SECTION 14: ADVANCED PYTHON MODULES, 2 hours 23 minutes, 13 sections

5/13 Python Math and Random Modules

- -> math and random modules
- -> the math module holds functions
- -> the random module which contains random mathematical objects

- **-> in the .ipynb file**
    - import math
    - **help(math)**
        - **-> this returns the different functions in the maths module**

    - **to round numbers**
        - value = 4.35
        - **math.floor(value) <- round down**
        - **math.ceil(value) <- round up**
        - **round(4.35) <- this rounds to the nearest whole**
        - **round(4.5) <- this rounds up**
        - it rounds up because we either want it to round to all evens or all odds
            - **-> it's rounding towards the even numbers -> 4.5 is rounded to 4, but 5.5 goes to 6**
        - -> you can also from math import pi -> typing pi then prints out pi
        - -> math.e -> this prints e
        - -> math.inf, math.nan
        - -> math.e
        - -> math.log(math.e) <- this is 1
        - -> **math.log**
        - **-> math.log(100,10) <- what number do you have to raise 10 to to get to 100? -> this returns 2**
        - -> you can also math.sin(10)
        - -> another example is math.degrees(pi/2)
        - -> math.radians(180) <- this returns pi

    - **random numbers**
        - **-> how computers generate random numbers (pseudo random number generators)**
        - -> random seeds

        - **-> import random**
            - **random.randint(0,100) <- a random integer between 0 and 100**
            - **random.seed(101)**
                - **or 42 -> HH guide to the galaxy**

            - **random.randint(0,100)**
                - **-> this returns 74**
                - **-> the seed starts the same series of random integers**
                - **-> the same random numbers start repeating**
                - -> we're working with series of random integers
                - -> random.seed(101)
                    - it's a seed for any infinite set of random numbers

            - **taking a random item from a list**
                - mylist = list(range(0,20))

- **them random.choice(name_of_list) <- this returns a random item from the list**
  - **-> and once one of those items has been randomly called, then it won't be called again**

- **random.choices(population=mylist, k=10)**
  - **-> this returns an array**
  - **-> it's the population which we're picking from and the number of elements which we want randomly selected from it**
  - -> this is sampling with replacement

- **-> sampling without replacement**
  - random.sample(population=mylist, k=10)
    - -> none of those numbers are repeated

- **to shuffle a list**
  - **-> this sets the list equal to itself shuffled (the type of it is NoneType)**
  - **-> random.shuffle(mylist)**
  - -> this shuffles the list

- **you can select numbers according to a distribution**
  - random.uniform(a=0,b=100) <- all have an equal chance of being chosen
  - **random.gauss(mu=0,sigma=1) <- normal distribution centred at 0, with a standard deviation of 1 -> choosing random numbers according to the distribution**
    - -> for more information on this see the numpy library