**DATA VISUALIZATION**

***Rishabh Chatterjee***

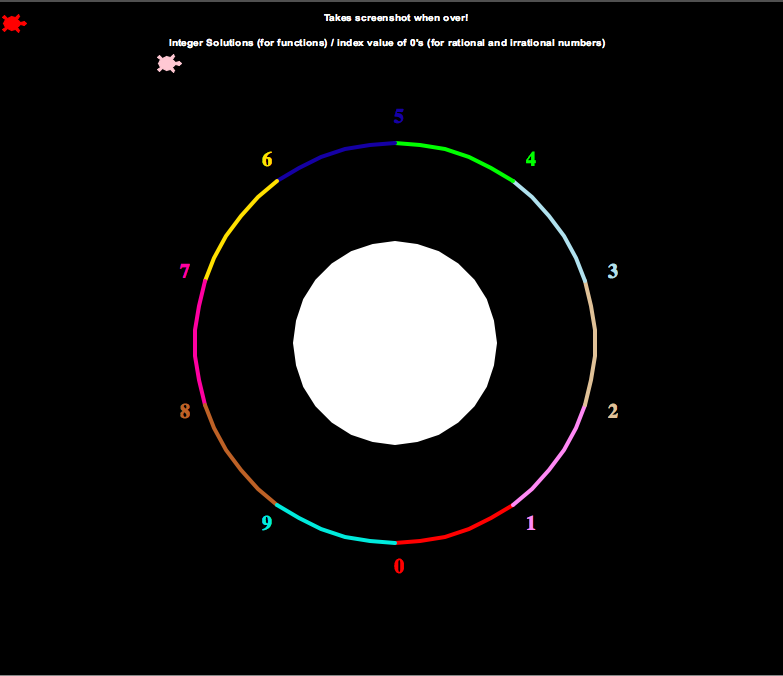
**Goal**

The goal of DataVisualization is help visualize rational and irrational numbers, and functions using three methods –scatterplot, gradient, and network. By creating interesting, and at times (as in the case of pi and some really cool functions) mesmerizing images from numbers, I wish to make numbers more ‘user-friendly’. We need to understand the beauty and significance of numbers rather than just seem them as simple digits.

**Approach**

The approach I am taking is to tackle this visualization issue from three sides simultaneously. Using turtle graphics and tkinter, I am first creating a list of values that need to be visualized (will be explained later) and then moving multiple turtles simultaneously to create the image ‘dynamically’ while traversing through the list of values.

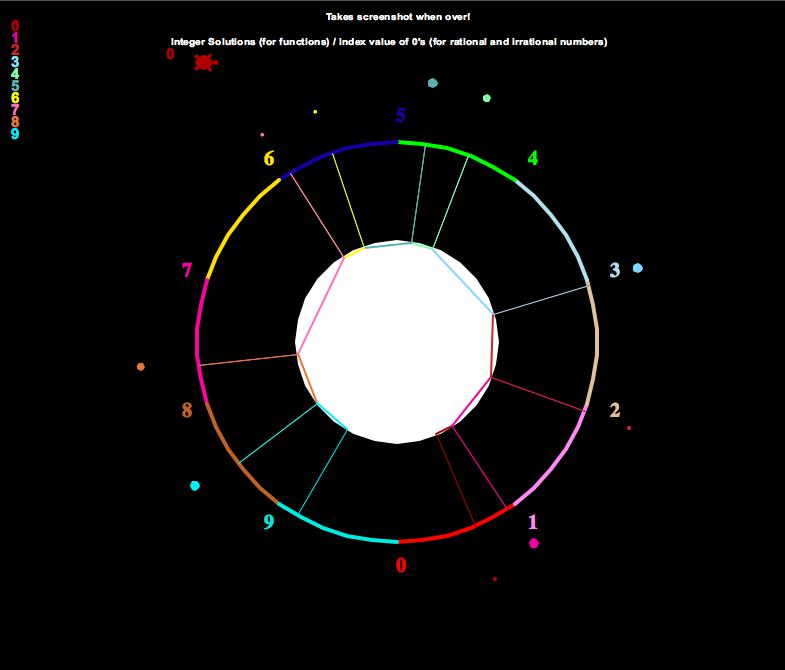
First, there is a circle that looks like this.



Each digit has a different color and a different allocated segment in the circle.

If the list has [3,1,5,3,2] the turtle moves (albeit following an indirect path as to create a network along with the gradient) from 3 to 1 to 5 to 3 to 2 etc. It is somewhat like pointers where one element points to the next. However, the exact point to where the lines are connected is randomly generated within the arc of the needed digit, and the position of the dots for the scatterplot are also randomly generated within a space behind the needed digit. There is also another turtle in the top center of the screen that keeps track of 0s and notes the x-values (solutions) for functions and index value of the 0s in a rational/irrational number/ modular mathematical function. Each time a ‘solution’ is encountered a beep sound (random sound out of 5 choices) is made to notify the user.

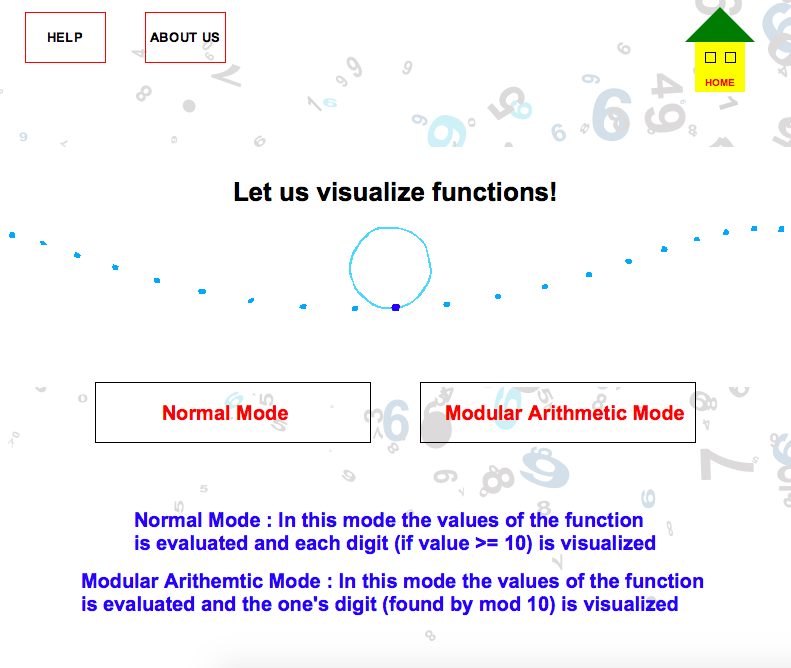
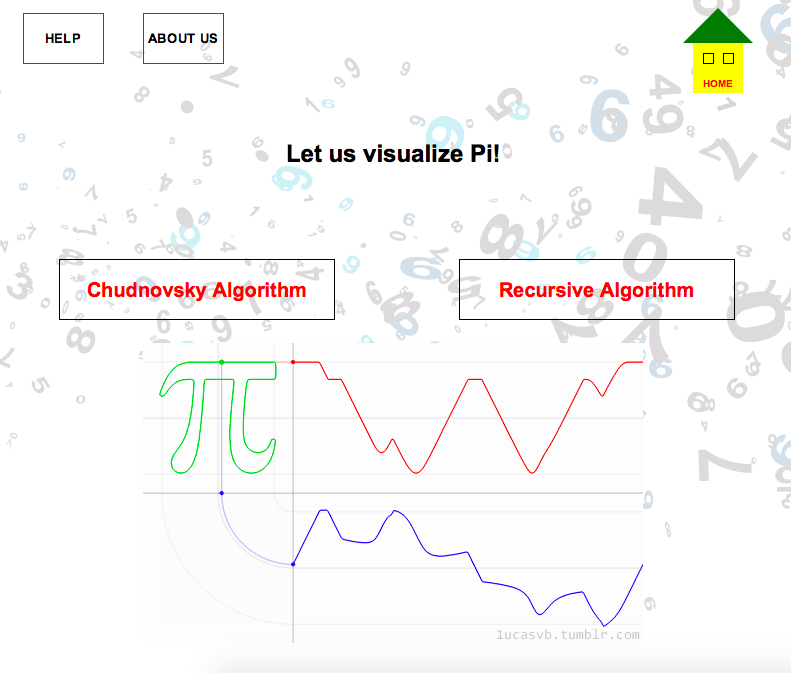
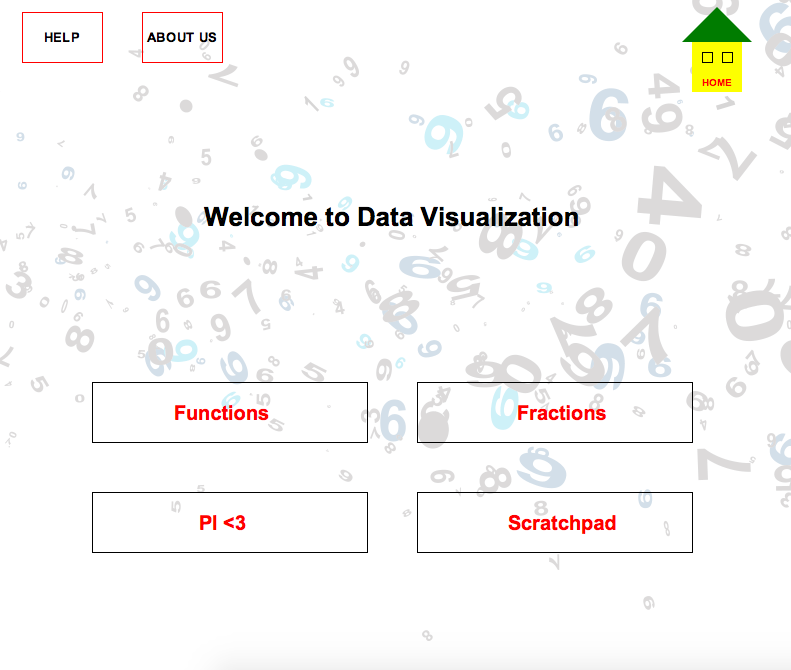
For example, this is plotting list(range(10)).

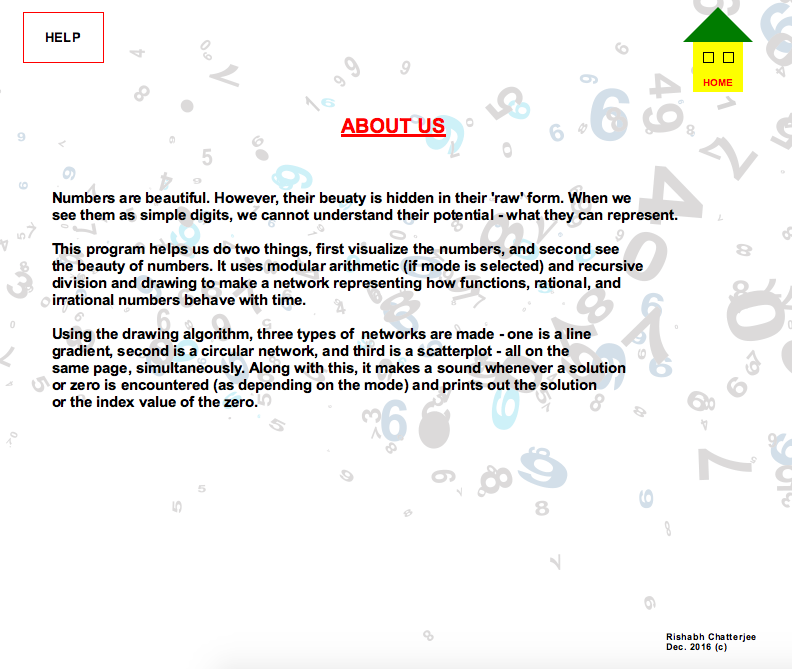
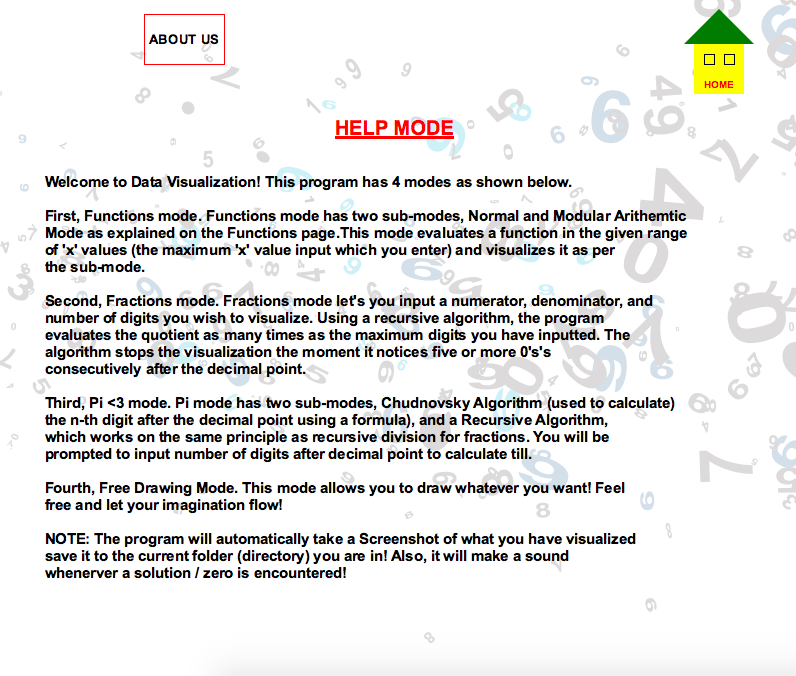
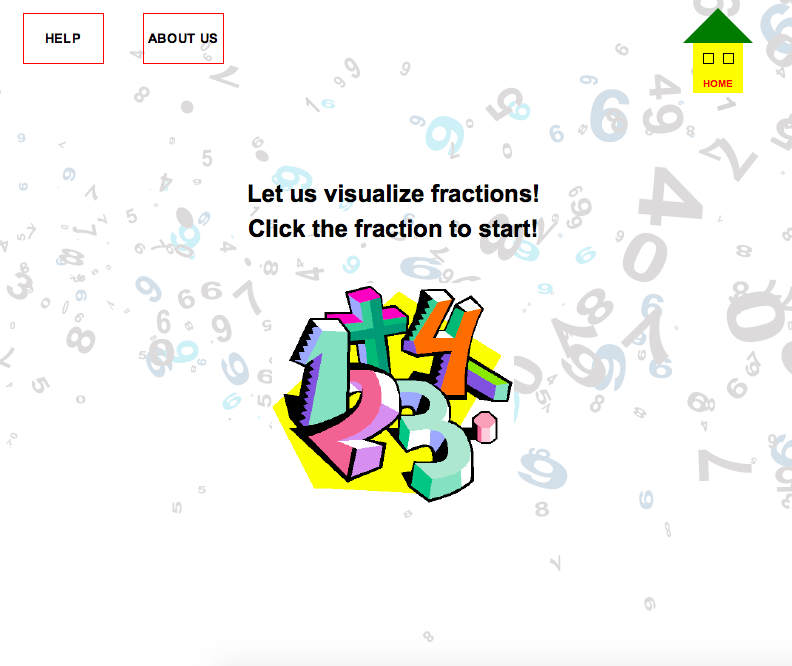


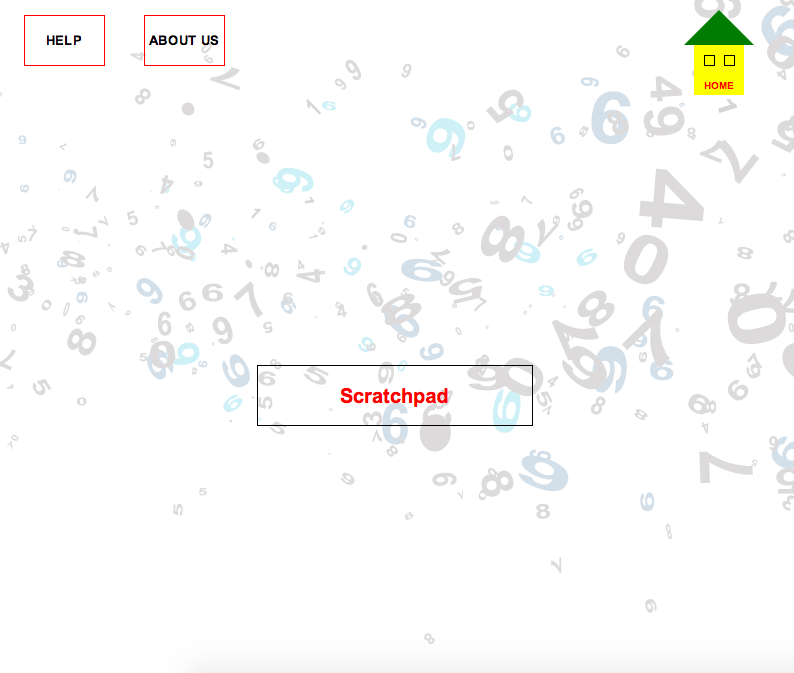
It is important to note that the network, and gradient and scatterplot work on two different algorithms although they are being generated simultaneously. The gradient of say the digit 2 is a combination of all colored lines from all other digits to 2. The network line of 2 is made is 2’s color. The scatterplot works similar to the gradient as in it forms a density scatterplot around 2’s area whenever a number points to 2. The density of the scatterplot and gradient around any digit can determine how many times that digit occurs (direct correlation).

Now that we have a broad knowledge of how it works (see video for more clarity), let’s see the UI.

The pages look like this.





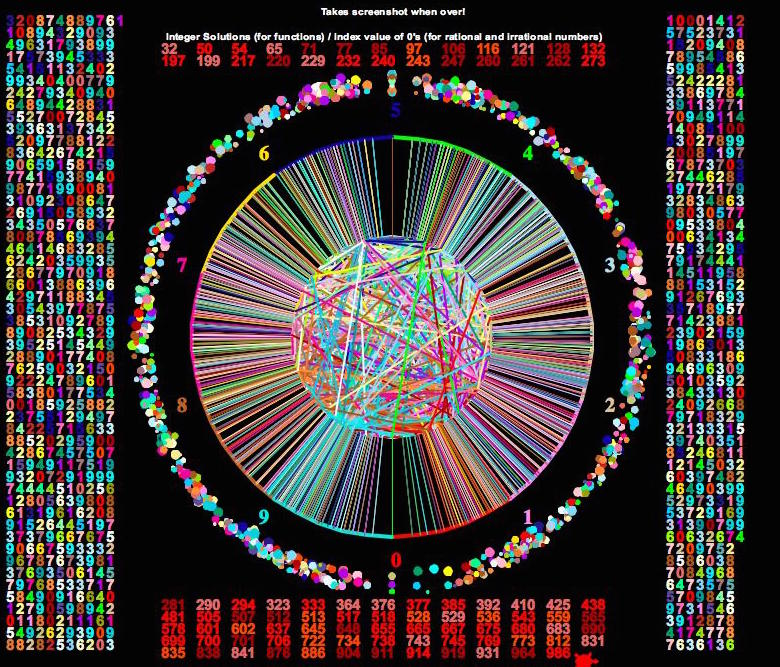


**Justification**

I chose this interface for the following reasons:

* The background is random numbers and that’s in sync with my goals
* I feel a light theme promotes more interest in the topic
* It is colorful and catches the eye
* A help screen is needed to guide the user
* An About Us page is needed to give the user a reason as to why this has been made
* I made mouse clicks over key presses because it is more user-friendly
* I have added a different image on each page to make it both look distinct and grab the attention of the user; also the images are relevant to each page
* At times I’ve also made the user click an image instead of a button for a functionality to bring diversity
* The background image has been kept same for consistency
* There is also a copyright © stamp on the bottom right page of the About Us since this is somewhat of a product
* Note when you are on the help page the help page button is not there to avoid redundancy, same in the About Us page
* There are popup boxes for input and in case of wrong input there are also a popup boxes asking for right input, for example in case of wrong input



A fully completed program looks something like this. This is the first 1000 digits of Pi.

**Some Usages**

I have divided my project into multiple files and am importing whatever I need form the required files so as to make the code easy to read. Along with this, the Fraction and Function part of the program uses OOP as these are classes. The major functions in the program are the:

* The draw function in the testingGeneratingNumbers file. This has multiple important functions inside it so as to avoid declaring global functions as they are used locally, for example for generating random points, drawing the numbers, the zeroes, etc.
* The division function in the fractions file. This is a recursive method which is used to calculate fractions as well as the recursive algorithm for Pi
* The findFunctionValues function that finds the values of the function depending on whether it is in normal or modular mode and returns the list to visualize.
* The play function which plays a .wav file whenever a 0 is encountered as explained before
* The freeDraw function which allows user to use a Scratchpad
* The piChudnovsky function which finds the nth digit of Pi

All other functions are also important but at the top of my mind these took more time to finish.