**ASSIGNMENT 1**

**1 .What makes NumPy.shape() different from NumPy.size()?**

**Ans**: Some numpy operations take an argument called shape, such as np.zeros, whereas some others take an argument called size, such as np.random.randint. To me, those arguments have the same function and the fact that they have different names is a bit confusing. Actually, size seems a bit off since it really specifies the .shape of the output. shape() is used to get complete structural shape of our 2D array. For example (3,4) . numpy. size() will give us how many elements are present in total.

**2** .**In NumPy, describe the idea of broadcasting.**

**Ans:** The term broadcasting refers to the ability of NumPy to treat arrays of different shapes during arithmetic operations. Arithmetic operations on arrays are usually done on corresponding elements. If two arrays are of exactly the same shape, then these operations are smoothly performed.

**Example :**

import numpy as np

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

b = np.array([10,20,30,40])

c = a \* b

print c

**Output:**

[10 40 90 160]

**3. What makes Python better than other libraries for numerical computation?**

## ****Ans****: One of the key features of Python is its numerous libraries and packages. In this article, we will list down the popular packages and libraries in Python that are being widely used for numeric and scientific applications.

1)Scipy(scientific numeric library):

 The library consists of modules for optimization, image processing, FFT, special functions and signal processing. The SciPy package includes algorithms and functions which are the crux of Python scientific computing capabilities.

2)Pandas(data analytics library):

Pandas is the most important data analysis library of Python. Being open source, it is used for analysing data with Python. It can take data formats of CSV or TSV files, or a SQL database and convert it into Python data frames with rows and columns which is similar to tables in statistical formats. The package makes comparisons with dictionaries with the aid of ‘for’ loops which are very easy to understand and operate.

3)IPython(command shell):

IPython is a command shell which is designed for interactive calculation in various programming languages. It offers self-examination, rich media, shell syntax, tab completion, and history.

4)Numeric Python(fundamental numeric package):

Better known as Numpy, numeric Python has developed a module for Python, mostly written in C.  Numpy guarantees swift execution as it is accumulated with mathematical and numerical functions.

**4. How does NumPy deal with files?**

**Ans**:NumPy introduces a simple file format for ndarray objects. This **.**npy file stores data, shape, dtype and other information required to reconstruct the ndarray in a disk file such that the array is correctly retrieved even if the file is on another machine with different architecture.

numpy.save()

The numpy.save() file stores the input array in a disk file with npy extension.

import numpy as np

a = np.array([1,2,3,4,5])

np.save('outfile',a)

To reconstruct array from outfile.npy, use load() function.

import numpy as np

b = np.load('outfile.npy')

print b

It will produce the following output −

array([1, 2, 3, 4, 5])

**5. Mention the importance of NumPy.empty().**

**Ans:** The numpy module of Python provides a function called **numpy.empty()**. This function is used to create an array without initializing the entries of given shape and type.

Just like **numpy.zeros()**, the **numpy.empty()** function doesn't set the array values to zero, and it is quite faster than the **numpy.zeros()**. This function requires the user to set all the values in the array manually and should be used with caution.

### Syntax

numpy.empty(shape, dtype=float, order='C')

### Parameters:

**shape: int or tuple of ints**

This parameter defines the shape of the empty array, such as (3, 2) or (3, 3).