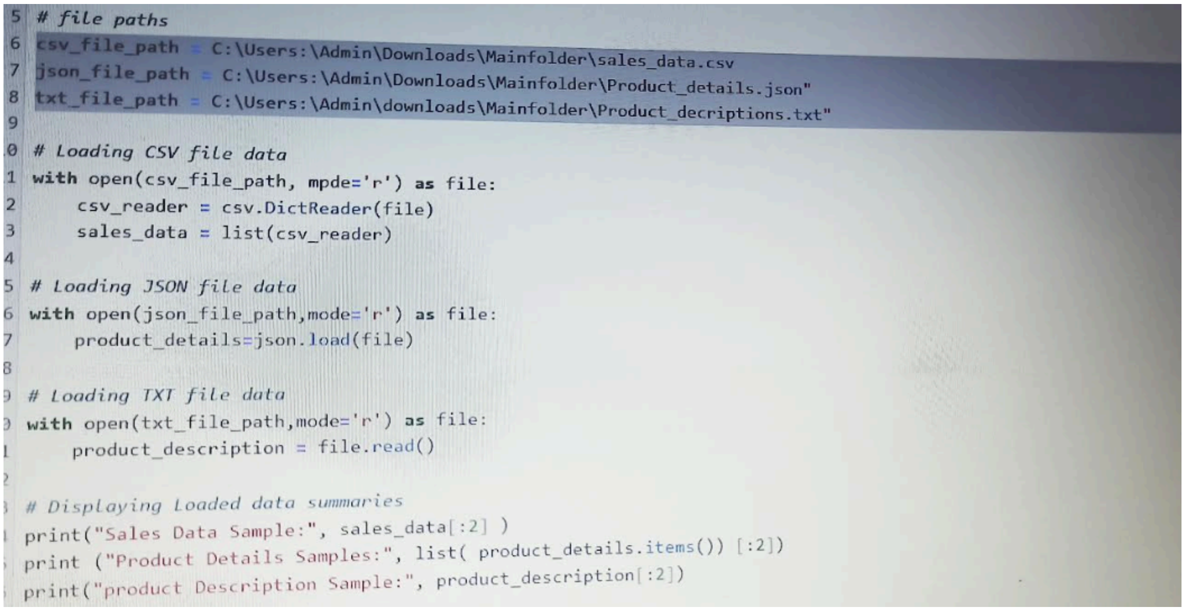

```
# file paths
csv_file_path = C:\Users:\Admin\Downloads\Mainfolder\sales_data.csv
json_file_path = C:\Users:\Admin\Downloads\Mainfolder\Product_details.json"
txt_file_path = C:\Users:\Admin\downloads\Mainfolder\Product_decriptions.txt"

# Loading CSV file data
with open(csv_file_path, mpde='r') as file:
    csv_reader = csv.DictReader(file)
    sales_data = list(csv_reader)

# Loading JSON file data
with open(json_file_path,mode='r') as file:
    product_details=json.load(file)

# Loading TXT file data
with open(txt_file_path,mode='r') as file:
    product_description = file.read()

# Displaying Loaded data summaries
print("Sales Data Sample:", sales_data[:2] )
print ("Product Details Samples:", list( product_details.items()) [:2])
print("product Description Sample:", product_description[:2])
```

In [1]:
```
csv_file_path = r"C:\Users\AEPAC\Desktop\WS\Sales_Csv\Jitendra.csv"
```

In [2]:
```
csv_file_path
```

Out[2]:
```
'C:\\Users\\AEPAC\\Desktop\\WS\\Sales_Csv\\Jitendra.csv'
```

Python Operating System Module

- This module or video will help you interms of handling the files and folder structure and its navigation.

https://www.youtube.com/watch?v=Tp4qTuHROX4&list=PLWuFHho1zKhWb-f-SJAMUCK--f8PJlG46&index=26 (https://www.youtube.com/watch?v=Tp4qTuHROX4&list=PLWuFHho1zKhWb-f-SJAMUCK--f8PJlG46&index=26)

In [3]:
```
import os
```

In [4]:
```
os.getcwd()
```

Out[4]:
```
'C:\\Users\\AEPAC\\Desktop\\KnowledgeHut\\2024\\2024_upGrad\\Python\\10 - 7t
h Sep - Python NumPy - 1'
```

In [5]:
```
os.chdir(os.environ["userprofile"] + "\Desktop\WS\Sales_Csv")
```

In [6]:
```
os.getcwd()
```

Out[6]:
```
'C:\\Users\\AEPAC\\Desktop\\WS\\Sales_Csv'
```

In [7]:

```
# Can you display what are files I have?
```

In [8]:

```
os.listdir()
```

Out[8]:

```
['Connecticut.csv',
 'CV-Avdhesh Saraswat (Lead Associate).pdf',
 'Data Sets - Shortcut.lnk',
 'DataVisualization.pdf',
 'Delaware.csv',
 'District of Columbia.csv',
 'East.csv',
 'Edureka',
 'Future - Shortcut.lnk',
 'Goa Ticket.pdf',
 'Illinois.csv',
 'Indiana.csv',
 'Iowa.csv',
 'Jitendra.csv',
 'Kansas.csv',
 'Letters',
 'LY - Shortcut.lnk',
 'Maine.csv',
 'Maryland.csv',
 'Massachusetts.csv',
 'Michigan.csv',
 'Missouri.csv',
 'Nebraska.csv',
 'New Jersey.csv',
 'New York.csv',
 'North Dakota.csv',
 'Ohio.csv',
 'Oklahoma.csv',
 'RAil Ticket',
 'Recycle Bin - Shortcut.lnk',
 'South Dakota.csv',
 'SQL.pdf',
 'SuperStore - Shortcut.lnk',
 'TBD - Shortcut.lnk',
 'Testing - Shortcut.lnk',
 'Texas.csv',
 'Today - Shortcut.lnk',
 'Tutorial - Shortcut.lnk',
 'upGrad - Shortcut.lnk',
 'Wisconsin.csv',
 'xl - Shortcut.lnk',
 'xlwings Tutorial']
```

In [ ]:

```
os.environ -> c:\users\yoursytemName
```

In [10]:

```
for fl in os.listdir():
    if fl.endswith("csv"):
        if fl.startswith("M") or fl.startswith("N"):
            print(fl)
```

```
Maine.csv
Maryland.csv
Massachusetts.csv
Michigan.csv
Missouri.csv
Nebraska.csv
New Jersey.csv
New York.csv
North Dakota.csv
```

Introduction of NumPy:

Numpy is python package, which is suitable for mathematical computing, NumPy stands for Numerical Python. Using the numpy we can handle data and data manipulation.

In [11]:

```
import numpy as np
```

How we can creating an array?

What is an array?

Array is the collection of similar data type. We can not store composite data type in a array.

In [13]:

```
np_array = np.array([])
```

In [14]:

```
np_array
```

Out[14]:

```
array([], dtype=float64)
```

In [15]:

```
type(np_array)
```

Out[15]:

```
numpy.ndarray
```

In [16]:

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

In [17]:

```
arr
```

Out[17]:

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

```
In [18]: type(arr)
```

```
Out[18]: numpy.ndarray
```

```
In [255]: lstOfNumber = [10,20,30,40,50,60]
```

Converting above list into the numpy array

```
In [256]: arr = np.array(object = lstOfNumber)
          # arr = np.array(lstOfNumber)
```

```
In [257]: arr
```

```
Out[257]: array([10, 20, 30, 40, 50, 60])
```

```
In [258]: list(arr)
```

```
Out[258]: [10, 20, 30, 40, 50, 60]
```

```
In [259]: tuple(arr)
```

```
Out[259]: (10, 20, 30, 40, 50, 60)
```

```
In [260]: set(arr)
```

```
Out[260]: {10, 20, 30, 40, 50, 60}
```

```
In [22]: # We can not store multiple data type in array.
```

```
In [23]: lstOfNumber
```

```
Out[23]: [10, 20, 30, 40, 50, 60]
```

```
In [24]: lstOfNumber.append("Firdos")
         # if you want to add an item in the array
         np.append(YourArrayName,YourItemName)
```

```
In [25]: lstOfNumber
```

```
Out[25]: [10, 20, 30, 40, 50, 60, 'Firdos']
```

```
In [26]: arr = np.array(lstOfNumber)
```

```
In [27]: arr
```

```
Out[27]: array(['10', '20', '30', '40', '50', '60', 'Firdos'], dtype='<U11')
```

```
In [28]: arr = np.array(["Abhisheak","PushpanjaliRaout","Firdos"])
```

```
In [29]: arr
```

```
Out[29]: array(['Abhisheak', 'PushpanjaliRaout', 'Firdos'], dtype='<U16')
```

```
In [30]: arr = np.array(lstOfNumber)
```

```
In [31]: arr
```

```
Out[31]: array(['10', '20', '30', '40', '50', '60', 'Firdos'], dtype='<U11')
```

Note:

Slicing concept would be remain same as on string or list as we have seen already in the past sessions.

```
In [32]: arr
```

```
Out[32]: array(['10', '20', '30', '40', '50', '60', 'Firdos'], dtype='<U11')
```

```
In [33]: arr[3]
```

```
Out[33]: '40'
```

```
In [34]: arr[-1]
```

```
Out[34]: 'Firdos'
```

```
In [35]: arr[::2]
```

```
Out[35]: array(['10', '30', '50', 'Firdos'], dtype='<U11')
```

```
In [36]: arr[::-1]
```

```
Out[36]: array(['Firdos', '60', '50', '40', '30', '20', '10'], dtype='<U11')
```

In [37]:

```
arr
```

Out[37]:

```
array(['10', '20', '30', '40', '50', '60', 'Firdos'], dtype='<U11')
```

In [38]:

```
type(arr[-1])
```

Out[38]:

```
numpy.str_
```

In [39]:

```
np_array
```

Out[39]:

```
array([], dtype=float64)
```

In [40]:

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

In [41]:

```
type(arr[-1])
```

Out[41]:

```
numpy.int32
```

In [42]:

```
np_arr = np.array([1,2,3,4], dtype = 'float')
```

In [43]:

```
np_arr
```

Out[43]:

```
array([1., 2., 3., 4.])
```

In [44]:

```
np_arr = np.array([1,2,3,4], dtype = 'str')
```

In [45]:

```
np_arr
```

Out[45]:

```
array(['1', '2', '3', '4'], dtype='<U1')
```

In [46]:

```
np_arr = np.array([1.5,2,3,4], dtype = 'int')
```

In [47]:

```
np_arr
```

Out[47]:

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

```
In [48]: np_arr = np.array([1,2,3,4,"Abhishek"], dtype = 'float')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[48], line 1
----> 1 np_arr = np.array([1,2,3,4,"Abhishek"], dtype = 'float')

ValueError: could not convert string to float: 'Abhishek'
```

```
In [49]: np_arr = np.array([1,2,3,4,"6"], dtype = 'float')
```

```
In [50]: np_arr
```

```
Out[50]: array([1., 2., 3., 4., 6.])
```

```
In [51]: int("5")
```

```
Out[51]: 5
```

```
In [52]: int("A")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[52], line 1
----> 1 int("A")

ValueError: invalid literal for int() with base 10: 'A'
```

Creating 2D array

```
In [53]: arr = np.array([[1,2,3],[4,5,6]])
```

```
In [54]: arr
```

```
Out[54]: array([[1, 2, 3],
                [4, 5, 6]])
```

```
In [55]: arr.shape # here 2 rows and 3 cols
```

```
Out[55]: (2, 3)
```

```
In [56]: arr.size

Out[56]: 6

In [57]: arr.ndim

Out[57]: 2

In [58]: arr = np.array([[[1,2,3],[4,5,6],[7,8,9]]])

In [59]: arr

Out[59]: array([[[1, 2, 3],
                 [4, 5, 6],
                 [7, 8, 9]]])

In [60]: arr.shape

Out[60]: (1, 3, 3)

In [61]: arr.size

Out[61]: 9

In [62]: arr.ndim

Out[62]: 3

In [63]: arr

Out[63]: array([[[1, 2, 3],
                 [4, 5, 6],
                 [7, 8, 9]]])

In [66]: arr[0][0]

Out[66]: array([1, 2, 3])

In [67]: arr[0][1]

Out[67]: array([4, 5, 6])

In [68]: arr[0][2]

Out[68]: array([7, 8, 9])
```

```
In [69]:  arr[0][2][-1]

Out[69]:  9

In [266]: arr = np.array([1,2], ndmin = 5)

In [267]: arr

Out[267]: array([[[[[1, 2]]]]])

In [76]:  arr[0][0][0][0][-1]

Out[76]:  2

In [78]:  np.array([1,2,4,5]).shape

Out[78]:  (4,)

In [80]:  np.array([[1,2,4,5],[1,2,3,4]]).shape

Out[80]:  (2, 4)

In [81]:  arr = np.ones((3,3))

In [82]:  arr

Out[82]:  array([[1., 1., 1.],
                 [1., 1., 1.],
                 [1., 1., 1.]])

In [268]: np.zeros([3,3])

Out[268]: array([[0., 0., 0.],
                 [0., 0., 0.],
                 [0., 0., 0.]])

In [270]: np.diagonal([[1,2,3],[4,5,6],[7,8,9]])

Out[270]: array([1, 5, 9])

In [271]: arr = np.array([[1,2,3],[4,5,6],[7,8,9]])
```

In [272]:

```
arr
```

Out[272]:

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

In [273]:

```
np.diagonal(arr)
```

Out[273]:

```
array([1, 5, 9])
```

How to generate numbers using numpy

In [275]:

```
np.random.randn(10)
```

Out[275]:

```
array([-1.06255354, -1.14993969, -0.47948669,  0.05653042,  0.41837912,
       -2.11659915,  1.42015449, -0.33606978,  0.32417921,  2.68190528])
```

In [84]:

```
numbers = np.random.randint(low = 10, high = 50, size = 150, dtype = 'int
```

In [85]:

```
numbers
```

Out[85]:

```
array([15, 43, 44, 36, 11, 21, 31, 14, 49, 44, 16, 25, 25, 33, 43, 27, 18,
       48, 26, 16, 17, 35, 23, 44, 40, 45, 12, 15, 45, 42, 12, 22, 47, 22,
       21, 14, 45, 32, 19, 46, 25, 28, 22, 40, 47, 40, 15, 32, 47, 13, 28,
       46, 14, 42, 20, 39, 11, 30, 49, 30, 15, 27, 32, 25, 23, 46, 19, 43,
       18, 32, 14, 37, 46, 28, 21, 34, 17, 37, 34, 37, 31, 38, 22, 29, 48,
       25, 43, 22, 27, 21, 17, 30, 19, 30, 47, 18, 31, 31, 31, 32, 26, 41,
       36, 17, 49, 38, 27, 27, 15, 46, 41, 15, 41, 47, 29, 21, 39, 39, 39,
       34, 25, 34, 11, 45, 28, 14, 42, 22, 45, 33, 26, 45, 24, 33, 35, 23,
       30, 43, 38, 29, 45, 18, 30, 30, 45, 27, 13, 47, 39, 28])
```

In [86]:

```
len(numbers)
```

Out[86]:

```
150
```

In [87]:

```
numbers = np.random.randint(10,50, size = 20)
```

In [88]:

```
numbers
```

Out[88]:

```
array([29, 25, 33, 22, 42, 27, 24, 28, 23, 30, 40, 13, 45, 25, 42, 25, 24,
       12, 22, 25])
```

NumPy arange function:

In [90]:

```
arr = np.arange(20) # array range -> (arange)
```

In [91]:

```
arr
```

Out[91]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

In [92]:

```
print(arr)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```

In [93]:

```
np.arange(10,20,2)
```

Out[93]:

```
array([10, 12, 14, 16, 18])
```

In [94]:

```
np.arange(100,10,-5)
```

Out[94]:

```
array([100,  95,  90,  85,  80,  75,  70,  65,  60,  55,  50,  45,  40,
        35,  30,  25,  20,  15])
```

In [95]:

```
np.arange(10, dtype = 'float64')
```

Out[95]:

```
array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
```

In [98]:

```
bl = np.array([True,False,False,True])
```

In [99]:

```
type(bl)
```

Out[99]:

```
numpy.ndarray
```

In [100]:

```
bl.dtype
```

Out[100]:

```
dtype('bool')
```

In [103]:

```
numbers = np.random.randint(low = 10, high = 500, size = 150)
```

In [104]:

```
numbers
```

Out[104]:

```
array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
        41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
       435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
        88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
       236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
       177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
       366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
       105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
        49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
       191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
       407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
       302, 335, 266, 191, 404, 165, 246])
```

In [105]:

```
# how to convert above array into the list
type(numbers)
```

Out[105]:

```
numpy.ndarray
```

In [109]:

```
lstofnumber = list(numbers)
```

In [108]:

```
numbers
```

Out[108]:

```
array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
        41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
       435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
        88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
       236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
       177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
       366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
       105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
        49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
       191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
       407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
       302, 335, 266, 191, 404, 165, 246])
```

In [115]:

```
count = 0
for num in lstofnumber:
    if num > 400:
        count = count + 1
        print(num, end = " ")
```

```
494 492 401 424 497 473 419 435 489 440 457 404 473 470 487 451 429 443 490
430 424 463 458 479 468 498 417 426 427 419 436 407 421 409 491 404
```

In [116]:

```
count
```

Out[116]:

```
36
```

In [111]:

```
numbers
```

Out[111]:

```
array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
        41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
       435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
        88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
       236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
       177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
       366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
       105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
        49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
       191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
       407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
       302, 335, 266, 191, 404, 165, 246])
```

In [113]:

```
sum(numbers > 400)
```

Out[113]:

```
36
```

Can you extract those number

In [117]:

```
numbers[-1]
```

Out[117]:

```
246
```

In [118]:

```
numbers[numbers > 400]
```

Out[118]:

```
array([494, 492, 401, 424, 497, 473, 419, 435, 489, 440, 457, 404, 473,
       470, 487, 451, 429, 443, 490, 430, 424, 463, 458, 479, 468, 498,
       417, 426, 427, 419, 436, 407, 421, 409, 491, 404])
```

In [119]:

```
len(numbers[numbers > 400])
```

Out[119]:

```
36
```

In [120]:

```
numbers
```

Out[120]:

```
array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
        41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
       435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
        88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
       236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
       177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
       366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
       105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
        49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
       191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
       407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
       302, 335, 266, 191, 404, 165, 246])
```

```
In [121]: numbers[numbers % 5 == 0]
```

```
Out[121]: array([160, 335, 370, 435, 440, 365, 470, 310, 225,  90, 490, 230, 430,
                 290,  50, 105, 195,  65,  80, 300, 280, 390, 215,  55, 380, 335,
                 165])
```

```
In [125]: numbers[(numbers % 5 == 0) | (numbers % 7 == 0)]
```

```
Out[125]: array([273, 160, 335, 497, 370, 435, 238, 259, 440, 365, 161, 294, 470,
                 310, 225, 119,  90,  91, 490, 230, 430, 290,  91, 287,  50, 105,
                 195,  65,  49,  80, 203, 427, 300, 329, 280, 259, 390,  42, 215,
                  55, 380, 335, 266, 165])
```

```
In [126]: numbers[(numbers % 5 == 0) & (numbers % 7 == 0)]
```

```
Out[126]: array([490, 105, 280])
```

Adding a number in a array

```
In [130]: numbers = np.append(numbers, 35)
```

```
In [131]: numbers
```

```
Out[131]: array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
                  41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
                 435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
                  88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
                 236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
                 177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
                 366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
                 105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
                  49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
                 191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
                 407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
                 302, 335, 266, 191, 404, 165, 246,  35])
```

```
In [132]: numbers[(numbers % 5 == 0) & (numbers % 7 == 0)]
```

```
Out[132]: array([490, 105, 280,  35])
```

```
In [133]: arr = np.random.randint(5,50,20)
```

```
In [134]: arr
```

```
Out[134]: array([18, 10, 47, 25,  8, 37, 45, 49, 39, 35, 44, 24, 26, 35, 15, 10, 10,
                 44, 26, 45])
```

```
In [135]:  arr * 2

Out[135]:  array([36, 20, 94, 50, 16, 74, 90, 98, 78, 70, 88, 48, 52, 70, 30, 20, 20,
                  88, 52, 90])
```

```
In [136]:  arr - 2

Out[136]:  array([16,  8, 45, 23,  6, 35, 43, 47, 37, 33, 42, 22, 24, 33, 13,  8,  8,
                  42, 24, 43])
```

```
In [137]:  arr / 2

Out[137]:  array([ 9. ,  5. , 23.5, 12.5,  4. , 18.5, 22.5, 24.5, 19.5, 17.5, 22. ,
                  12. , 13. , 17.5,  7.5,  5. ,  5. , 22. , 13. , 22.5])
```

```
In [138]:  arr

Out[138]:  array([18, 10, 47, 25,  8, 37, 45, 49, 39, 35, 44, 24, 26, 35, 15, 10, 10,
                  44, 26, 45])
```

```
In [139]:  np.sum(arr)

Out[139]:  592
```

```
In [140]:  np.min(arr)

Out[140]:  8
```

```
In [141]:  np.max(arr)

Out[141]:  49
```

```
In [142]:  np.std(arr)

Out[142]:  13.810865287881132
```

```
In [143]:  arr

Out[143]:  array([18, 10, 47, 25,  8, 37, 45, 49, 39, 35, 44, 24, 26, 35, 15, 10, 10,
                  44, 26, 45])
```

```
In [144]:  arr ** 2

Out[144]:  array([ 324,  100, 2209,  625,   64, 1369, 2025, 2401, 1521, 1225, 1936,
                   576,  676, 1225,  225,  100,  100, 1936,  676, 2025])
```

numpy Split Array

In [145]:

```
arr = np.arange(10)
```

In [146]:

```
arr
```

Out[146]:

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [148]:

```
list(np.split(arr,5))
```

Out[148]:

```
[array([0, 1]), array([2, 3]), array([4, 5]), array([6, 7]), array([8, 9])]
```

In [149]:

```
np.split(arr, 4)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[149], line 1
----> 1 np.split(arr, 4)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\lib\s
hape_base.py:864, in split(ary, indices_or_sections, axis)
    862     N = ary.shape[axis]
    863     if N % sections:
--> 864         raise ValueError(
    865             'array split does not result in an equal division') from
None
    866 return array_split(ary, indices_or_sections, axis)

ValueError: array split does not result in an equal division
```

In [150]:

```
np.split(arr,2)
```

Out[150]:

```
[array([0, 1, 2, 3, 4]), array([5, 6, 7, 8, 9])]
```

In [151]:

```
np.split(arr,10)
```

Out[151]:

```
[array([0]),
 array([1]),
 array([2]),
 array([3]),
 array([4]),
 array([5]),
 array([6]),
 array([7]),
 array([8]),
 array([9])]
```

In [152]:
```
arr
```
Out[152]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [153]:
```
np.split(arr,[2,4,5,7])
```
Out[153]:
```
[array([0, 1]), array([2, 3]), array([4]), array([5, 6]), array([7, 8, 9])]
```

In [154]:
```
arr
```
Out[154]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [155]:
```
np.split(arr,[2,5,7]) # [0,1],[2,3,4],[5,6],[6,7,8,9]
```
Out[155]:
```
[array([0, 1]), array([2, 3, 4]), array([5, 6]), array([7, 8, 9])]
```

In [156]:
```
np.split(arr,[3,7])
```
Out[156]:
```
[array([0, 1, 2]), array([3, 4, 5, 6]), array([7, 8, 9])]
```

In [157]:
```
np.split(arr,[3,10])
```
Out[157]:
```
[array([0, 1, 2]), array([3, 4, 5, 6, 7, 8, 9]), array([], dtype=int32)]
```

numpy reshpae function

In [158]:
```
arr = np.arange(16).reshape(4,4)
```

In [159]:
```
arr
```
Out[159]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15]])
```

In [160]:
```
arr = np.arange(25).reshape(5,5)
```

In [161]:
```
arr
```
Out[161]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

In [163]:

```
arr = np.arange(24).reshape(2,6,2)
```

In [164]:

```
arr
```

Out[164]:

```
array([[[ 0,  1],
        [ 2,  3],
        [ 4,  5],
        [ 6,  7],
        [ 8,  9],
        [10, 11]],

       [[12, 13],
        [14, 15],
        [16, 17],
        [18, 19],
        [20, 21],
        [22, 23]]])
```

In [165]:

```
arr = np.arange(24).reshape(2,3,2,2)
```

In [166]:

```
arr
```

Out[166]:

```
array([[[[ 0,  1],
         [ 2,  3]],

        [[ 4,  5],
         [ 6,  7]],

        [[ 8,  9],
         [10, 11]]],


       [[[12, 13],
         [14, 15]],

        [[16, 17],
         [18, 19]],

        [[20, 21],
         [22, 23]]]])
```

In [170]:

```
arr = np.arange(48).reshape(2,4,3,2)
```

```
In [171]: arr
```

```
Out[171]: array([[[[ 0,  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],
                   [28, 29]],

                  [[30, 31],
                   [32, 33],
                   [34, 35]],

                  [[36, 37],
                   [38, 39],
                   [40, 41]],

                  [[42, 43],
                   [44, 45],
                   [46, 47]]]])
```

```
In [172]: arr = np.arange(16).reshape(4,4)
```

```
In [173]: arr
```

```
Out[173]: array([[ 0,  1,  2,  3],
                 [ 4,  5,  6,  7],
                 [ 8,  9, 10, 11],
                 [12, 13, 14, 15]])
```

```
In [174]: np.split(arr, [1,3])
```

```
Out[174]: [array([[0, 1, 2, 3]]),
           array([[ 4,  5,  6,  7],
                  [ 8,  9, 10, 11]]),
           array([[12, 13, 14, 15]])]
```

In [176]:

```
np.split(arr, indices_or_sections=[1,3]) #by default axis=0, based on row
```

Out[176]:

```
[array([[0, 1, 2, 3]]),
 array([[ 4,  5,  6,  7],
        [ 8,  9, 10, 11]]),
 array([[12, 13, 14, 15]])]
```

In [177]:

```
arr
```

Out[177]:

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15]])
```

In [178]:

```
np.split(arr, indices_or_sections=[1,3], axis = 1) # it is based on cols
```

Out[178]:

```
[array([[ 0],
        [ 4],
        [ 8],
        [12]]),
 array([[ 1,  2],
        [ 5,  6],
        [ 9, 10],
        [13, 14]]),
 array([[ 3],
        [ 7],
        [11],
        [15]])]
```

In [179]:

```
arr = np.arange(64).reshape(8,8)
```

In [180]:

```
arr
```

Out[180]:

```
array([[ 0,  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, 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],
       [56, 57, 58, 59, 60, 61, 62, 63]])
```

In [181]:

```
np.split(arr,[2,4]) # based on row
```

Out[181]:

```
[array([[ 0,  1,  2,  3,  4,  5,  6,  7],
        [ 8,  9, 10, 11, 12, 13, 14, 15]]),
 array([[16, 17, 18, 19, 20, 21, 22, 23],
        [24, 25, 26, 27, 28, 29, 30, 31]]),
 array([[32, 33, 34, 35, 36, 37, 38, 39],
        [40, 41, 42, 43, 44, 45, 46, 47],
        [48, 49, 50, 51, 52, 53, 54, 55],
        [56, 57, 58, 59, 60, 61, 62, 63]])]
```

In [182]:

```
arr
```

Out[182]:

```
array([[ 0,  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, 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],
       [56, 57, 58, 59, 60, 61, 62, 63]])
```

In [183]:

```
np.split(arr,[2,4], axis = 1)
```

Out[183]:

```
[array([[ 0,  1],
        [ 8,  9],
        [16, 17],
        [24, 25],
        [32, 33],
        [40, 41],
        [48, 49],
        [56, 57]]),
 array([[ 2,  3],
        [10, 11],
        [18, 19],
        [26, 27],
        [34, 35],
        [42, 43],
        [50, 51],
        [58, 59]]),
 array([[ 4,  5,  6,  7],
        [12, 13, 14, 15],
        [20, 21, 22, 23],
        [28, 29, 30, 31],
        [36, 37, 38, 39],
        [44, 45, 46, 47],
        [52, 53, 54, 55],
        [60, 61, 62, 63]])]
```

In [184]:

```
arr
```

Out[184]:

```
array([[ 0,  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, 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],
       [56, 57, 58, 59, 60, 61, 62, 63]])
```

In [185]:

```
np.split(arr, [1,5])
```

Out[185]:

```
[array([[0, 1, 2, 3, 4, 5, 6, 7]]),
 array([[ 8,  9, 10, 11, 12, 13, 14, 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]]),
 array([[40, 41, 42, 43, 44, 45, 46, 47],
        [48, 49, 50, 51, 52, 53, 54, 55],
        [56, 57, 58, 59, 60, 61, 62, 63]])]
```

In [186]:

```
arr
```

Out[186]:

```
array([[ 0,  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, 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],
       [56, 57, 58, 59, 60, 61, 62, 63]])
```

In [187]:

```
np.split(arr, [1,5], axis = 1)
```

Out[187]:

```
[array([[ 0],
        [ 8],
        [16],
        [24],
        [32],
        [40],
        [48],
        [56]]),
 array([[ 1,  2,  3,  4],
        [ 9, 10, 11, 12],
        [17, 18, 19, 20],
        [25, 26, 27, 28],
        [33, 34, 35, 36],
        [41, 42, 43, 44],
        [49, 50, 51, 52],
        [57, 58, 59, 60]]),
 array([[ 5,  6,  7],
        [13, 14, 15],
        [21, 22, 23],
        [29, 30, 31],
        [37, 38, 39],
        [45, 46, 47],
        [53, 54, 55],
        [61, 62, 63]])]
```

numpy where method:

In [195]:

```
arr = np.array([1,4,7,5,11,13,19,26,31, 10, 12, 22, 6])
```

In [196]:

```
arr
```

Out[196]:

```
array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6])
```

In [197]:

```
lst = list(arr)
```

In [198]:

```
lst
```

Out[198]:

```
[1, 4, 7, 5, 11, 13, 19, 26, 31, 10, 12, 22, 6]
```

In [199]:

```
arr
```

Out[199]:

```
array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6])
```

```
In [200]: np.where(arr <= 12)

Out[200]: (array([ 0,  1,  2,  3,  4,  9, 10, 12], dtype=int64),)

In [208]: arr = np.append(arr,5)

In [209]: arr

Out[209]: array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6,  5,  5])

In [210]: np.where(arr == 5)

Out[210]: (array([ 3, 13, 14], dtype=int64),)

In [211]: arr[12]

Out[211]: 6

In [212]: arr

Out[212]: array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6,  5,  5])

In [213]: np.where(arr > 10, "A", arr)

Out[213]: array(['1', '4', '7', '5', 'A', 'A', 'A', 'A', 'A', '10', 'A', 'A', '6',
                 '5', '5'], dtype='<U11')

In [214]: lst

Out[214]: [1, 4, 7, 5, 11, 13, 19, 26, 31, 10, 12, 22, 6]

In [215]: lst.index(5)

Out[215]: 3

In [216]: lst.append(5)

In [217]: lst

Out[217]: [1, 4, 7, 5, 11, 13, 19, 26, 31, 10, 12, 22, 6, 5]

In [218]: lst.index(5,lst.index(5)+1)

Out[218]: 13
```

In [219]:

```
np.where(np.array(lst) == 5)
```

Out[219]:

```
(array([ 3, 13], dtype=int64),)
```

In [220]:

```
numbers
```

Out[220]:

```
array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
        41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
       435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
        88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
       236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
       177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
       366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
       105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
        49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
       191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
       407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
       302, 335, 266, 191, 404, 165, 246,  35])
```

In [221]:

```
numbers[numbers % 2 == 0]
```

Out[221]:

```
array([ 16, 160, 106, 164, 494, 492, 424, 296, 296, 384, 152, 370, 144,
       158, 238,  72, 374, 114, 440, 352,  88, 404,  74, 294, 314, 362,
       470, 166, 236, 102, 204, 310, 132,  62, 274, 104,  90, 166, 490,
        86, 366, 230, 430, 290, 424,  78,  50, 334, 458,  24, 202, 352,
       468, 498, 348,  80, 354, 426, 348, 300, 306, 280,  32, 390, 436,
        42, 362, 380, 302, 266, 404, 246])
```

In [222]:

```
numbers[np.where(numbers % 2 == 0)]
```

Out[222]:

```
array([ 16, 160, 106, 164, 494, 492, 424, 296, 296, 384, 152, 370, 144,
       158, 238,  72, 374, 114, 440, 352,  88, 404,  74, 294, 314, 362,
       470, 166, 236, 102, 204, 310, 132,  62, 274, 104,  90, 166, 490,
        86, 366, 230, 430, 290, 424,  78,  50, 334, 458,  24, 202, 352,
       468, 498, 348,  80, 354, 426, 348, 300, 306, 280,  32, 390, 436,
        42, 362, 380, 302, 266, 404, 246])
```

In [223]:

```
arr = np.random.randint(1,50,25).reshape(5,5)
```

In [224]:

```
arr
```

Out[224]:

```
array([[26, 29,  9, 38, 11],
       [14, 36, 16, 39, 41],
       [31, 38, 12, 19,  1],
       [24, 36,  6, 48, 17],
       [42, 43,  4, 24, 45]])
```

```
In [225]: lst

Out[225]: [1, 4, 7, 5, 11, 13, 19, 26, 31, 10, 12, 22, 6, 5]

In [226]: np.where(np.array(lst) == 5)

Out[226]: (array([ 3, 13], dtype=int64),)

In [228]: np.array(lst)[np.where(np.array(lst) == 5)]

Out[228]: array([5, 5])

In [229]: lst_array = np.array([10,3,5,8,12,45,21,31,45,89,20])

In [230]: np.where(lst_array % 2 == 0)

Out[230]: (array([ 0,  3,  4, 10], dtype=int64),)

In [231]: lst_array[np.where(lst_array % 2 == 0)]

Out[231]: array([10,  8, 12, 20])

In [232]: lst_array

Out[232]: array([10,  3,  5,  8, 12, 45, 21, 31, 45, 89, 20])

In [233]: lst

Out[233]: [1, 4, 7, 5, 11, 13, 19, 26, 31, 10, 12, 22, 6, 5]

In [234]: lst[4]

Out[234]: 11

In [235]: lst[10]

Out[235]: 12

In [236]: convert_lst = np.array(lst)

In [237]: convert_lst

Out[237]: array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6,  5])
```

```
In [239]: convert_lst[[4,10]]

Out[239]: array([11, 12])
```

```
In [240]: lst_array

Out[240]: array([10,  3,  5,  8, 12, 45, 21, 31, 45, 89, 20])
```

```
In [241]: lst_array[np.where(lst_array % 2 != 0)]

Out[241]: array([ 3,  5, 45, 21, 31, 45, 89])
```

```
In [242]: np.where(lst_array % 2 != 0)

Out[242]: (array([1, 2, 5, 6, 7, 8, 9], dtype=int64),)
```

```
In [243]: numbers

Out[243]: array([ 16, 273, 353, 160, 106, 164,  51, 494, 492, 369, 401, 335, 113,
                  41, 424, 497, 473, 296, 183, 296, 419, 384, 152, 277,  61, 370,
                 435, 144, 158, 238, 259, 489,  72, 374, 114, 440, 352, 457, 365,
                  88, 404,  74,  27, 473, 227, 249, 161, 294, 314, 362, 470, 166,
                 236, 102, 204, 310, 132,  62, 487, 274, 451, 383, 225, 104, 179,
                 177, 119,  90, 299,  29, 429, 177,  91, 391, 443, 166, 490,  86,
                 366, 230, 430, 290, 424,  91,  78,  69, 287,  50, 334, 211, 463,
                 105, 458,  24, 195, 479, 202, 339, 352,  65,  13, 468, 249, 498,
                  49, 348,  80, 354, 417, 426, 203,  31, 209, 351, 348, 427, 300,
                 191, 306, 383, 141, 329, 280,  32, 383, 259, 390, 419, 436, 131,
                 407,  42, 331, 362,  93, 215, 421,  55, 101, 380,  59, 409, 491,
                 302, 335, 266, 191, 404, 165, 246,  35])
```

```
In [245]: np.where((numbers >= 50) & (numbers <= 100),"Abhishek",numbers)

Out[245]: array(['16', '273', '353', '160', '106', '164', 'Abhishek', '494', '492',
                 '369', '401', '335', '113', '41', '424', '497', '473', '296',
                 '183', '296', '419', '384', '152', '277', 'Abhishek', '370', '435',
                 '144', '158', '238', '259', '489', 'Abhishek', '374', '114', '440',
                 '352', '457', '365', 'Abhishek', '404', 'Abhishek', '27', '473',
                 '227', '249', '161', '294', '314', '362', '470', '166', '236',
                 '102', '204', '310', '132', 'Abhishek', '487', '274', '451', '383',
                 '225', '104', '179', '177', '119', 'Abhishek', '299', '29', '429',
                 '177', 'Abhishek', '391', '443', '166', '490', 'Abhishek', '366',
                 '230', '430', '290', '424', 'Abhishek', 'Abhishek', 'Abhishek',
                 '287', 'Abhishek', '334', '211', '463', '105', '458', '24', '195',
                 '479', '202', '339', '352', 'Abhishek', '13', '468', '249', '498',
                 '49', '348', 'Abhishek', '354', '417', '426', '203', '31', '209',
                 '351', '348', '427', '300', '191', '306', '383', '141', '329',
                 '280', '32', '383', '259', '390', '419', '436', '131', '407', '42',
                 '331', '362', 'Abhishek', '215', '421', 'Abhishek', '101', '380',
                 'Abhishek', '409', '491', '302', '335', '266', '191', '404', '165',
                 '246', '35'], dtype='<U11')
```

In [249]:

```
np.where((numbers >= 50) & (numbers <= 100),"Abhishek",
         np.where((numbers >= 400) & (numbers <= 500),"Modi",numbers))
```

Out[249]:

```
array(['16', '273', '353', '160', '106', '164', 'Abhishek', 'Modi',
       'Modi', '369', 'Modi', '335', '113', '41', 'Modi', 'Modi', 'Modi',
       '296', '183', '296', 'Modi', '384', '152', '277', 'Abhishek',
       '370', 'Modi', '144', '158', '238', '259', 'Modi', 'Abhishek',
       '374', '114', 'Modi', '352', 'Modi', '365', 'Abhishek', 'Modi',
       'Abhishek', '27', 'Modi', '227', '249', '161', '294', '314', '362',
       'Modi', '166', '236', '102', '204', '310', '132', 'Abhishek',
       'Modi', '274', 'Modi', '383', '225', '104', '179', '177', '119',
       'Abhishek', '299', '29', 'Modi', '177', 'Abhishek', '391', 'Modi',
       '166', 'Modi', 'Abhishek', '366', '230', 'Modi', '290', 'Modi',
       'Abhishek', 'Abhishek', 'Abhishek', '287', 'Abhishek', '334',
       '211', 'Modi', '105', 'Modi', '24', '195', 'Modi', '202', '339',
       '352', 'Abhishek', '13', 'Modi', '249', 'Modi', '49', '348',
       'Abhishek', '354', 'Modi', 'Modi', '203', '31', '209', '351',
       '348', 'Modi', '300', '191', '306', '383', '141', '329', '280',
       '32', '383', '259', '390', 'Modi', 'Modi', '131', 'Modi', '42',
       '331', '362', 'Abhishek', '215', 'Modi', 'Abhishek', '101', '380',
       'Abhishek', 'Modi', 'Modi', '302', '335', '266', '191', 'Modi',
       '165', '246', '35'], dtype='<U11')
```

In [250]:

```
convert_lst
```

Out[250]:

```
array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6,  5])
```

numpy delete function

using this function we can delete an item based on the item's index.

In [251]:

```
np.delete(convert_lst,[0,-1])
```

Out[251]:

```
array([ 4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6])
```

In [252]:

```
np.delete(convert_lst,[2,6,-3])
```

Out[252]:

```
array([ 1,  4,  5, 11, 13, 26, 31, 10, 12,  6,  5])
```

In [253]:

```
convert_lst
```

Out[253]:

```
array([ 1,  4,  7,  5, 11, 13, 19, 26, 31, 10, 12, 22,  6,  5])
```

In [254]:

```
np.delete(convert_lst,np.where(convert_lst % 2 != 0))
```

Out[254]:

```
array([ 4, 26, 10, 12, 22,  6])
```

In [ ]:

```
```