In [40]:
```
import pandas as pd
```

In [41]:
```
data = [["Saurabh", 20], ["Lionel", 36]]
df = pd.DataFrame(data, columns = ["Name", "Age"], index = ["rank1", '
```

In [42]:
```
df
```

Out[42]:

	Name	Age
rank1	Saurabh	20
rank2	Lionel	36

In [43]:
```
df.shape
```

Out[43]:
```
(2, 2)
```

In [44]:
```
df.dtypes
```

Out[44]:
```
Name    object
Age      int64
dtype: object
```

In [45]:
```
df.T
```

Out[45]:

	rank1	rank2
Name	Saurabh	Lionel
Age	20	36

In [46]:
```
df.columns
```

Out[46]:
```
Index(['Name', 'Age'], dtype='object')
```

In [47]:
```
df.index
```

Out[47]:
```
Index(['rank1', 'rank2'], dtype='object')
```

In [48]:
```
df.sort_index
```

Out[48]:
```
<bound method DataFrame.sort_index of          Name  Age
rank1  Saurabh   20
rank2   Lionel   36>
```

In [49]:
```
df.sort_values
```

Out[49]:
```
<bound method DataFrame.sort_values of          Name  Age
rank1  Saurabh   20
rank2   Lionel   36>
```

In [50]:
```
df
```

Out[50]:

	Name	Age
rank1	Saurabh	20
rank2	Lionel	36

In [51]:
```
df.iloc[1]
```

Out[51]:
```
Name    Lionel
Age         36
Name: rank2, dtype: object
```

In [52]:
```
df.iloc[:2]
```

Out[52]:

	Name	Age
rank1	Saurabh	20
rank2	Lionel	36

In [53]:
```
df.iloc[0, 0]
```

Out[53]:
```
'Saurabh'
```

In [54]:
```
df[['Name', 'Age']]
```

Out[54]:

	Name	Age
rank1	Saurabh	20
rank2	Lionel	36

In [55]:
```
df[df.Age > 20]
```

Out[55]:

	Name	Age
rank2	Lionel	36

In [56]:
```
df.drop('Age', axis = 1, inplace = True)
```

In [57]:
```
df
```

Out[57]:

	Name
rank1	Saurabh
rank2	Lionel

In [58]:
```
df.drop('rank1', axis = 0, inplace = True)
```

In [59]:
```
df
```

Out[59]:

	Name
rank2	Lionel

In [60]:
```
name = "Male"
age = 20
print("Saurabh is %s and age is %d!\n" % (name, age))
```

```
Saurabh is Male and age is 20!
```

In [61]:
```
n = 5
i = 0
while i < n:
    print(i)
    i += 1
else:
    print("Exiting while loop!")
```

```
0
1
2
3
4
Exiting while loop!
```

In [62]:
```
n = int(input("Enter number of elements: "))
nums = []
for i in range(n):
    nums.append(int(input("Enter a number: ")))
print("Entered numbers: ", nums)

i = 0
print("The negative numbers are:")
while i < n:
    if nums[i] < 0:
        print(nums[i])
    i += 1
```

```
Enter number of elements: 5
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: -1
Enter a number: -3
Entered numbers:  [1, 2, 3, -1, -3]
The negative numbers are:
-1
-3
```

In [65]:
```
import pandas as pd

data = {"Name" : ["Saurabh", "Lionel", "Cristiano"],
        "Height" : [180, 176, 182],
        "Qualification" : ["B.Tech", "Master's", "MBA"]}
df = pd.DataFrame(data, columns = ["Name", "Height", "Qualification"],

address = ["Hyderabad", "Buenos Aires", "Lisbon"]
df["Address"] = address

df
```

Out[65]:

	Name	Height	Qualification	Address
P1	Saurabh	180	B.Tech	Hyderabad
P2	Lionel	176	Master's	Buenos Aires
P3	Cristiano	182	MBA	Lisbon

In [74]:
```
import pandas as pd

data = {"Name" : ["Saurabh", "Lionel", "Cristiano"],
        "Height" : [180, 176, 182],
        "Qualification" : ["B.Tech", "Master's", "MBA"]}
df = pd.DataFrame(data, columns = ["Name", "Height", "Qualification"],

address = ["Hyderabad", "Buenos Aires", "Lisbon"]
df.insert(1, "Address", address)

df
```

Out[74]:

	Name	Address	Height	Qualification
P1	Saurabh	Hyderabad	180	B.Tech
P2	Lionel	Buenos Aires	176	Master's
P3	Cristiano	Lisbon	182	MBA

In [76]:
```
import pandas as pd

data1 = {"Name" : ["Ram", "Diya", "Chandan", "James", "Alice"]}
df1 = pd.DataFrame(data1)

data2 = {"Maths" : [80.0, 90.0, 77.5, 87.5, 86.5],
         "Physics" : [81.0, 94.0, 74.5, 83.0, 82.5],
         "Chemistry" : [91.5, 86.5, 85.5, 90.0, 91.0],
         "Biology" : [82.5, 83.5, 84.5, 85.0, 93.0]}
df2 = pd.DataFrame(data2)

df3 = pd.concat((df1, df2), axis = 1)
df3["Total"] = df3[["Maths", "Physics", "Chemistry", "Biology"]].sum(a
df3
```

Out[76]:

	Name	Maths	Physics	Chemistry	Biology	Total
0	Ram	80.0	81.0	91.5	82.5	335.0
1	Diya	90.0	94.0	86.5	83.5	354.0
2	Chandan	77.5	74.5	85.5	84.5	322.0
3	James	87.5	83.0	90.0	85.0	345.5
4	Alice	86.5	82.5	91.0	93.0	353.0

In [87]:
```
import pandas as pd

data = {"Name" : ["Annie", "Diya", "Charles", "James", "Emily"],
        "Quiz_1/10" : [8.0, 9.0, 7.5, 8.5, 6.5],
        "In_Sem_1/15" : [11.0, 14.0, 14.5, 13.0, 12.5],
        "Quiz_2/10" : [9.5, 6.5, 8.5, 9.0, 9.0],
        "In_Sem_2/15" : [12.5, 13.5, 14.5, 15.0, 13.0]}
df = pd.DataFrame(data)
df["Total"] = df[["Quiz_1/10", "In_Sem_1/15", "Quiz_2/10", "In_Sem_2/1
df.loc["Mean"] = df[["Quiz_1/10", "In_Sem_1/15", "Quiz_2/10", "In_Sem_
df
```

Out[87]:

	Name	Quiz_1/10	In_Sem_1/15	Quiz_2/10	In_Sem_2/15	Total
0	Annie	8.0	11.0	9.5	12.5	41.0
1	Diya	9.0	14.0	6.5	13.5	43.0
2	Charles	7.5	14.5	8.5	14.5	45.0
3	James	8.5	13.0	9.0	15.0	45.5
4	Emily	6.5	12.5	9.0	13.0	41.0
Mean	NaN	7.9	13.0	8.5	13.7	NaN

In [90]:
```
n = int(input("Enter a number: "))
factors = []
for i in range(1, n+1):
    if n % i == 0:
        factors.append(i)
print("Factors of %d are: " % (n), factors)
```

```
Enter a number: 12
Factors of 12 are:  [1, 2, 3, 4, 6, 12]
```

In [92]:
```
import numpy as np

rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))
matrix = np.empty((rows, cols), dtype = int)

for i in range(rows):
    for j in range(cols):
        matrix[i][j] = int(input("Enter a number: "))

print("Entered Matrix:")
print(matrix)

row_sums = np.sum(matrix, axis = 1)
col_sums = np.sum(matrix, axis = 0)

print("Row Sums : ")
for i in range(rows):
    print("Row %d : " % (i), row_sums[i])
print("Column Sums : ")
for i in range(cols):
    print("Column %d : " % (i), col_sums[i])
```

```
Enter the number of rows: 2
Enter the number of columns: 3
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Entered Matrix:
[[1 2 3]
 [4 5 6]]
Row Sums :
Row 0 :  6
Row 1 :  15
Column Sums :
Column 0 :  5
Column 1 :  7
Column 2 :  9
```

In [98]:
```
nums = [1.2, 3.4, 5.6, 7.8, 9.0]
arr = np.array(nums, dtype = float)
print(arr)
```

```
[1.2 3.4 5.6 7.8 9. ]
```

In [96]:
```
t = (1, 2, 3, 4, 5)
arr = np.array(t)
print(arr)
```

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

In [99]:
```
matrix = np.zeros((3, 4), dtype = int)
print(matrix)
```

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

In [101]:
```
nums = np.linspace(0, 20, 5)
print(nums)
```

```
[ 0.  5. 10. 15. 20.]
```

In [103]:
```
matrix = matrix.reshape((2, 2, 3))
print(matrix)
```

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

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

In [105]:
```
import numpy as np

rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))
matrix = np.empty((rows, cols), dtype = int)

for i in range(rows):
    for j in range(cols):
        matrix[i][j] = int(input("Enter a number: "))

print("Entered Matrix:")
print(matrix)

row_min = np.min(matrix, axis = 1)
row_max = np.max(matrix, axis = 1)
col_min = np.min(matrix, axis = 0)
col_max = np.max(matrix, axis = 0)

print("Row Minimums: ", row_min)
print("Row Maximums: ", row_max)
print("Column Minimums: ", col_min)
print("Column Maximums: ", col_max)
```

```
Enter the number of rows: 3
Enter the number of columns: 2
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Entered Matrix:
[[1 2]
 [3 4]
 [5 6]]
Row Minimums:  [1 3 5]
Row Maximums:  [2 4 6]
Column Minimums:  [1 2]
Column Maximums:  [5 6]
```

In [107]:
```
rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))

matrix = np.empty((rows, cols), dtype = int)

for i in range(rows):
    for j in range(cols):
        matrix[i][j] = int(input("Enter a number: "))

print("Entered Matrix:")
print(matrix)

transposed_matrix = np.transpose(matrix)
print("Transposed Matrix:")
print(transposed_matrix)
```

```
Enter the number of rows: 2
Enter the number of columns: 3
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Entered Matrix:
[[1 2 3]
 [4 5 6]]
Transposed Matrix:
[[1 4]
 [2 5]
 [3 6]]
```

In [111]:
```
rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))

m1 = np.empty((rows, cols), dtype = int)
m2 = np.empty((rows, cols), dtype = int)
m3 = np.empty((rows, cols), dtype = int)

print("MATRIX 1:")
for i in range(rows):
    for j in range(cols):
        m1[i][j] = int(input("Enter a number: "))

print("Entered Matrix 1:")
print(m1)

print("MATRIX 2:")
for i in range(rows):
    for j in range(cols):
        m2[i][j] = int(input("Enter a number: "))

print("Entered Matrix 2:")
print(m2)

m3 = m1 + m2
print("Resultant Matrix:")
print(m3)
```

```
Enter the number of rows: 2
Enter the number of columns: 3
MATRIX 1:
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Entered Matrix 1:
[[1 2 3]
 [4 5 6]]
MATRIX 2:
Enter a number: 6
Enter a number: 5
Enter a number: 4
Enter a number: 3
Enter a number: 2
Enter a number: 1
Entered Matrix 2:
[[6 5 4]
 [3 2 1]]
Resultant Matrix:
[[7 7 7]
 [7 7 7]]
```

In [112]:
```
rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))

m1 = np.empty((rows, cols), dtype = int)
m2 = np.empty((rows, cols), dtype = int)
m3 = np.empty((rows, cols), dtype = int)

print("MATRIX 1:")
for i in range(rows):
    for j in range(cols):
        m1[i][j] = int(input("Enter a number: "))

print("Entered Matrix 1:")
print(m1)

print("MATRIX 2:")
for i in range(rows):
    for j in range(cols):
        m2[i][j] = int(input("Enter a number: "))

print("Entered Matrix 2:")
print(m2)

m3 = m1 * m2
print("Resultant Matrix:")
print(m3)
```

```
Enter the number of rows: 2
Enter the number of columns: 3
MATRIX 1:
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Entered Matrix 1:
[[1 2 3]
 [4 5 6]]
MATRIX 2:
Enter a number: 6
Enter a number: 5
Enter a number: 4
Enter a number: 3
Enter a number: 2
Enter a number: 1
Entered Matrix 2:
[[6 5 4]
 [3 2 1]]
Resultant Matrix:
[[ 6 10 12]
 [12 10  6]]
```

In [ ]:

In [ ]:

In [ ]:
```
1
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: