0.1 Function Class

Our basic Function class will contain two main methods:

- Parse - This method will convert the user's input into a data structure that we can use to evaluate. This data structure will be stored as a private attribute.
- Evaluate - This method will take a value of x and input it into our function and return the value $f(x)$.

Our class will contain more methods and attributes later (colour of the line, roots, turning points etc.) but these can be considered later as these are quite small parts. There are two distinct methods to parse a function:

- We can convert the input into a data structure, which we can then use to evaluate a value x.
- We can convert the input into its equivalant in a scripting language, such as ***Lua***, and then use the scripting language to evaluate a value x.

An example of the second is shown below:

```
import org.luaj.vm2.*;
import org.luaj.vm2.lib.jme.*;
public double evaluate(double x) {
        ScriptEngineManager mgr = new ScriptEngineManager();
        ScriptEngine e = mgr.getEngineByName("luaj");
        e.put("x", x);
        e.eval("y = math.sqrt(x)");
        return e.get("y");
}
```

The problem with using scripts is that they are take a huge hit on performance as you are effectively creating a virtual machine during your program. Especially in Java where a VM is used to run your programs, creating a VM inside a VM is not very effiecient. Also we may make upto 10000 evaluate calls to render multiple lines, and therefore we will need to be as effiecient as possible. While the first method is harder to implement, it will have better performance, and it will be easier to debug as we are not using external tools. Therefore we will use the first method.
There are two main structures we can convert a mathematical input into:

- Binary Trees
- Stack Based Programming

both of which we will analysis in detail.