Data Handling
Numpy

Scanned with CamScanner

Introduction

Python is a flexible, easy, general-purpose programming language on its own.

With help of few popular libraries (such as **NumPy**, **scipy**, **matplotlib**) it becomes powerful environment for scientific computing and is being extensively used for Data Sciences.

Scanned with CamScanner

What is NumPy?

NumPy is a popular Python library used for scientific computing applications and is acronym for "**Numerical Python**".

It is an open source module of Python that provides functions for fast mathematical computation on arrays and matrices.

It provides a high performance multi-dimensional array objects and tools for working with these arrays.

Scanned with CamScanner

Installing NumPy

Step 1: Open command prompt

type cmd and press enter

Step 2: c:\Users\HCL>

Step 3: cd appdata\local\programs\python\python37-32 \scripts

Step 4: pip install numpy

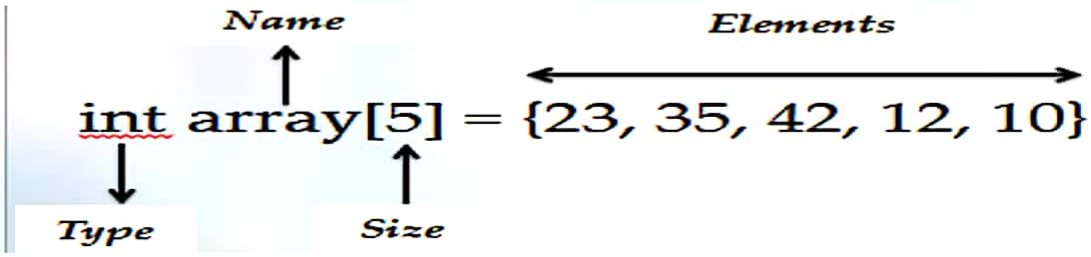
Scanned with CamScanner

Array

Array

An array is a container which can hold fixed number of items and these items should be of the same type.

Array representation



elements	23	35	42	12	10
index	0	1	2	3	4

Scanned with CamScanner

It is known as vector.

The number of subscript/index determines t
dimension of the array.

num[0]	num[1]	num[2]	num[3]	num[4]
2	5	7	4	6
Element 1	Element 2	Element 3	Element 4	Element 5

fig: 1D Representation in memory

Scanned with CamScanner

columns

rows	0	1	2	3	
0	55	12	32	25	num[0][0]
1	22	19	17	75	num[1][3]
2	65	54	27	43	num[2][2]

num[3,4]

fig: 2D Representation in memory

Scanned with CamScanner

Creating Arrays

To create a NumPy array, we can simply pass a python list to the array method.

Create one dimensional NumPy array

```
import numpy as np
Lst=[1,3,5,7]
arr1=np.array(Lst)
print(arr1)
```

Output: [1 3 5]

Scanned with CamScanner

Creating Arrays

Create two dimensional NumPy array

```
import numpy as np
Lst=([1,3,5,7],[2,4,6,8])
arr2=np.array(Lst)
print(arr2)
```

Output:

```
[[1 3 5 7]
 [2 4 6 8]]
```

Scanned with CamScanner

Accessing individual elements of an Array

For 1D array **<array-name>[<index>]**
e.g. arr[3]

e.g.

```
import numpy as np
Lst=[1,3,5,7]
arr1=np.array(Lst)
print(arr1[0])
print(arr1[2])
print(arr1.shape)
```

Output:

```
1
5
(4,)
```

Scanned with CamScanner

Accessing individual elements of an Array

For 2D array **<array-name>[row, col]**

e.g. arr[1, 3]

or

arr[1][2]

e.g.

```
import numpy as np
Lst=([1,3,5,7],[2,4,6,8])
arr2=np.array(Lst)
print(arr2[0,3])
print(arr2[1,2])
```

Output:

```
7
6
(2, 4)
```

Scanned with CamScanner

Write a NumPy program to import NumPy as np and print the version number.

Python Code:

```
import numpy as np
print(np.__version__)
```

Scanned with CamScanner

Creating a new array and returns with all zeroes as its elements

```
import numpy as np
arr3=np.zeros((5,), dtype=np.int)
print(arr3)
```

Output:
[0 0 0 0 0]

Scanned with CamScanner

```
import numpy as np
arr4=np.ones([2,2], dtype=int)
print(arr4)
```

Output:

```
[[1 1]
 [1 1]]
```

Creating a two dimensional array of fixed value of 8

Scanned with CamScanner

Write a NumPy program to create a1-D array containing numeric values from 0 to 9

```
import numpy as np
x = np.arange(10)
print("Array from 0 to 9:")
print(x)
```

Output:

```
Array from 0 to 9:
[0 1 2 3 4 5 6 7 8 9]
```

Scanned with CamScanner

Write a NumPy program to create a 1-D array of odd numbers from 1 to 10

```
import numpy as np
x = np.arange(1,10,2)
print("Array of odd numbers 1 to 10:")
print(x)
```

Output:

Array from 1 to 10:
[1 3 5 7 9]

Scanned with CamScanner

To create numpy array with all values as True

```
import numpy as np
arr5 = np.ones([2,3], dtype=bool)
print(arr5)
```

Output:

```
[[ True  True  True]
 [ True  True  True]]
```

Scanned with CamScanner

float

```
import numpy as np
arr3=np.zeros((5,), dtype=float)
print(arr3)
```

Output:
[0. 0. 0. 0. 0.]

To create an 1D array of five ones with data type as

Scanned with CamScanner

float

```
import numpy as np
arr5=np.zeros([2,3], dtype=float)
print(arr5)
```

Output:

```
[[0. 0. 0.]
 [0. 0. 0.]]
```

To create an 2D array of ones with data type as float

Scanned with CamScanner

copy() – Create a copy of an existing array

e.g.

```
import numpy as np
x=np.array([1,2,3,4])
print(x)
y=np.copy(x)
print(y)
```

Output:

```
[1 2 3 4]

[1 2 3 4]
```

Scanned with CamScanner

reshape() – Create a 2D array from 1D array

e.g.

```
import numpy as np
x=np.array([1,2,3,4,5,6])
y=np.reshape(x, [2, 3])

print("1-D array")
print(x)

print("2-D array")
print(y)
```

Output:

```
1-D array
[1 2 3 4 5 6]

2-D array
[[1 2 3]
 [4 5 6]]
```

Scanned with CamScanner

Arithmetic operation on arrays

Perform basic operations on 1D array

e.g.

```
import numpy as np
a=np.array([1,2,3,4])
b=np.array([10,20,30,40])
c=a+b
print(c)
c=a-b
print(c)
c=a*b
print(c)
c=a/b
print(c)
```

Output:

```
[11 22 33 44]

[ -9 -18 -27 -36]

[ 10  40  90 160]

[0.1 0.1 0.1 0.1]
```

Scanned with CamScanner

Arithmetic operation on arrays

Perform aggregate operations on 1D array

e.g.

```
import numpy as np
a=np.array([1,2,3,4])
print(a.sum())
print(a.min())
print(a.max())
print(a.mean())
print(np.median(a))
```

Output:

```
10
1
4
2.5
2.5
```

Scanned with CamScanner

Arithmetic operation on arrays

Perform basic operations on 2D array

e.g.

```
import numpy as np
x=np.array([[1,3,5],[2,4,6]])
print(x)
y=np.array([10,20,30])
print(y)
print("----------------")
z=np.add(x,y)
print(z)

z=np.subtract(x,y)
print(z)

z=np.multiply(x,y)
print(z)

z=np.divide(x,y)
print(z)
```

Output:

```
[[1 3 5]
 [2 4 6]]

[10 20 30]
----------------
[[11 23 35]
 [12 24 36]]

[[ -9 -17 -25]
 [ -8 -16 -24]]

[[ 10  60 150]
 [ 20  80 180]]

[[0.1        0.15       0.16666667]
 [0.2        0.2        0.2       ]]
```

Scanned with CamScanner

Arithmetic operation on arrays

Perform basic operations using values on 2D array

e.g.

```
import numpy as np
x=np.array([[1,3,5],[2,4,6]])
print(x)

print("-----------------")
z=np.add(x,2)
print(z)

z=np.subtract(x,2)
print(z)

z=np.multiply(x,2)
print(z)

z=np.divide(x,2)
print(z)
```

Output:

```
[[1 3 5]
 [2 4 6]]

-----------------
[[3 5 7]
 [4 6 8]]

[[-1  1  3]
 [ 0  2  4]]

[[ 2  6 10]
 [ 4  8 12]]

[[0.5 1.5 2.5]
 [1.  2.  3. ]]
```

Scanned with CamScanner