A0 - Area of a Circle

You now understand that Python statements can be algebraic equations. Use the IDLE text editor (File → New File) to write your first Python program. A template is provided below.

Assignment Task: Write a function that accepts the radius of a circle as input and returns its area. Your function should only return the area and nothing else. No printing inside the function and no returning a string. Do not change the name of the function.

After you are satisfied that your function works as expected, save the file (Eg. *area_circle.py* (any name will do) *)*. Submit the program through Autolab against the appropriate Assignment. Login to Autolab through IRIS. Autolab will run your program against a few inputs (some are shown in the Table below) and check if the outputs generated are as expected.

Go through the Autolab output and try to make sense of the output. The input(s) given to your function, the returned output and the expected output should be shown in the output.

Area of a Circle Template (area_circle.py)

```
'''
CS101. A0 - Area of a Circle
'''

def area_circle(radius):
       """Calculate and return the area of a circle

       :param radius: float - radius of the circle.
       :return: area of the circle.

       Function to calculate the area of a circle.
       Assume the value of pi to be 3.141593
       """

       pass
```

Test Cases

Function	Inputs	Output	Remarks
area_circle	1	3.141593	
area_circle	1.1	3.8013275300000005	
area_circle	0.1	0.03141593	
area_circle	10.1	320.47390192999995	

****

A0a - Square Root

The same Autolab submission and evaluation process is followed for all the assignment / quiz / assignment programs throughout the Semester.

Assignment Task: Use an algebraic equation to compute the square root. Write a function that accepts a positive number as input and returns its square root. Write the appropriate docstrings.

Square root Template (square_root.py)

```
'''
CS101. A0a - Square root program
'''

def square_root(n):
	"""Calculate and return the square root of a positive integer

	:param n: int positive number.
	:return: square root of n.

	Function that takes in a positive integer, calculates its square root
	using a simple algebraic equation. Returns the result.
	"""

	pass
```

Test Cases

Function	Inputs	Output	Remarks
square_root	100	10.0	
square_root	1	1.0	
square_root	2	1.4142135623730951	
square_root	17	4.123105625617661	
square_root	1024	32.0	
square_root	6241	79.0	

A0b - Energy Mass Equation

Task: Calculate the equivalent energy in Joules of a given mass as input to the emc2 function. Assume light travels at 299792458 m/s. Write the appropriate docstrings.

Energy Mass Equivalence Template (emc2.py)

```
'''
CS101. A0b - Energy Mass Equivalence
'''

def emc2(m):
	pass
```

Test Cases

Function	Inputs	Output	Remarks
emc2	1.0	8.987551787368176e+16	
emc2	1.4142135623730951	1.2710317630226627e+17	
emc2	0.0001	8987551787368.176	

****

A0c - Acceleration

Task: Calculate the acceleration (rate of change in velocity) in m/s^2. Inputs: initial velocity (u), final velocity (v) and time taken, t, in seconds to reach velocity v m/s starting from u m/s.

Acceleration Template (acceleration.py)

```
'''
CS101. A0c - Acceleration
'''

def acceleration(u, v, t):
      pass
```

Test Cases

Function	Inputs: u, v, t	Output	Remarks
acceleration	1.0, 1.0, 10	0.0	
acceleration	1.0, 2.0, 10	0.1	
acceleration	1.0, 10.0, 1.0	9.0	
acceleration	1.0, 1.0001, 0.0001	0.9999999999998899	

****

A0d - Gravitational Force Equation

Task: Calculate the gravitational force exerted by two bodies of masses m1 and m2 separated by a distance of r. Inputs: m1, m2, r. Assume Gravitational constant is $6.6743 \times 10^{-11}\ m^3\ kg^{-1}\ s^{-2}$.

Gravitational Force Template (gravity.py)

```
'''
CS101. A0d - Gravitational Force
'''

def gforce(m1, m2, r):
      pass
```

Test Cases

Function	Inputs: u, v, t	Output	
gforce	1.0, 1.0, 10	6.674299999999999e-13	
gforce	75.0, 75.0, 1.0	3.75429375e-07	
gforce	5.97219e+24, 7.34767309e+22, 384400	1.9820856039721175e+26	
gforce	5.97219e+24, 1.989e+30, 150750000	3.4886667104109302e+28	

****