# Introduction to Programming for Data Science Assignment 1

Dr Kieran Hughes

January 2024

## Instructions

This assignment is due by 11pm on Friday 23rd February. The answer to all your questions must be in a Jupyter workbook. Name your workbook as IntroProgAssignment1 - StudentID StudentNAME.ipynb. Label the start of each question and put any written answers in markdown cells. All answers should appear in order, so your answer to question 2 B should be below the answer to question 2 A and above the answer to question 2 C.

Include any libraries when you first require them. Ensure when you are uploading your work you only upload your own work and not any dataset. Just upload the ipynb file and CSV file requested. Do not zip anything. This is NOT a group assignment. You must complete the assignment individually. If you share your work, your marks will be divided. Some students may be asked to demonstrate their knowledge of the submitted work.

A Note on Libraries: For completing this assignment you are only to use basic Python libraries, mentioned either in the Assignment description, or ones used in class. These include numpy, random, pandas, Polynomial, matplotlib, seaborn and string. Other libraries are not permitted even if they will make the assignment easier. (you may not use nltk, Counter or others that have not been referred to in class or in the assignment).

Please note that you will be docked 5% for each day you are late with submission, in accordance with IT Sligo Marks and Standards.

## Questions

1. (a) Write a function called `factorial` (do not use math.factorial) that uses recursion to calculate the factorial of a non-negative integer input. Ensure there is error checking on the input. Note that 0! is defined to be 1.
   (b) Vectorise the function `factorial` by using the Numpy function `vectorize` and call this function `vectorised_factorial`.
   (c) Call the function `vectorised_factorial` passing it `range(20)`
   (d) Comment on the limitations of the function `vectorised_factorial`.

2. Let $S$ be the sum defined as

$$S = \sum_{k=0}^{\infty} \frac{k-1}{(k+1)!}, \tag{1}$$

   where $k!$ is the factorial of $k$.
   (a) Write a function called `Approximate_S` that takes a single non-negative integer argument $n$, and returns an approximation to $S$ based on the first $n$ terms of the sum in Equation (1). Use the convention that empty sums evaluate to 0 and the function `factorial` from Question 1.
   (b) Plot a graph, where `Approximate_S` is on the $y$-axis and $n$ is on the $x$-axis.
   (c) Assume that $S$ converges to a constant of the form $A + Be + Ce^2$, where $e$ is the Euler number (The base of the natural logarithm, which is approximately 2.718281828459) and $-10 \leq A, B, C \leq 10$. Use interactive sliders to plot $A + Be + Ce^2$ and use this plot to estimate the values of $A, B$ and $C$.

3. (a) Let $f$ be the invertible function from the alphabet to the integers modulo 26 defined by Table 1. Let $S_n$ be the function that shifts a letter n places forward in the alphabet. That is, $S_n$ is the function from the alphabet to the alphabet defined by $S_n(x) = f^{-1}(f(x) + n \pmod{26})$. For example $S_1(a) = b$ and $S_3(z) = c$.

   Write a function called `Text_Dynamics` that has a single string as input called `text`. The function should perform the following tasks in order:

   1. Remove all characters from `text` that are not in the English alphabet (either uppercase or lowercase)
   2. Convert all remaining characters to lowercase
   3. Remove any consecutive duplicates, so "ywadddfghhi" will go to "ywadfghi".
   4. Shift the characters based on their position in the string. The first character gets shifted up by 1, the second by 2 and so on. So for example "way" goes to $S_1(w)S_2(a)S_3(y) =$ "xcb". Call this text `shifted_text`.
   5. Repeat parts 3 (Remove any consecutive duplicates) and 4 (Shift the characters) until convergence, which is defined by the `shifted_text` being the same as any previous `shifted_text`.
   6. return the first repeated `shifted_text` if any and the `number_of_iterations`.

   (b) Write a function `Analyse_Text_Dynamics` that will generate a suitable number of random texts of particular lengths and analyse the output from `Text_Dynamics` on these texts.

   (c) Run `Analyse_Text_Dynamics` and comment on your results.

| $x$ | a | b | c | d | e | f | g | h | i | j | k | l | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $f(x)$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| $x$ | n | o | p | q | r | s | t | u | v | w | x | y | z |
| $f(x)$ | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

Table 1: Definition of the function $f$ from the alphabet to the integers modulo 26

4. (a) Write a function called `is_broadcast_possible` that has 4 positive integers as arguments, namely `row_sizeA, col_sizeA, row_sizeB` and `col_sizeB`. The function should, using the inputs as dimensions, generate matrices A and B with random integer entries. The function should try to perform the broadcast operation A*B and return a Boolean depending on whether the broadcast operation is possible.

   (b) Write a function called `Number_of_Broadcasts_Possible` that has 1 positive integer argument $n$. This function should, for all possible values of `row_sizeA, col_sizeA, row_sizeB` and `col_sizeB` from 1 to $n$ inclusive (there are $n^4$ possible values), check if the broadcast is possible and return the count of the number possible, that is, the count on the number of times that `is_broadcast_possible` returns True.

   (c) The output of `Number_of_Broadcasts_Possible` is a function in $n$. Perform a quadratic regression on this function by using the NumPy function `polyfit`. Create a poly1d object called `quadratic_fit` of the result of the quadratic regression, by using the Polynomial class from the numpy.polynomial module in NumPy (from numpy.polynomial import Polynomial).

   (d) Illustrate the output from `Number_of_Broadcasts_Possible` and `quadratic_fit` using the same Matplotlib plot and comment on the plot.

5. Use the pandas DataFrame to work with the iris dataset which is provided as a CSV file on the Moodle page. Ensure you use this file and don't get iris from elsewhere. This dataset is frequently used for Machine Learning clustering and classification algorithms.

   (a) Read iris.csv into Python as a pandas DataFrame. Note that the CSV file includes column headers. Answer these questions in the workbook: How many data points are there in this data set? What are the data types of the columns? What are the column names? The column names correspond to flower species names, as well as four basic measurements one can make of a flower: the width and

length of its petals and the width and length of its sepal (the part of the pant that supports and protects the flower itself). How many species of flower are included in the data?

(b) It is known that the dataset contains errors in 2 of the rows. Using 1-indexing, these are in the 35th and 38th rows. The 35th row should read 4.9, 3.1, 1.5, 0.2, "Iris-setosa", where the fourth feature is incorrect as it appears in the file, and the 38th row should read 4.9, 3.6, 1.4, 0.1, "Iris-setosa", where the second and third features are incorrect as they appear in the file. Display the values for just these rows before and after correcting the values.

(c) It is useful to have two additional features sometimes called Petal Ratio and Sepal ratio defined as the ratio of the petal length to petal width and the ratio of the sepal length to sepal width, respectively. Add two columns to your DataFrame corresponding to these two new features. Name these columns PetalRatio and SepalRatio, respectively and then print the first few rows of the DataFrame.

(d) Save your corrected and extended DataFrame to a csv file called iriscorrected.csv. Please include this file in your submission. (You can upload multiple files to an assignment on Moodle. Do not zip them together.

(e) Use a pandas aggregate operation to determine the mean, median, minimum, maximum and standard deviation of the petal and sepal ratio for each of the three species in the data set. Note: you should be able to get all five numbers in a single table (indeed, in a single line of code) using a well-chosen group-by or aggregate operation. Display these statistics using the print function.