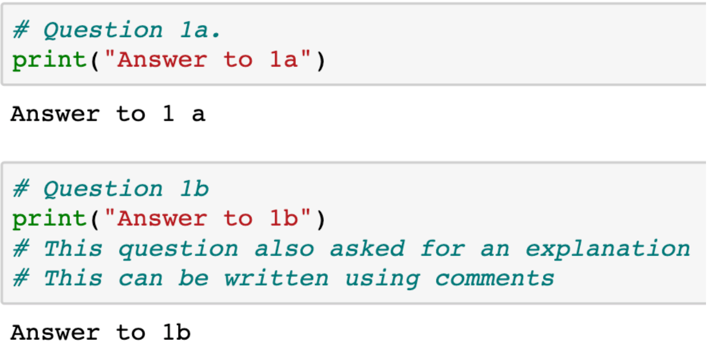
Computational Science
ACM20030
Assignment 1

This assignment is worth 5% of your total mark.
The assignment is marked out of 30.
Question 0: 2 marks
Question 1: 5 marks
Question 2: 13 marks
Question 3: 10 marks

Remember to add comments throughout your notebook using the ‘#’ symbol. You can use these to number questions in the notebook and provide written answers if asked, e.g.,



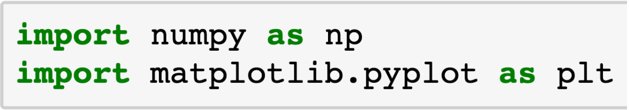
```
# Question 1a.
print("Answer to 1a")
```

```
Answer to 1 a
```

```
# Question 1b
print("Answer to 1b")
# This question also asked for an explanation
# This can be written using comments
```

```
Answer to 1b
```

0) Create a new Python-notebook in your ACM20030-Assignments folder called ACM20030-Assignment-1. In the first cell of the notebook write.



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

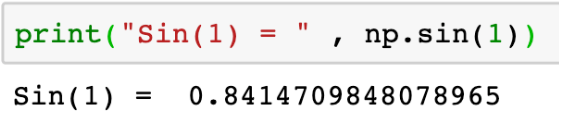
Execute this using shift+enter. Save the notebook. Commit the notebook to your git repository and push it to GitHub. Log into GitHub in your web browser, go to your ACM20030 repository and check that the ACM20030-Assignment-1.ipynb file appears online.

To commit, click ‘commit’ on SourceTree, select the file you want to add, type a ‘commit message’ and the click ‘commit’ in the bottom right.

1) a) Write a function with signature VectorLength(x, y) that computes the length of a vector, v with components in cartesian coordinates (x, y). i.e., $|v| = \sqrt{x^2 + y^2}$ Use NumPy for the square root function.

b) Using the VectorLength(x, y) function **print** the length of vectors with components v = (5, 6) and v = (-1, 5)

c) Modify this command



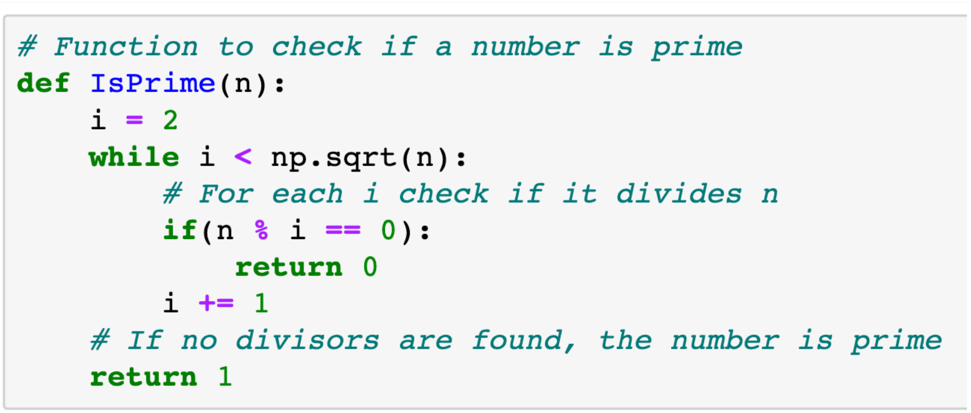
```
print("Sin(1) = " , np.sin(1))
```

```
Sin(1) =  0.8414709848078965
```

to print the result to only three decimal places

d) Commit your updated notebook with a suitable commit message. Push your changes to GitHub.

2) The following function claims to check if a number is prime



```
# Function to check if a number is prime
def IsPrime(n):
    i = 2
    while i < np.sqrt(n):
        # For each i check if it divides n
        if(n % i == 0):
            return 0
        i += 1
    # If no divisors are found, the number is prime
    return 1
```

Copy it in to your Python Jupyter notebook.

a) Does the function work correctly? Check the values from 1 to 10

b) At the start of the function add an if-statement to correct the n=1 case

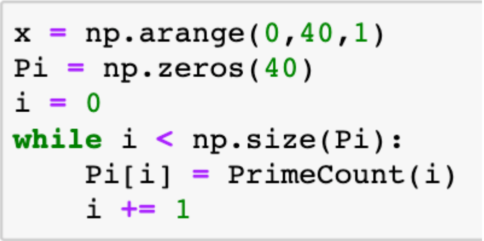
c) By putting a print statement inside the while loop, work out why the function returns, e.g., 1 for n=9. Fix the function to make it work. Write a comment that describes the fix.

d) Commit the code using the message “Fixed IsPrime function” and push the fixed code to GitHub.

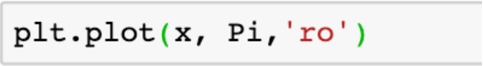
e) Using the corrected IsPrime( ) function, write a loop that calculates the number of prime numbers less than 1000. Print the total number of primes less than 1000

f) The prime counting function $\pi(x)$ is defined to be the number of primes less than or equal to x. By using your answer to part e) write a function with signature PrimeCount(x) that computes $\pi(x)$. You can check your function against the results in the Table on http://mathworld.wolfram.com/PrimeCountingFunction.html. Print the values of PrimeCount(1000) and PrimeCount(10000)

g) The following code will plot the PrimeCounting function for values less than 40



```
x = np.arange(0,40,1)
Pi = np.zeros(40)
i = 0
while i < np.size(Pi):
    Pi[i] = PrimeCount(i)
    i += 1
```



```
plt.plot(x, Pi,'ro')
```

copy it into your notebook and add a grid, an x-label saying ‘x’, a y-label saying ‘Pi(x)’

h) Commit and push your code to GitHub.

Bonus [not marked]: write the answer to part (g) without using a loop. hint: look up np.vectorize( ).

3) a) Download the file PowerLawOrExponential.txt from BrightSpace and save it into your ACM20030-Assignments folder. Commit this file to the repository and push the changes to GitHub.

b) Load the data using the np.loadtxt( ) function. Define an ‘x’ variable using the data in the first column, and a ‘y’ variable usage the data in the second column.

c) Make two plots showing the data on log and log-log scales. Make sure both plots are labelled and have a grid.

d) Is the data a discrete representation of the x^n (power law) or e^{nx} ? State why you have given the answer you have.

e) By examining the plots, determine the value of n.

f) Commit and push your code to GitHub.

4) To submit your assignment you should zip the ACM20030-Assignments folder. Rename the zip file ACM20030-Assignments-STUDENTNUMBER.zip where STUDENTNUMBER is your student number. Upload the zip file to BrightSpace.

5) Still looking for ways to improve your Python coding? Try the problems at projecteuler.net/archives or try to run some of the examples from matplotlib (matplotlib.org/gallery)