Exercise: Market share

Two new electric car companies (Volture and NeonDrive) are launching cars. Which will become dominant?

The cars launched by each company are very similar, with nothing technically or in terms of style or quality to choose between them, so at first people buy randomly. After a while, however, people are more likely to buy a brand if they know people who drive it or if they see it a lot on the roads.

1. Construct a model that uses random numbers

Let's combine the idea that a new buyer might know someone who owns a brand and might have seen more on the roads into a simple measure: the probabiltiy of someone buying a brand is based on the proportion of people who already own that brand. We'll act on the populaton of people who are buying a new car.

We are also making a number of assumptions ignoring factors such as other companies in the market, second-hand buying, and the effect of marketing.

2. Run this many times

For the first phase, let's have five people randomly choose either Volture or NeonDrive with equal probability. To do this we'll use `random`.

```
from random import random
```

We need to count how many own each brand.

```
Volture = 0
NeonDrive = 0
```

Now we can loop five times and use `random()` to decide whether to have this user buy one brand or the other.

```
for i in range(5):
    if random()<0.5:
        Volture += 1
    else:
        NeonDrive += 1
```

Finally, we can output the results.

```
print(Volture, NeonDrive)
```

If you run this a few times, you should get different splits among your first five users.

Now let's have word of mouth kick in, so people are more likely to buy the brand with the most owners. To do this, we will shift the probability of people joining Volture from 0.5 to the proportion of the current population that currently own Volture, i.e.

$$P(\text{Buys Volture}) = \frac{\text{No. Volture users}}{\text{No. Volture owners} + \text{No. NeonDrive owners}}.$$

In our code, this proportion is `Volture/(Volture+NeonDrive)`.

The loop is otherwise very like the one above. Let's use this to top our population up to 1000 owners.

```
for i in range(995):
    if random()<Volture/(Volture+NeonDrive):
        Volture += 1
    else:
        NeonDrive += 1
```

If we again print the counts, we can see how many people have bought each brand.

```
print(Volture, NeonDrive)
```

(Note: if you were so inclined, you could do this with one loop, testing for if we're in the first five iterations set the probability to `0.5` and otherwise set it to `Volture/(Volture+NeonDrive)`.)

3. Statistically analyse the results

Running this a few times, we can see different results. I ran my code and noticed some results where the initial distribution of users was 7:3 in favour of Volture and the end distribution was along these lines as 734:266. But of course, this is random and fluctuations do occur - one time I started with 7:3 in favour of Volture and ended up slightly in favour of NeonDrive at 491:509.

One thing we could do at this point is plot the development of our brands over time. To do this requires a different approach, since we must collect the number of owners of each brand at every iteration of the loop.

At the top of the file, tell Python you want to use MatPlotLib.

```
import matplotlib.pyplot as plt
```

Now before the loop we need lists to keep track of Volture, NeonDrive and the passage of time. We assume one user buys one car every time interval.

```
times = []
Volture_owners = []
NeonDrive_owners = []
```

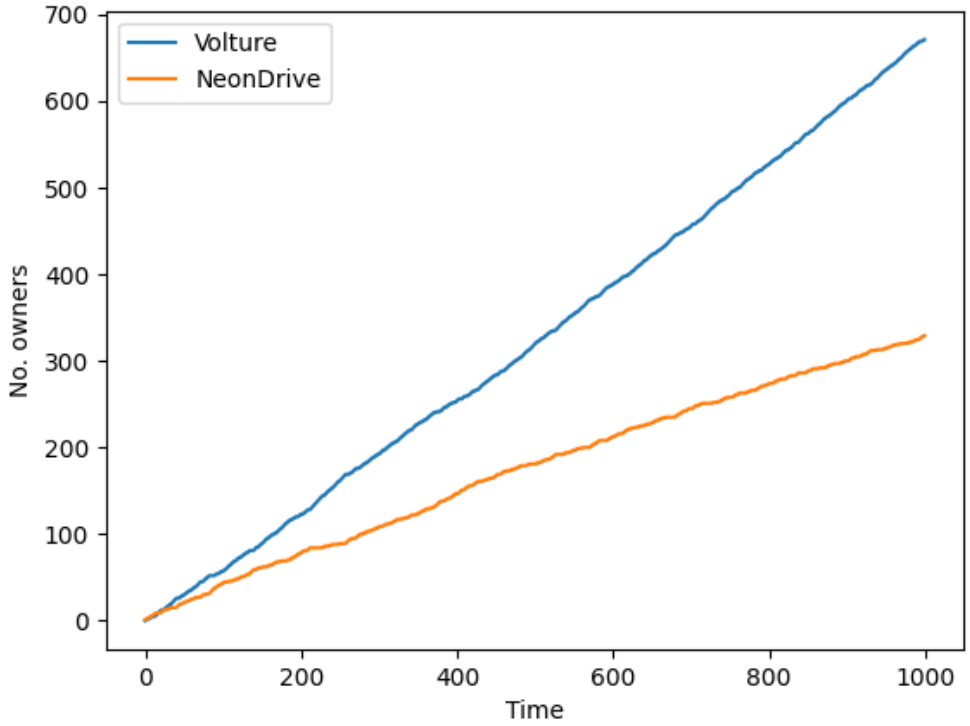
At the end of each iteration of both loops, we need to append the current values to our lists - the loop counter to `times` and the number of owners of each brand to its list.

```
times.append(i)
Volture_owners.append(Volture)
NeonDrive_owners.append(NeonDrive)
```

At the end of the program, you can plot `times` against `Volture_owners` and `times` against `NeonDrive_owners` on the same plot. We add labels to the axes, and `plt.legend()` is used to name the two lines of data.

```
plt.plot(times,Volture_owners)
plt.plot(times,NeonDrive_owners)
plt.xlabel('Time')
plt.ylabel('No. owners')
plt.legend(['Volture', 'NeonDrive'])
plt.show()
```

If you run this, you should get something like this - of course, your lines may look quite different to mine! This run of the simulation started with a 3:2 split among the first five users.



Rather than output and plot, we could run this simulation many times and count how many times each company comes out with the most owners. We would do this in a similar way to how we did with the coin toss exercise. Can you update your code to run this simulation a large number of times? It's best to remove (or comment out) the code that prints the numbers and plots the data each time if you're going to run it lots. Are you surprised by the result?