**Assignment 1**

* Your solution will be a Python notebook that does the following basic computations.
* Use whichever modules you think are appropriate.
* Denote the solution to each part with proper markdown, e.g.:

# Part I: Question 1

code goes here

* To get credit for your answer, you must show in full the code that produced the answer.
* **Note:** Please test your programs with Python's 3.x Jupyter notebook. Some IDE's (such  
  as Spyder and PyCharm) may have slightly different behavior in some cases. Please note that Python 2 will not be accepted as valid run time environment.
* **Reminder:** Documentation is MUST! don't forget to add it at the beginning of each module/file and at the beginning of each function.
* Exercises need to be submitted through the course's moodle.

Part 1: Python and NumPy  
  
1. The bacteria *P. pythonicus* replicates every one hour, in a 100 ml tube. Being a very unfriendly bacteria, they reach stationary phase when there are 1,000,000 or more bacteria in the tube.

Write a program that will calculate the number of bacteria after one hour, two hours, etc, until stationarity is reached. The program will receive the starter size (number of bacteria to begin with), and start calculating from there. At each time point, the following message should be printed:  
< time > hours: < no. of bacteria > bacteria

2. Please solve the following problem using NumPy utilities.

1. Create a random array of integers of length 1000, x.
2. Replace in x numbers divisible by 3 and 4 by 12.
3. Convert x to a 2-dimensional array with 4 rows.
4. Create two random arrays of integers between 1 and 100, each of length 100, x1 and x2.
5. Go over NumPy's set routines, here: <https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.set.html>
6. Find the positions of elements in x1 and x2 that match (i.e. positions where the elements in the two arrays are the same).
7. Find the intersection of x1 and x2, and save it to a third array, x3.
8. Remove the intersection of x1 and x2 from both arrays, x1 and x2.

3. Write a program that will find numbers between 2000 and 3000 that are divisible by 7 but not by 5. Print all these numbers. Try to find a way to print the numbers in one line, separated by a comma.

Part 2: Pandas

* 1. Create a Panda's dataframe 20\*5, with random integers between 1 to 100.
  2. Multiply the second column by 2, using number index.
  3. Give your columns names: 'col1', 'col2' and so on.
  4. Multiply the third column by 3, using column names.
  5. Multiply entries with odd indices in column four by 5.
  6. Add a column to the dataframe with NaNs.Give this column a name, 'new'.
  7. For each line where the third column is divisible by 5, copy the value to the new column you created before. For lines that do no meet this criteria, set the value of the new column to 3.

1. Open the red wine data (accompanying file), winequality-red.csv. Make sure that you obtain a dataset consisting of 12 columns and 1599 lines. You might have to change default parameters before you obtain the desired table. Examine the features.   
   You should see the following: quality, fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol.
   1. Change to precision of the density column – rounding it to 3 precision digits, using apply or map. You can see apply examples here: <https://thispointer.com/pandas-apply-apply-a-function-to-each-row-column-in-dataframe/>  
      and map:  
      <https://www.programiz.com/python-programming/methods/built-in/map>
   2. Take the first 5 rows of the data and copy into a new object. Take the last 10 rows of the data and copy into a new object. Use Panda's concat to create another object with 15 rows. Print out the object.
   3. Find rows for which 'fixed acidity' is > 15.7 and print these out. How many are there?
   4. Create a function which, given a series, calculates the number of missing values. Apply the function to the columns of the data using apply. Now apply the function on the rows using apply again.
   5. Write a function that converts a fraction into a percentage. Apply it using apply to the citric acid column.
   6. Print out means for the columns.