NUMPY

Ex.No. :1

Date :

AIM:

Use of numpy python library in N-dimensional array.

SOFTWARE USED:

Jupyter notebook

NUMPY-1

DESCRIPTION:

1. Declaration of an one dimensional array.
2. Declaration of a two dimensional array.
3. Declaration of a three dimensional array.
4. Declaration of a array with zeros.
5. Declaration of a array with random values.
6. Declaration of a sequenced array.
7. Reshaping an array.
8. Flattening an array.
9. Dimension of an array.
10. Shape of an array.
11. Datatype of the array.
12. Size of the array.
13. Change of Datatype.

PROGRAM:

```
import numpy as np

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

print("One Dimensional array:\n",a)


import numpy as np

b=np.array([[10,20,30,40],[50,60,70,80]])

print("Two Dimensional array:\n",b)


import numpy as np
```

```
c=np.array([[10,20,30,40],[50,60,70,80],[90,100,110,120]])
print("Three Dimensional array:\n",c)

import numpy as np
d=np.zeros((3,5))
print("Array with all zeros:\n",d)

import numpy as np
e=np.random.random((2,3))
print("\nRandom value:\n",e)

import numpy as np
f=np.arange(20,60,5)
print("\nSequence array:\n",f)

import numpy as np
g=np.array([[10,20,30,40],[50,60,70,80],[90,100,110,120]])
h=g.reshape(4,3)
print("\nnormal three dimensional array:\n",g,"\nreshaped array:\n",h)

import numpy as np
g=np.array([[10,20,30,40],[50,60,70,80],[90,100,110,120]])
i=g.flatten()
print("\nnormal three dimensional array:\n",g,"\nFlatten array:\n",i)

import numpy as np
g=np.array([[[10,20,30,40],[50,60,70,80]],[[90,100,110,120],[130,140,150,160]]])
print("\nArray dimension:\n",g.ndim)

import numpy as np
c=np.array([[10,20,30,40],[50,60,70,80],[90,100,110,120]])
```

```
print("Shape of the array:\n",c.shape)

import numpy as np
c=np.array([[10,20,30,40],[50,60,70,80],[90,100,110,120]])
print("Datatype of the array:\n",c.dtype)

import numpy as np
c=np.array([[10,20,30,40],[50,60,70,80],[90,100,110,120]])
size=len(c)
print("size of the array:\n",size)

import numpy as np
a=[1,2,3]
a=np.array(a)
newtype=a.astype('float64')
print(a)
print(newtype)
print(newtype.dtype)
```

OUTPUT:

```
One Dimensional array:
 [10 20 30 40]

 Two Dimensional array:
 [[10 20 30 40]
 [50 60 70 80]]

Three Dimensional array:
 [[ 10  20  30  40]
 [ 50  60  70  80]
 [ 90 100 110 120]]
```

Array with all zeros:

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

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

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

Random value:

[[0.58129539 0.56067485 0.99197945]

[0.45644843 0.22265461 0.12042483]]

Sequence array:

[20 25 30 35 40 45 50 55]

normal three dimensional array:

[[10 20 30 40]

[50 60 70 80]

[90 100 110 120]]

reshaped array:

[[10 20 30]

[40 50 60]

[70 80 90]

[100 110 120]]

normal three dimensional array:

[[10 20 30 40]

[50 60 70 80]

[90 100 110 120]]

Flatten array:

[10 20 30 40 50 60 70 80 90 100 110 120]

Array dimension:

3

Shape of the array:

(3, 4)

Datatype of the array:

int64

size of the array:

3

[1 2 3]

[1. 2. 3.]

float64

NUMPY-2

DESCRIPTION:

1. Slicing an array.
2. Slicing with a step of 2.
3. Slicing with a step of 3.
4. Slicing with a step of 4.
5. Reversing the array.
6. Extracting first row, first column and return every other row.
7. Array indexing.
8. Vertical joining .
9. Horizontal joining.
10. Depth joint.
11. Array splitting.

PROGRAM :

```
import numpy as np

a=np.array([1,2,3,4,5,6,7,8,9])

slice1=a[2:6]

print(slice1)
```

```
import numpy as np
a=np.array([1,2,3,4,5,6,7,8,9])
slice2=a[::2]
print(slice2)
import numpy as np
a=np.array([1,2,3,4,5,6,7,8,9])
slice3=a[::3]
print(slice3)

import numpy as np
a=np.array([1,2,3,4,5,6,7,8,9])
slice4=a[::4]
print(slice4)

import numpy as np
a=np.array([1,2,3,4,5,6,7,8,9])
rev_array=a[::-1]
print(rev_array)

import numpy as np
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
subarr=a[0:2,1:3]
print(subarr)

import numpy as np
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
subarr=a[:1,:3]
print(subarr)

import numpy as np
```

```
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
col1=a[:,0]
print(col1)
row1=a[0,:]
print(row1)

import numpy as np
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(a[0:3:2])

import numpy as np
import array
a=array.array('i',[1,2,3,4,5])
print(a[0])
print(a[3])

import numpy as np
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(a[0,0])
print(a[2,1])

import numpy as np
arr=[10,20,30]
print(arr[1])
print(arr[-1])
print(arr[1:4])

import numpy as np
arr=np.array([10,20,30,40,50])
print(arr[arr>25])
print(arr[[1,3]])
```

```
import numpy as np
arr1=np.array([[1,2],[3,4]])
arr2=np.array([[5,6],[7,8]])
result=np.hstack((arr1,arr2))
print(result)
```

```
import numpy as np
arr1=np.array([[1,2],[3,4]])
arr2=np.array([[5,6],[7,8]])
result=np.vstack((arr1,arr2))
print(result)
```

```
import numpy as np
arr1=np.array([[1,2],[3,4]])
arr2=np.array([[5,6],[7,8]])
result=np.dstack((arr1,arr2))
print(result)
```

```
import numpy as np
arr=np.array([1,2,3,4,5,6,7])
res=np.array_split(arr,3)
print(res)
```

OUTPUT:

[3 4 5 6]

[1 3 5 7 9]

[1 4 7]

[1 5 9]

[9 8 7 6 5 4 3 2 1]

[[2 3]
 [5 6]]

[[1 2 3]]

[1 4 7]
[1 2 3]

[[1 2 3]
 [7 8 9]]

1
4

1
8

20
30
[20, 30]

[30 40 50]

[20 40]

[[1 2 5 6]

[3 4 7 8]]

[[1 2]

[3 4]

[5 6]

[7 8]]

[[[1 5]

[2 6]]

[[3 7]

[4 8]]]

[array([1,2,3]),array([4,5]),array([6,7])]

NUMPY-3

DESCRIPTION:

1. Index retrival
2. Searching values to insert
3. Sorting arrays.
4. 2D array sorting
5. Boolean indexing
6. Filtering

PROGRAM :

```
import numpy as np

arr=np.array([1,6,2,7,6,4])

x=np.where(arr==7)
```

```
print(x)

import numpy as np
arr=np.array([1,6,2,7,6,4])
x=np.where(arr%3==1)
print(x)

import numpy as np
arr=np.array([0,1,2,2,7,8,9])
x=np.searchsorted(arr,8,side='right')
print(x)

import numpy as np
arr=np.array([9,5,2,7,6,4])
print(np.sort(arr))

import numpy as np
arr=np.array([[7,4,9],[9,1,4]])
print(np.sort(arr))

import numpy as np
arr=np.array([31,78,69,100])
x=[False,True,False,True]
newarr=arr[x]
print(x)
print(newarr)

import numpy as np
arr=np.array([31,78,69,100])
filter_arr=arr>70
newarr=arr[filter_arr]
```

```
print(filter_arr)
print(newarr)
```

OUTPUT:

```
(array([3]),)

(array([0, 3, 5]),)

6

[2 4 5 6 7 9]

[[4 7 9]
 [1 4 9]]

[False, False, False, False]
[]

[False  True False  True]
[ 78 100]
```

NUMPY-3

DESCRIPTION:

Vector operations

PROGRAM :

```
import numpy as np
arr1=[20,40,50,70,10]
arr2=[6,2,9,7,1]
a=np.array(arr1)
b=np.array(arr2)
print(a)
```

```
print(b)
print(a+b)
print(a-b)
print(a*b)
print(a/b)
print(a.dot(b))
print()
sclr=3
print("scalar value:",sclr)
print("array:",a)
print("Result:",a*sclr)

import numpy as np
a=np.array([[10,20],[30,40]])
b=np.array([[3,7],[5,9]])
print(a%b)

def my_func(x,y):
    if x>y:
        return x-y
    else:
        return x+y
arr1=[10,7,2]
arr2=[6,5,3]
vect_func=np.vectorize(my_func)
print("Array1:",arr1)
print("Array2:",arr2)
print("Result:",vect_func(arr1,arr2))
```

OUTPUT:

```
[20 40 50 70 10]
```

[6 2 9 7 1]

[26 42 59 77 11]

[14 38 41 63 9]

[120 80 450 490 10]

[3.33333333 20.   5.55555556 10.   10.   ]

1150

scalar value: 3

array: [20 40 50 70 10]

Result: [60 120 150 210 30]

[[1 6]

 [0 4]]

Array1: [10, 7, 2]

Array2: [6, 5, 3]

Result: [4 2 5]

RESULT:

Thus, various operations has been performed in N-dimensional array using numpy.