Numpy part 1

```
# import the module
import numpy as np

'''
    [1] Numpy: stands for numerical python
    [2] is a python library used for working with arrays
    [3] is also has a functions for working in domain of linear algebra, fourier
transform, and matrices
    [4] Numpy is up to 50X faster than traditional python lists
    [5] the array object in Numpy is called "ndarray"
'''

# create new list
lst = [1, 2, 3, 4]
print(lst)          # [1, 2, 3, 4]

# create 1D array
arr1 = np.array(lst)
print(arr1)         # [1 2 3 4]

arr2 = np.array(["A", "B"])
print(arr2)         # ['A' 'B']

# array elements can have different data types
# if string exist will convert the numbers to string
arr3 = np.array([1, "Osama", 120.2])
print(arr3)         # ['1' 'Osama' '120.2']

# if the array elements is numbers, if floating point number exist, will convert
numbers to float
arr4 = np.array([11, 21, 10, 10.2])
print(arr4)         # [11.  21.  10.  10.2]

# get the type of array element
print(arr1.dtype)       # int64

# get the shape of the array
# shape: get the form of array element
# syntax (rows, cols)
print(arr1.shape)       # (4,)

# get the size of array elements
print(arr2.size)            # 2

# determine the dimensions of array
# check if the array is 1D, 2D, multidimension
print(arr1.ndim)        # 1

# create 2D array
arr5 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(arr5)
```

```
'''
Output:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
'''

# get the number of dimensions
print(arr5.ndim)        # 2

# get the shape of the array
print(arr5.shape)       # (3, 3)

# get the size of array
print(arr5.size)        # 9

# get the type of object
print(type(arr5))           # <class 'numpy.ndarray'>

# get the type of the data that exist on object
print(arr5.dtype)       # int64

# print the "π" value
pi = np.pi
print(pi)       # 3.141592653589793

# get the Trigonometry
x = np.sin(0)
print(x)        # 0.0

y = np.cos(0)
print(y)        # 1.0

z = np.array([1, 2, 3, 0])
a = np.cos(z)
print(a)        # [ 0.54030231 -0.41614684 -0.9899925   1. ]

# create 2D array
# create array from a list of lists
array = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(array)
''''
[[1 2 3]
 [4 5 6]
 [7 8 9]]
'''

# get the shape of the array
# syntax (rows, cols)
print(array.shape)      # (3, 3)

# get the number of dimensions of array
print(array.ndim)       # 2

# get the size of array
# syntax: (rows * cols)
print(array.size)       # 9
```

```
# get the type of elements on the array
print(array.dtype)      # int64

array2 = np.array(["Hello", "World"], dtype="str")
print(array2)           # ['Hello' 'World']

# get the shape of array element
print(array2.shape)     # (2,)

# get the type of array elements
print(type(array2))     # <class 'numpy.ndarray'>

array3 = np.array([1, 2, 2.2])

# get the type of array elements
print(array3.dtype)     # float64

# get the type of array object
print(type(array3))     # <class 'numpy.ndarray'>

# ====================================
# == Converting Elements Data Types ==
# ====================================

# create an array with floating point number elements
x = np.array([1.1, 2.2, 3.3])
print(x)        #    [1.1 2.2 3.3]

# change the type of the array elements
x = np.array([1.1, 2.2, 3.3], np.int64)
print(x)        # [1 2 3]

# get the type of array elements
print(x.dtype)      # int64

# =====================
# == Saving An Array ==
# =====================
# Save an array to a binary file in NumPy ".npy" format.
# save the array in current director
x = np.array(["Hello", "World"])
np.save("array", x)

# call the array
y = np.load("array.npy")
print(y)        # ['Hello' 'World']

# =====================
# == Types of Arrays ==
# =====================
# 1- scalar: this array contain only one row, one column like: [10], (1*1)
# 2- vector: a vector consists of group of scalars (m*1)
# 3- Matrix: a matrix is a collection of vectors (m*n)
# 4- Tensor: scalar, vector, and matrix is a tensors but in a different dimensions

# Rank 0: refers to the scalar
# Rank 1: refers to the vector
# Rank 2: refers to the matrix
# Rank 3: the dimensions expressed by (k*m*n)
# Rank 3: also called a group of matrices

# k: refers to the number of dimensions
# m: refers to the number of rows
# n: refers to the number of columns


# Rank 0
a0 = 5
print(a0)       # 5

# Rank 1
a1 = np.array([1, 2, 3, 4])
print(a1)       # [1 2 3 4]

# Rank 2
a2 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(a2)
'''
[1 2 3]
 [4 5 6]
 [7 8 9]]
'''

# Rank 3
a3 = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
print(a3)
'''
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
'''
print(a3.shape)     # (2, 2, 3)
print(a3.ndim)      # 3
print(a3.size)      # 12

arr = np.array([1, 2, 3, 4], ndmin=5)
```