In [1]:

```
!pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\munep\anaconda3\lib\site-
packages (1.21.5)
```

In [4]:

```
# things to import numpy lib:
import numpy as np
```

Difference between array v/s list

```
1. array ocupy less memory
2.it works faster than list
3.i is very convenient
```

the N-Dimensional Array

In [31]:

```
# create a 2 list of height and weight
person_height=[5.2,5.4,8.9,4,5,6]
person_weight=[81,55,65,75,44,54]


# how to convert the following list to array
person_height=np.array(person_height)
person_weight=np.array(person_weight)
```

In [32]:

```
person_height
```

Out[32]:

```
array([5.2, 5.4, 8.9, 4. , 5. , 6. ])
```

In [20]:

```
print(person_weight)
```

```
[81 55 65 75 44 54]
```

In [ ]:

```

```

ASSIGNMENTS

In [ ]:

```
Write a Python program to add two given lists using map and lambda.

Original list:
[1, 2, 3] [4, 5, 6]

Result: after adding two list [5, 7, 9]
```

In [22]:

```
list1 = [1, 2, 3]
list2 = [4, 5, 6]

result = list(map(lambda a, b: a+b,(list1), (list2)))

print(" after adding two lists:", result)
```

```
 after adding two lists: [5, 7, 9]
```

In [33]:

```
# consider of numpy array over list
num_list=[4,5,6,1,2,3]
# add 1 to each element
num_modified_list=[i+1 for i in num_list]
num_modified_list
```

Out[33]:

```
[5, 6, 7, 2, 3, 4]
```

In [5]:

```
# create an array from a list
num_array=np.array([4,5,6,1,2,3])
# add 1 to each element:
num_modified_array=num_array+1
num_modified_array
```

Out[5]:

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

In [8]:

```

# create a numpy array using a arrange functions
my_array=np.arange(10)
# create a python list using range:
my_list=list(range(10))
print("Range functions by array-:",my_array)
print("Range function by lists-:",my_list)
```

```
Range functions by array-: [0 1 2 3 4 5 6 7 8 9]
Range function by lists-: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [6]:

```
from numpy import zeros
my_array = zeros([4,5,6])
print(my_array)
```

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

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

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

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

In [3]:

```
from numpy import zeros
my_array = zeros([4,5])
print(my_array)
```

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

In [11]:

```
# create a an array of two random numbers between 2 and 9
# randint- will not include 10,the numbers will be generated between 2 and 9
my_array = np.random.randint(1,11,3)
my_array
```

Out[11]:

```
array([ 8,  2, 10])
```

ASSIGNMENT

In [ ]:

```
write a python program to remove None  values from a given list using the lambda functio

list1=[12,0,None,23,-55,234,89,None,0,6,-12]
```

In [10]:

```
list1=[12,0,None,23,-55,234,89,None,0,6,-12]
list2=list(filter(lambda x:x is not None,list1))
print(list2)
```

```
[12, 0, 23, -55, 234, 89, 0, 6, -12]
```

In [11]:

```
list1=[12,0,None,23,-55,234,89,None,0,6,-12]
list2=tuple(filter(lambda x:x is not None,list1))
print(list2)
```

```
(12, 0, 23, -55, 234, 89, 0, 6, -12)
```

ASSIGNMENT

In [ ]:

```
Bhuvanesh Chellapandian9:12 PM
Write a NumPy program to create a 2D array with 1 on the border and 0 inside
Expected Output:
Original array:
[[ 1. 1. 1. 1. 1.]
...................
[ 1. 1. 1. 1. 1.]]
1 on the border and 0 inside in the array
[[ 1. 1. 1. 1. 1.]
...................
[ 1. 1. 1. 1. 1.]]
```

In [15]:

```
from numpy import ones
my_array = np.ones([5,5])
my_array
```

Out[15]:

```
array([[1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.]])
```

In [7]:

```
import numpy as np
from numpy import ones
my_array = np.ones([5,5])
my_array[1:4]=0
my_array
```

Out[7]:

```
array([[1., 1., 1., 1., 1.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [1., 1., 1., 1., 1.]])
```

In [11]:

```
import numpy as np
my_array = np.ones([5,5])
my_array[1:4,1:4]=0
my_array
```

Out[11]:

```
array([[1., 1., 1., 1., 1.],
       [1., 0., 0., 0., 1.],
       [1., 0., 0., 0., 1.],
       [1., 0., 0., 0., 1.],
       [1., 1., 1., 1., 1.]])
```

In [7]:

```
import numpy as np
from numpy import ones
my_array = np.array([[5,5],[5,5]])
my_array[1:4,1:4]=0
my_array
```

Out[7]:

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

In [8]:

```
array_string=np.array(['pyton','sql','R'])
array_string
```

Out[8]:

```
array(['pyton', 'sql', 'R'], dtype='<U5')
```

In [10]:

```
array=np.array(["jhahaaaaaaaaaaaaaa habz jkkk zbbbbbbbb kna"])
print(np.char.capitalize(array)[2])
```

```
---------------------------------------------------------------------------
-
IndexError                                Traceback (most recent call las
t)
~\AppData\Local\Temp\ipykernel_15572\4207220575.py in <module>
      1 array=np.array(["jhahaaaaaaaaaaaaaa habz jkkk zbbbbbbbb kna"])
----> 2 print(np.char.capitalize(array)[2])

IndexError: index 2 is out of bounds for axis 0 with size 1
```

In [13]:

```
array=np.array(["jhahaaaaaaaaaaaaaa habz jkkk zbbbbbbbb kna"])
print(np.char.capitalize(array)[0][25:29])
```

```
jkkk
```

In [14]:

```
array=np.array(["jhahaaaaaaaaaaaaaa habz jkkk zbbbbbbbb kna"])
print(np.char.capitalize(array)[0][2])
```

```
a
```

ASSIGNMENT

```
Write a NumPy program to create a vector with values ••ranging from 15 to 55 and print
all values ••except the first and last.
```

In [26]:

```
vector=np.arange(15,56)
print(vector)


print(vector[1:-1])
```

```
[15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38
 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55]
[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54]
```

```
Write a NumPy program to create a vector of length 5 filled with arbitrary integers
from 0 to 10
```

In [37]:

```
vam = np.random.randint(0, 11, size=5)
print(vam)
```

```
[1 4 1 9 4]
```

Write a NumPy program to create a 3x4 matrix filled with values ••from 10 to 21.

In [28]:

```
vamsi=np.arange(10,22)
vamsi
var=vamsi.reshape(3,4)
var
```

Out[28]:

```
array([[10, 11, 12, 13],
       [14, 15, 16, 17],
       [18, 19, 20, 21]])
```

Write a NumPy program to create a 3x3x3 array filled with arbitrary values.

In [29]:

```
vamsi=np.arange(1,28)
vamsi
var=vamsi.reshape(3,3,3)
var
```

Out[29]:

```
array([[[ 1,  2,  3],
        [ 4,  5,  6],
        [ 7,  8,  9]],

       [[10, 11, 12],
        [13, 14, 15],
        [16, 17, 18]],

       [[19, 20, 21],
        [22, 23, 24],
        [25, 26, 27]]])
```

Write a NumPy program to compute the inner product of two given vectors.

In [31]:

```
abc=np.array([2,3,4])
bcd=np.array([4,5,6])
vec=abc.dot(bcd) #2*4 + 3*5 + 4*6 = 8 + 15 + 24 = 47

vec
```

Out[31]:

```
47
```

In [1]:

```
print('concate two strings')
print('np.char.add(['helllo],['xyz'])
```

```
  File "C:\Users\munep\AppData\Local\Temp\ipykernel_3824\3152825097.py", l
ine 2
    print('np.char.add(['helllo],['xyz'])
                             ^
SyntaxError: invalid syntax
```

In [3]:

```
# using upper and lower:
import numpy as np
print(np.char.upper('hello'))
print(np.char.lower(['HELLO WORLD']))
```

```
HELLO
['hello world']
```

In [4]:

```
# using multiply:
print(np.char.multiply('hello',3))
```

```
hellohellohello
```

In [5]:

```
# split:
print(np.char.split('hello how are you'))
print(np.char.split('hello\rhow are you'))
```

```
['hello', 'how', 'are', 'you']
['hello', 'how', 'are', 'you']
```

In [7]:

```
# strip
print(np.char.strip(['arora/','*********admin','jaava*/>','aaa'],'/>*'))
```

```
['arora' 'admin' 'jaava' 'aaa']
```

In [8]:

```
# join:
print(np.char.join(':','dmy'))
print(np.char.join([':','-'],['dmy','ymd']))
```

```
d:m:y
['d:m:y' 'y-m-d']
```

In [9]:

```
# ussage of replace:
print(np.char.replace('He is a Good Boy','is','was'))
```

```
He was a Good Boy
```

```
**numpy.median()**
** meadian is defined as the value separating the higher half of the data from the
lower half
```

In [12]:

```
a=np.array([[30,65,70],
            [69,45,23],
            [23,44,56]])
```

In [13]:

```
np.median(a)
```

Out[13]:

```
45.0
```

```
# ** np.mean()**
** arithmetic mean is the sum of elements alomg the axis devided by the number of
elements
** it returns the arithmetic mean of all elements in the array
```

np.average()

the weight average is calculated by adding the product the corresponding elemnts

and deviding by the sum of weights

```
a=[1,2,3,4]
weight=[4,3,2,1]
weighted_average=(1*4+2*3+3*2+4*1)/(4+3+2+1)
```

In [19]:

```
a=np.array([1,2,3,4])
print('Applying the Average () function: ')
print(a)
print('\n')


wts = np.array([4,3,2,1])
print('Applying the Average () function again:')
print(np.average(a,weights=wts))
print('\n')

print('sum of weights:')
# if the returned parameter is true- it returns the sum oof weights
print(np.average([1,2,3,4],weights=[4,3,2,1],returned =True))
```

```
Applying the Average () function:
[1 2 3 4]


Applying the Average () function again:
2.0


sum of weights:
(2.0, 10.0)
```

ASSIGNMENT

In [ ]:

```
Given a 2D binary matrix A of dimensions NxM, determine the row that contains a minimum
Note-The matrix contains only 1s and 0s. Also, if two or more rows contain the minimum n

Input:
N=4,M=4
A=[[1,1,1,1],[1,1,0,0],[0,0,1,1],[1,1,1,1]]
Output:
2
Explanation:
Rows 2 and 3 contain the minimum number
of 1's(2 each).Since, 2 is less than 3.
Thus, the answer is 2.
```

In [ ]:

```

```