In this activity, we will begin creating plots using python. I'll start by talking you through the parts of a python program, and then letting you create an actual plot. *Please do not copy/paste the code from here into your code. Copying from a browser or PDF file can introduce bugs that are hard to track down, and using your own fingers helps you to remember what you did.*

You will begin every Python program with

```
import numpy as np
import matplotlib.pyplot as plt
```

These two statements tell python to use numpy and matplotlib, and to call these `np` and `plt` respectively.

If you want to make a plot, then the next thing you need to do is create a variable for the horizontal coordinate. For instance, if we want to plot $\sin(x)$ on the range $0 \leq x \leq 2\pi$, we would:

```
x = np.linspace(0, 2*np.pi, 1000)
```

In this code, `np.linspace` is a function that creates a set of values with a lower and upper bound. The lower bound here is 0 and the upper bound is given as `2*np.pi`. Here `np.pi` is a constant that has the value π, and `*` represents multiplication. The number `1000` is the number of points to create, and you could use any integer here that you care for.

Once you have your independent variable, you are ready to create a plot:

```
plt.plot(x, np.sin(x))
plt.show()
```

The first line here creates the actual plot, and the second tells matplotlib to actually show the plot.

Your task

1. Create a plot of $f(x) = x^2$. Remember that you are encouraged to use web searches to find out how to do something in Python or with matplotlib.

2. Label your horizontal axis as x, and your vertical axis as x^2 or $f(x)$.

3. Create a second plot if $g(x) = \sqrt{x}$ on the same plot.

4. Add a legend that specifies which plot is which.

Extra challenge Create a legend indicating which line is which function. *There is an easy way and a hard way to do this, and sadly it is easier to discover the hard way to do it than the easy way.*

Fun challenge Create a plot of an elephant. Note that this is more of a mathematical than a programming challenge.

Pretty plots Change the colors and lines of your plot. Add symbols at each point (which will work best if you reduce the number of points in your `np.linspace`).