same random values are used. *hint: you may want to read up on randomSeed() as this can be used in one of many ways to solve this problem.*

**Solutions**

Two short videos show completed samples: one of the base features version, and one of the extra for experts version.

<https://www.youtube.com/watch?v=wDOjWBxgThM> //base features  
 <https://www.youtube.com/watch?v=_I8Q-TGhAuw> //extra for experts

**Extra for Extra-Experts**

For those who are up for an even greater challenge, the next step would be to extend your program to build a recursive tree in 3D space.

In Processing, creating primitive shapes in 3D is made reasonably simple. A description of the basics can be found here: <https://processing.org/tutorials/p3d/>

**To complete the extra for extra-experts, try to add at one or more of the following features:**

* Basic tree production rules, but applied in 3D
* Random number of branches spawned at each branch terminal
* Variable (random) branch length
* Incorporation of light(s) into the scene

*Completing this extra challenge won't count for additional marks, but will give you a primer in 3D graphics (which may come in handy for your capstone coding project), and will demonstrate your programming prowess!*

