**Assignment #05(PYTHON)**

**Submitted to Sir Jawad**

**Andleeb Ghazal Reg# 399697**

**Q1:Draw Circle using turtle:**

import turtle

x=turtle.Turtle()

def square(angle):

x.forward(5)

x.right(angle)

x.forward(5)

x.right(angle)

x.forward(5)

x.right(angle)

x.forward(5)

x.right(angle+5)

for i in range(40):

square(1)

**Q2:Python Math Library:**

**Function Description**

ceil(x) Returns the smallest integer greater than or equal to x.

copysign(x, y) Returns x with the sign of y

fabs(x) Returns the absolute value of x

factorial(x) Returns the factorial of x

fmod(x, y) Returns the remainder when x is divided by y)

fsum(iterable) Returns an accurate floating point sum of values in the iterab

modf(x) Returns the fractional and integer parts of x

trunc(x) Returns the truncated integer value

pow(x, y) Returns x raised to the power y

sqrt(x) Returns the square root of x

acos(x) Returns the arc cosine of x

asin(x) Returns the arc sine of x

Following are several Python math libraries –

1. Math

this library is adequate when you have to carry out basic mathematical operations.

For example: You can carry out the exponential of 3 using the exp() function of python math library as follows:

>>> from math import exp

>>> exp(3) #Calculates Exponential

2. Numpy

The numpy library in Python is most widely used for carrying out mathematical operations that involve matrices. For example, implement the dot product of two matrices as follows –

>>> import numpy as np

>>> mat1 = np.array([[1,2],[3,4]])

>>> mat2 = np.array([[5,6],[7,8]])

>>> np.dot(mat1,mat2)

array([[19, 22],

[43, 50]])

3. SciPy

This python math library provides all the scientific tools for Python. It contains various models for mathematical optimization, linear algebra, Fourier Transforms, etc. For example; linalg() function use from the SciPy library to calculate the determinant of a square matrix.

>>> from scipy import linalg

>>> import numpy as np

>>> mat1 = np.array([[1,2],[3,4]]) #DataFlair

>>> linalg.det(mat1)

-2.0

4. Scikit-learn

Machine Learning is an important Mathematical aspect of Data Science. Using the various machine learning tools, data can easily be classify and outcomes can be predict. Scikit-learn offers various functions to facilitate easy classification, regression, and clustering techniques.

>>> from sklearn import linear\_model

>>> regress = linear\_model.LinearRegression()

>>> regress.fit([[0,0],[1,1],[2,2]], [0,1,2])

LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)

>>> regress.coef\_

array([0.5, 0.5])

There are many more libraries for Mathematical operations in Python and many more are under development.

**Q3: Data.Shape and Data.dype function**

shape method returns the number of rows and columns as the result.

data type object (dtype) informs about the layout of the array like Type of the data (integer, float, Python object, etc.) Size of the data (number of bytes).

People also ask

Import numpy as np

Data1=np.array([1,2,3,4])

Data2 = np.array([4, 5, 8, 4],[6, 3, 2, 1],[8, 6, 4, 3])

print(data.shape)

print(data.dtype)

**output:**

for Data1

int32 and 4

for Data2

TypeError: array() takes from 1 to 2 positional arguments but 3 were given

**Q4: Numpy function**

Creating ndarrays

• data = [6, 7.5, 8, 0, 1]

• arr1 = np.array(data)

• print(arr1)

• Output

• [ 6. 7.5 8. 0. 1.]

Creating ndarrays

• data1 = [6, 7.5, 8, 0, 1]

• arr1 = np.array(data1)

• print(arr1)

print (arr1.ndim)

print(arr1.shape)

• Output

• [ 6. 7.5 8. 0. 1.]

1

5

Operations between arrays and

Scalars

fFunction Output

arr = np.array([1., 2., 3.]) [1,2,3]

• print arr [1,4,9]

• print arr \* arr [0,0,0]

• print arr – arr [1,0.5,0.3333]

• print 1 / arr

• print arr \*\* 0.5 [1,1.4142,1.7321]