Islamabad Help Desk

MUHAMMAD QASIM

RAUF UR RAHIM

MUHAMMAD ADAN

SHAHZAD AHMED

Topics we will cover

What is Numpy

Dimentionality in Numpy (creating and accessing elements)

Fancy Slicing

Concatenating and Splitting Arrays

Numpy – An Introduction

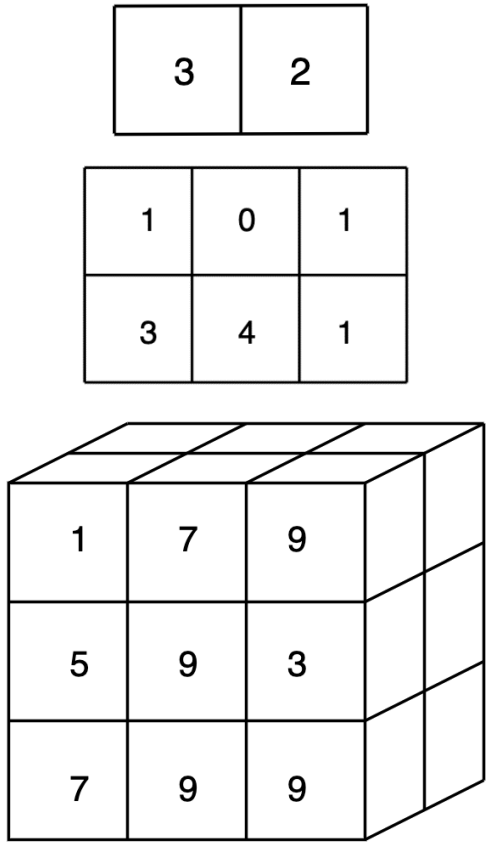
required for high performance scientific computing and data analysis

operations on entire arrays of data without having to write loops

Linear algebra, random number generation, and Fourier transform capabilities

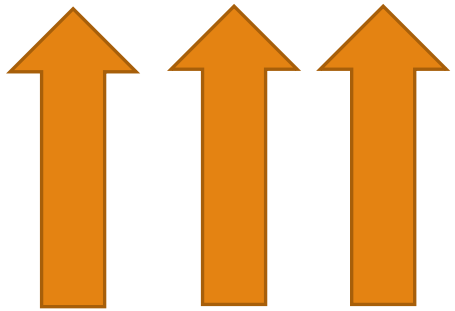
Dimentionality

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



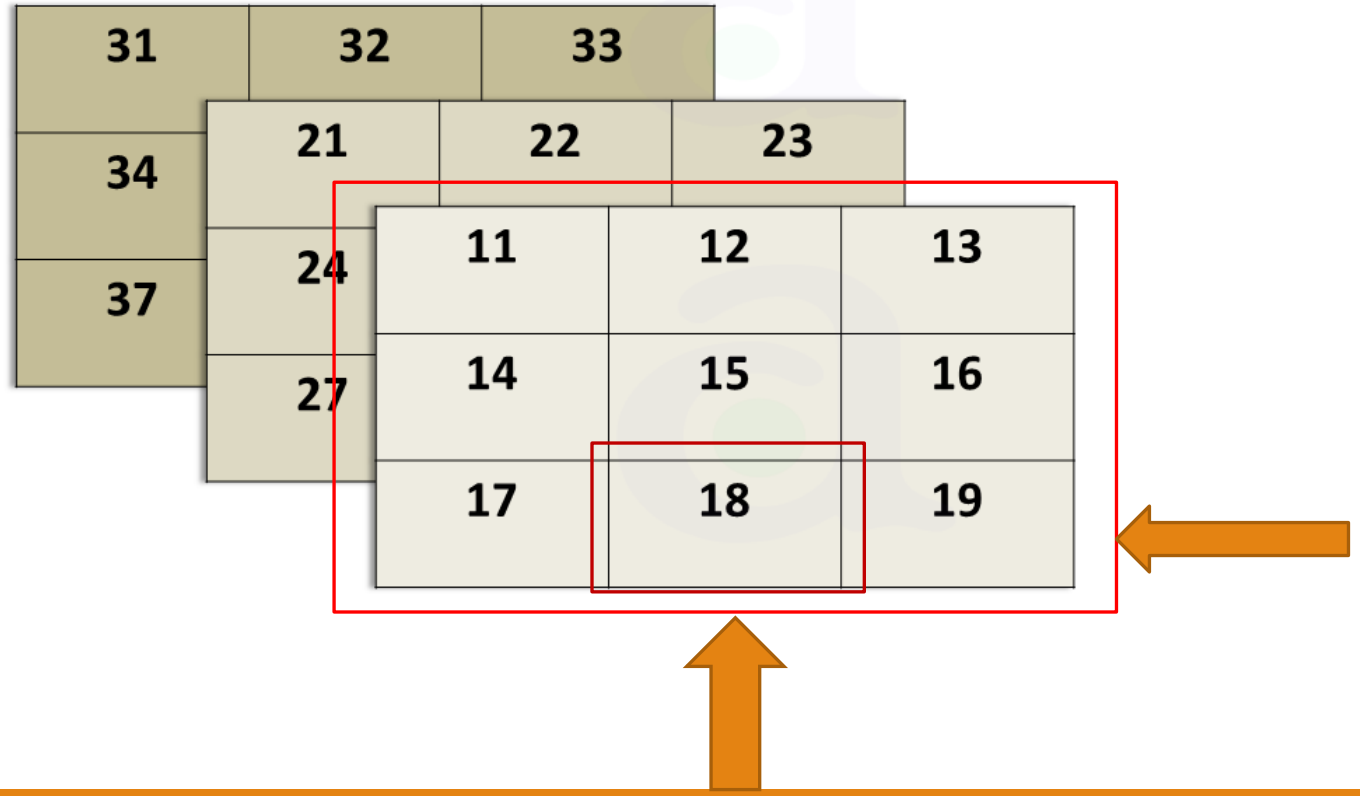
Accessing Elements of Array

a [0 , 2 , 1]



Slides	Row	Column
Axis 3	Axis 2	Axis 1

a[0,2,1]



Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],

[4 , 3 , 3 , 21],

[12 , 2 , 7 , 5],

[0 , 13 , 3 , 7],

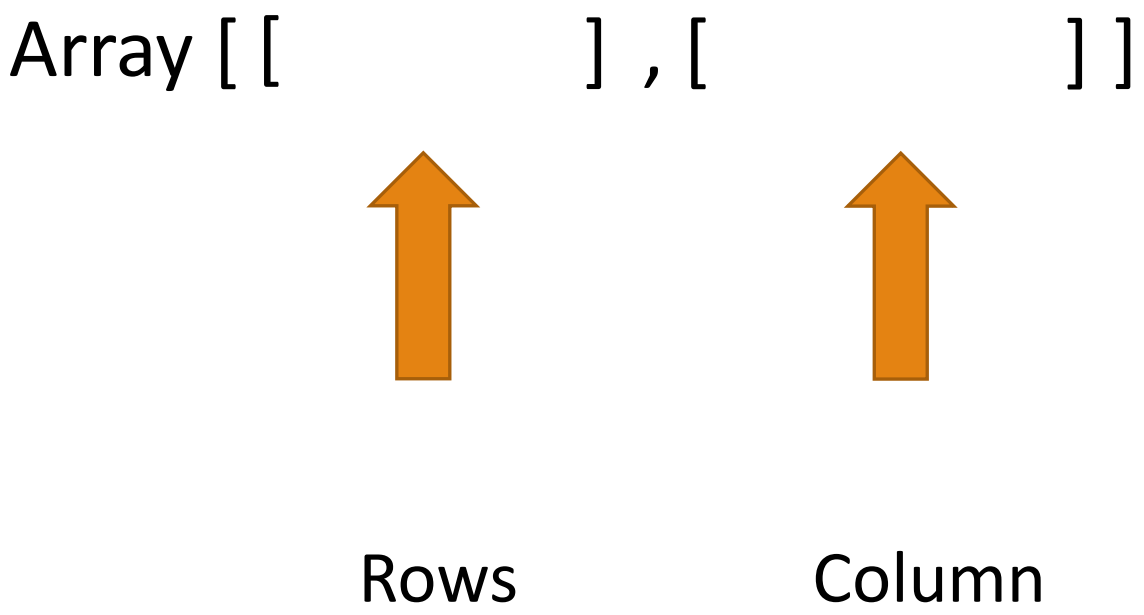
[41 , 8 , 5 , 1]])

5 X 4 array

Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],
[4 , 3 , 3 , 21],
[12 , 2 , 7 , 5],
[0 , 13 , 3 , 7],
[41 , 8 , 5 , 1]])

5 X 4 array



Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],

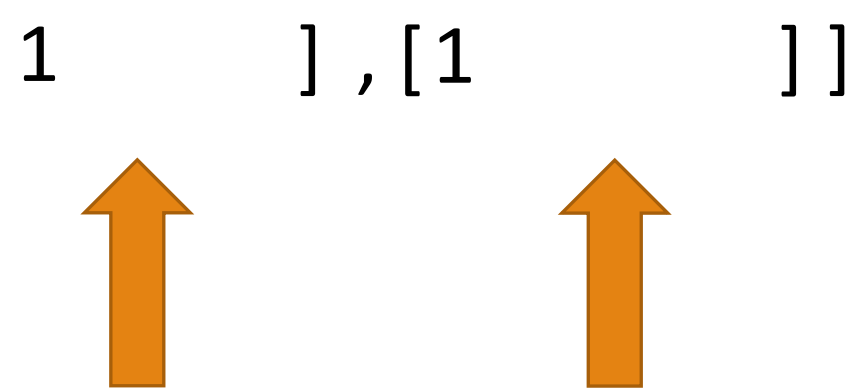
[4 , 3 , 3 , 21],

[12 , 2 , 7 , 5],

[0 , 13 , 3 , 7],

[41 , 8 , 5 , 1]])

5 X 4 array



Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],

[4 , 3 , 3 , 21],

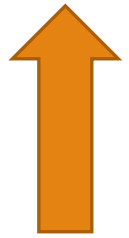
[12 , 2 , 7 , 5],

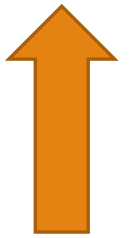
[0 , 13 , 3 , 7],

[41 , 8 , 5 , 1]])

5 X 4 array

Array [[1 , 3] , [1 , 3]]





Rows

Column

Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],

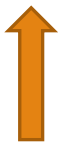
[4 , 3 , 3 , 21],

[12 , 2 , 7 , 5],

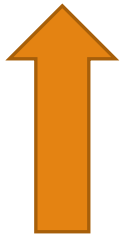
[0 , 13 , 3 , 7],

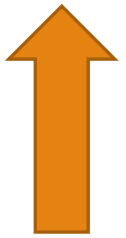
[41 , 8 , 5 , 1]])

5 X 4 array



Array [[1 , 3 , 4] , [1 , 3 , 1]]





Rows

Column

Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],

[4 , 3 , 3 , 21], Row 1

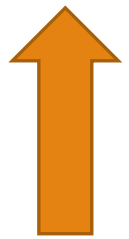
[12 , 2 , 7 , 5],

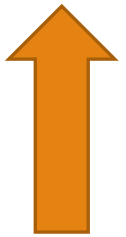
[0 , 13 , 3 , 7], Row 3

[41 , 8 , 5 , 1]]) Row 4

5 X 4 array

Array [[1 , 3 , 4]]





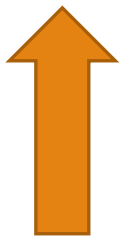
Rows Column

Fancy Slicing / Indexing

```
array( [ [ 10 , 0 , 2 , 7 ],
         [ 4 , 3 , 3 , 21 ],
         [ 12 , 2 , 7 , 5 ],
         [ 0 , 13 , 3 , 7 ],
         [ 41 , 8 , 5 , 1 ] ] )
```

5 X 4 array

Array [[1 , 3 , 4]]



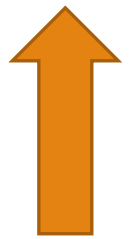
Rows

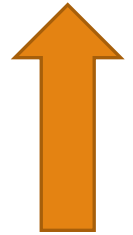
Fancy Slicing / Indexing

```
array( [ [ 10 , 0 , 2 , 7 ],
         [ 4 , 3 , 3 , 21 ],
         [ 12 , 2 , 7 , 5 ],
         [ 0 , 13 , 3 , 7 ],
         [ 41 , 8 , 5 , 1 ] ] )
```

5 X 4 array

Array [: , [1 , 3 , 4]]





Rows

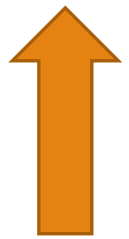
Column

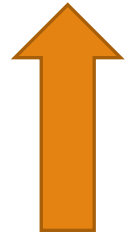
Fancy Slicing / Indexing

```
array( [ [ 10 , 0 , 2 , 7 ],
         [ 4 , 3 , 3 , 21 ],
         [ 12 , 2 , 7 , 5 ],
         [ 0 , 13 , 3 , 7 ],
         [ 41 , 8 , 5 , 1 ] ] )
```

5 X 4 array

Array [: , [1 , 3 , 2]]





Rows

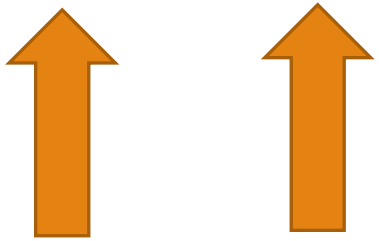
Column

Fancy Slicing / Indexing

```
array( [ [ 10 , 0 , 2 , 7 ],
         [ 4 , 3 , 3 , 21 ],
         [ 12 , 2 , 7 , 5 ],
         [ 0 , 13 , 3 , 7 ],
         [ 41 , 8 , 5 , 1 ] ] )
```

5 X 4 array

Array [: , [1 , 3 , 2]]



Rows Column

Fancy Slicing / Indexing

array([[10 , 0 , 2 , 7],

[4 , 3 , 3 , 21],

[12 , 2 , 7 , 5],

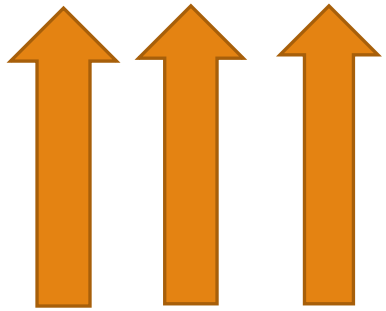
[0 , 13 , 3 , 7],

[41 , 8 , 5 , 1]])

5 X 4 array

Error !
Found 2D

Array [[1 , 3 , 4]]



Slides Axis 3

Row Axis 2

Column Axis 1

Concatenating and Splitting Arrays

Concatenation

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 0)

np.vstack ((arr1 , arr2))

np.concatenate ([arr1, arr2] , axis = 1)

np.hstack ((arr1 , arr2))

Concatenation (axis 0)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 0)

np.vstack ((arr1 , arr2))

np.concatenate ([arr1, arr2] , axis = 1)

np.hstack ((arr1 , arr2))

Concatenation (axis 1)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

np.concatenate ([arr1, arr2] , axis = 0)

np.vstack ((arr1 , arr2))

1	2	3
4	5	6
7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 1)

np.hstack ((arr1 , arr2))

Concatenation

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 0)

np.vstack ((arr1 , arr2))

1	2	3
4	5	6
7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 1)

np.hstack ((arr1 , arr2))

1	2	3	7	8	9
4	5	6	10	11	12

Difference between Concatenation & Stacking

Concatenation (axis 0)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 0)

np.stack ((arr1, arr2) , axis = 0)

Stack (axis 0)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 0)

np.stack ((arr1, arr2) , axis = 0)

Difference between Concatenation & Stacking

```
arr1 = np.array ( [ [ 1, 2, 3 ] , [ 4, 5, 6 ] ] )
arr2 = np.array ( [ [ 7, 8, 9 ] , [ 10, 11, 12 ] ] )
np.concatenate ( [arr1, arr2] , axis = 0 )
Out: array ( [ [ 1 , 2 , 3 ],
               [ 4 , 5 , 6 ],
               [ 7 , 8 , 9 ],
               [10,11,12] ] )
np.stack ( (arr1, arr2) , axis = 0 )
Out: array ( [ [ [ 1 , 2 , 3 ],
                 [ 4 , 5 , 6 ] ],

               [ [ 7 , 8 , 9 ]
                 [10,11,12] ] ] )
```

Stacking (Gulf of Alaska)

Concatenation (axis 1 / column wise)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 1)

np.stack ((arr1, arr2) , axis = 1)

Concatenation (axis 1 / Column Wise)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

1	2	3
4	5	6
7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 1)

np.stack ((arr1, arr2) , axis = 1)

Stack (Column Wise)

arr1 = np.array ([[1, 2, 3] , [4, 5, 6]])

1	2	3
4	5	6

arr2 = np.array ([[7, 8, 9] , [10, 11, 12]])

7	8	9
10	11	12

np.concatenate ([arr1, arr2] , axis = 1)

np.stack ((arr1, arr2) , axis = 1)

Difference between Concatenation & Stacking

```
arr1 = np.array ( [ [ 1, 2, 3 ] , [ 4, 5, 6 ] ] )

arr2 = np.array ( [ [ 7, 8, 9 ] , [ 10, 11, 12 ] ] )

np.concatenate ( [arr1, arr2] , axis = 1 )

Out: array ( [ [ 1 , 2 , 3 , 4 , 5 , 6 ],
                    [ 7 , 8 , 9 ,10,11,12] ] )

np.stack ( (arr1, arr2) , axis = 1 )

Out: array ( [ [ [ 1 , 2 , 3 ],
                     [ 7 , 8 , 9 ] ],

                    [ [ 4 , 5 , 6 ]
                     [10,11,12] ] ] )
```

End of Numpy

Pandas

Creating Series

obj1 = pd.series ([1 , 2 , 3 , 4 , 5])

Out:

0	1
1	2
2	3
3	4
4	5

Values

Index generated on auto

Creating Series

obj1 = pd.series ([1 , 2 , 3 , 4 , 5] , index = [“A” , “B” , “C” , “D” , “E”])

Out:

A	1
B	2
C	3
D	4
E	5

Values

Index generated on auto

Creating Series

obj1 = pd.series ([1 , 2 , 3 , 4 , 5] , index = [“A” , “B” , “C” , “D” , “E”])

Out:

A	1
B	2
C	3
D	4
E	5

obj1[“D”] = 10

obj1[[“C”, “A” , “D”]]

C	3
A	1
D	10

Creating DataFrame

	Roll No	Name	Program	Shift
0	1	Ali	BS	Morning
1	2	Mahmood	MS	Morning
2	3	Fatima	BS	Evening
3	4	Kashif	PhD	Morning
4	5	Asif	BS	Evening

First: 2D array is required

```
data = [ [1 , Ali , BS , Morning] ,
         [2 , Mahmood , MS, Morning] ,
         [3 , Fatima , BS , Evening] ,
         [4 , Kashif , PhD , Morning] ,
         [5 , Asif , BS , Evening ] ]
```

```
df1 = pd.DataFrame ( data )
```

	Roll No	Name	Program	Shift
0	1	Ali	BS	Morning
1	2	Mahmood	MS	Morning
2	3	Fatima	BS	Evening
3	4	Kashif	PhD	Morning
4	5	Asif	BS	Evening

	0	1	2	3
0	1	Ali	BS	Morning
1	2	Mahmood	MS	Morning
2	3	Fatima	BS	Evening
3	4	Kashif	PhD	Morning
4	5	Asif	BS	Evening

Customizing Columns Name

```
data = [ [1 , Ali , BS , Morning] ,
         [2 , Mahmood , MS, Morning] ,
         [3 , Fatima , BS , Evening] ,
         [4 , Kashif , PhD , Morning] ,
         [5 , Asif , BS , Evening ] ]
```

	Roll No	Name	Program	Shift
0	1	Ali	BS	Morning
1	2	Mahmood	MS	Morning
2	3	Fatima	BS	Evening
3	4	Kashif	PhD	Morning
4	5	Asif	BS	Evening

```
df1 = pd.DataFrame ( data )
```

```
df1 = pd.DataFrame ( data , column =  [“Roll No”,“Name”,“Program”,“Shift”])
```

	Roll No	Name	Program	Shift
0	1	Ali	BS	Morning
1	2	Mahmood	MS	Morning
2	3	Fatima	BS	Evening
3	4	Kashif	PhD	Morning
4	5	Asif	BS	Evening

Customizing Rows Name

```
data = [ [1 , Ali , BS , Morning] ,
         [2 , Mahmood , MS, Morning] ,
         [3 , Fatima , BS , Evening] ,
         [4 , Kashif , PhD , Morning] ,
         [5 , Asif , BS , Evening ] ]
```

	Roll No	Name	Program	Shift
0	1	Ali	BS	Morning
1	2	Mahmood	MS	Morning
2	3	Fatima	BS	Evening
3	4	Kashif	PhD	Morning
4	5	Asif	BS	Evening

```
df1 = pd.DataFrame ( data )
```

```
df1 = pd.DataFrame ( data , column =  [“Roll No”,“Name”,“Program”,“Shift”]
                                index =  [ “a” , ”b” , ”c” , ”d” , ”e” ] )
```

	Roll No	Name	Program	Shift
a	1	Ali	BS	Morning
b	2	Mahmood	MS	Morning
c	3	Fatima	BS	Evening
d	4	Kashif	PhD	Morning
e	5	Asif	BS	Evening

Concatenation

	ID	Name	Course	Fee	update course code
0	1	Ali	CNC	1500.0	0
1	2	Asif	A.I	1500.0	1
2	3	Hamza	A.I	3000.0	1
3	4	Kashif	AIOT	NaN	2
4	5	Ali	CNC	3000.0	0

	A	B	C	D	E
0	65	28	5	46	79
1	29	25	45	94	67
2	2	20	91	93	18
3	79	76	77	25	28
4	69	64	78	47	58

df = pd.concat ([df1, df2], axis=0)

	ID	Name	Course	Fee	update course code	A	B	C	D	E
0	Ali	CNC	1500.0	0	NaN	NaN	NaN	NaN	NaN
1	Asif	A.I	1500.0	1	NaN	NaN	NaN	NaN	NaN
2	Hamza	A.I	3000.0	1	NaN	NaN	NaN	NaN	NaN
3	Kashif	AIOT	NaN	2	NaN	NaN	NaN	NaN	NaN
4	Ali	CNC	3000.0	0	NaN	NaN	NaN	NaN	NaN
0	NaN	NaN	NaN	NaN	65.0	28.0	5.0	46.0	79.0
1	NaN	NaN	NaN	NaN	29.0	25.0	45.0	94.0	67.0
2	NaN	NaN	NaN	NaN	2.0	20.0	91.0	93.0	18.0
3	NaN	NaN	NaN	NaN	79.0	76.0	77.0	25.0	28.0
4	NaN	NaN	NaN	NaN	69.0	64.0	78.0	47.0	58.0

df = pd.concat ([df1, df2], axis=1)

ID	Name	Course	Fee	update course code	A	B	C	D	E	
0	Ali	CNC	1500.0	0	0.0	65.0	28.0	5.0	46.0	79.0
1	Asif	A.I	1500.0	1	1.0	29.0	25.0	45.0	94.0	67.0
2	Hamza	A.I	3000.0	1	1.0	2.0	20.0	91.0	93.0	18.0
3	Kashif	AIOT	NaN	2	2.0	79.0	76.0	77.0	25.0	28.0
4	Ali	CNC	3000.0	0	0.0	69.0	64.0	78.0	47.0	58.0

Sorting

Sort By Index

Sort By Values

- Ascending
- Descending

3 Rules

1: Choose Index or Values

Rows no / Columns no itself

Other values than Rows or Column

2: Row or Column

3: Ascending or Descending

Sort By Index

df.sort_index(axis = 0, ascending = False)

Rule 1 Rule 2

	ID	Name	Course	Fee	A	B	C	D	E
4	5	Ali	CNC	3000.0	54	4	83	81	4
3	4	Kashif	AIOT	NaN	56	74	87	74	66
2	3	Hamza	A.I	3000.0	67	26	30	19	2
1	2	Asif	A.I	1500.0	57	11	42	74	77
0	1	Ali	CNC	1500.0	68	30	93	63	46

Sort By Index

df.sort_index(axis = 0, ascending = False)

Rule 1 Rule 2 Rule 3

	ID	Name	Course	Fee	A	B	C	D	E
4	5	Ali	CNC	3000.0	54	4	83	81	4
3	4	Kashif	AIOT	NaN	56	74	87	74	66
2	3	Hamza	A.I	3000.0	67	26	30	19	2
1	2	Asif	A.I	1500.0	57	11	42	74	77
0	1	Ali	CNC	1500.0	68	30	93	63	46

Sort By Values

df.sort_values (“Course”, ascending = False)

	ID	Name	Course	Fee	A	B	C	D	E
0	1	Ali	CNC	1500.0	68	30	93	63	46
4	5	Ali	CNC	3000.0	54	4	83	81	4
3	4	Kashif	AIOT	NaN	56	74	87	74	66
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2

df.sort_values ([“Course”, “A”] , ascending = [False, True])

	ID	Name	Course	Fee	A	B	C	D	E
0	1	Ali	CNC	1500.0	68	30	93	63	46
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2
3	4	Kashif	AIOT	NaN	56	74	87	74	66
4	5	Ali	CNC	3000.0	54	4	83	81	4

4	5	Ali	CNC	3000.0	54	4	83	81	4

df.sort_values ([“Course”, “A”] , ascending = [False, True])

	ID	Name	Course	Fee	A	B	C	D	E
0	1	Ali	CNC	1500.0	68	30	93	63	46
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2
3	4	Kashif	AIOT	NaN	56	74	87	74	66
4	5	Ali	CNC	3000.0	54	4	83	81	4
4	5	Ali	CNC	3000.0	54	4	83	81	4
0	1	Ali	CNC	1500.0	68	30	93	63	46

df.sort_values ([“Course”, “A”] , ascending = [False, True])

	ID	Name	Course	Fee	A	B	C	D	E
0	1	Ali	CNC	1500.0	68	30	93	63	46
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2
3	4	Kashif	AIOT	NaN	56	74	87	74	66
4	5	Ali	CNC	3000.0	54	4	83	81	4
4	5	Ali	CNC	3000.0	54	4	83	81	4
0	1	Ali	CNC	1500.0	68	30	93	63	46
3	4	Kashif	AIOT	NaN	56	74	87	74	66

df.sort_values ([“Course”, “A”] , ascending = [False, True])

	ID	Name	Course	Fee	A	B	C	D	E
0	1	Ali	CNC	1500.0	68	30	93	63	46
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2
3	4	Kashif	AIOT	NaN	56	74	87	74	66
4	5	Ali	CNC	3000.0	54	4	83	81	4

4	5	Ali	CNC	3000.0	54	4	83	81	4
0	1	Ali	CNC	1500.0	68	30	93	63	46
3	4	Kashif	AIOT	NaN	56	74	87	74	66
1	2	Asif	A.I	1500.0	57	11	42	74	77

df.sort_values ([“Course”, “A”] , ascending = [False, True])

	ID	Name	Course	Fee	A	B	C	D	E
0	1	Ali	CNC	1500.0	68	30	93	63	46
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2
3	4	Kashif	AIOT	NaN	56	74	87	74	66
4	5	Ali	CNC	3000.0	54	4	83	81	4

4	5	Ali	CNC	3000.0	54	4	83	81	4
0	1	Ali	CNC	1500.0	68	30	93	63	46
3	4	Kashif	AIOT	NaN	56	74	87	74	66
1	2	Asif	A.I	1500.0	57	11	42	74	77
2	3	Hamza	A.I	3000.0	67	26	30	19	2

Queries?

WhatsApp

0345-7770757

Rauf ur Rahim