1]MATRIX ADDITION

rows = int(input("Enter the Number of rows : " ))

column = int(input("Enter the Number of Columns: "))

print("Enter the elements of First Matrix:")

matrix\_a= [[int(input()) for i in range(column)] for i in range(rows)]

print("First Matrix is: ")

for n in matrix\_a:

    print(n)

print("Enter the elements of Second Matrix:")

matrix\_b= [[int(input()) for i in range(column)] for i in range(rows)]

for n in matrix\_b:

    print(n)

result=[[0 for i in range(column)] for i in range(rows)]

for i in range(rows):

    for j in range(column):

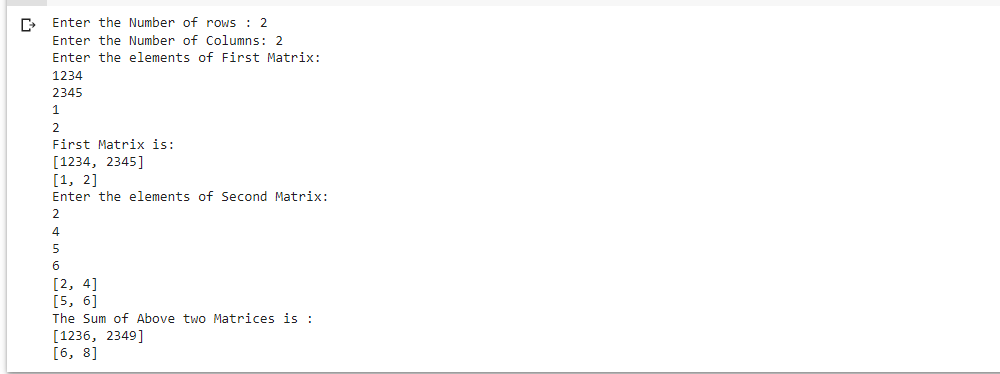
        result[i][j] = matrix\_a[i][j]+matrix\_b[i][j]

print("The Sum of Above two Matrices is : ")

for r in result:

    print(r)

output



2]matrix multiplication

rows\_a = int(input("Enter the Number of rows  for the first matrix: " ))

column\_a = int(input("Enter the Number of Columns for the first matrix: "))

print("Enter the elements of First Matrix:")

matrix\_a= [[int(input()) for i in range(column\_a)] for i in range(rows\_a)]

print("First Matrix is: ")

for n in matrix\_a:

    print(n)

#the number of columns of first matrix is equal to the number of rows of second matrix

column\_b = int(input("Enter the Number of Columns for the second matrix: "))

print("Enter the elements of Second Matrix:")

matrix\_b= [[int(input()) for i in range(column\_b)] for i in range(column\_a)]

for n in matrix\_b:

    print(n)

result=[[0 for i in range(column\_b)] for i in range(rows\_a)]

for i in range(len(matrix\_a)):

    for j in range(len(matrix\_b[0])):

        for k in range(len(matrix\_b)):

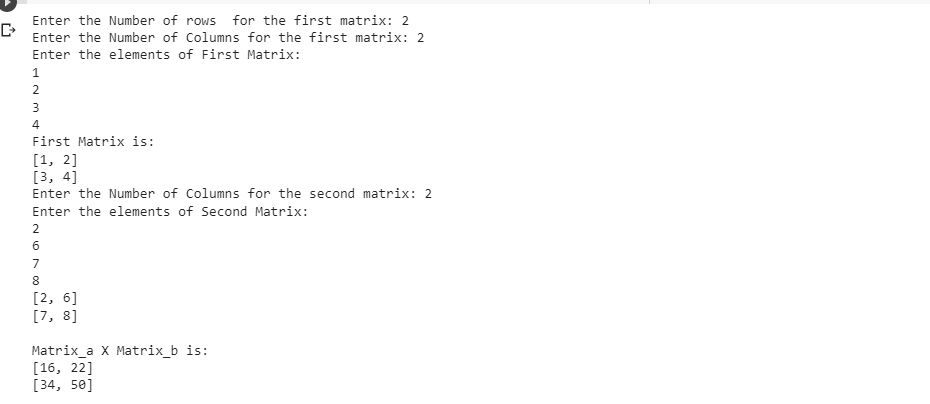
            result [i][j]+=matrix\_a[i][k]\*matrix\_b[k][j]

print("\nMatrix\_a X Matrix\_b is: ")

for r in result:

    print(r)

*output*



3]determinant of the matrix

import numpy

a= numpy.matrix('1 2; 3 4')

print("Original matrix is:")

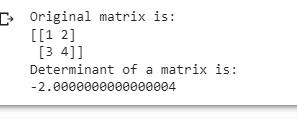
print(a)

d=numpy.linalg.det(a)

print("Determinant of a matrix is:")

print(d)

output



4]identity of the matrix

a = [

        [1, 0, 0],

        [0, 1, 0],

        [0, 0, 1]

    ];

flag = True;

rows = len(a);

cols = len(a[0]);

if(rows != cols):

    print("Matrix should be a square matrix");

else:

    for i in range(0, rows):

        for j in range(0, cols):

            if(i == j and a[i][j] != 1):

                flag = False;

                break;

            if(i != j and a[i][j] != 0):

                flag = False;

                break;

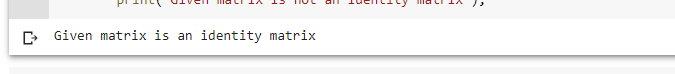
    if(flag):

        print("Given matrix is an identity matrix");

    else:

        print("Given matrix is not an identity matrix");

output



Pandas

csv\_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

# using the attribute information as the column names

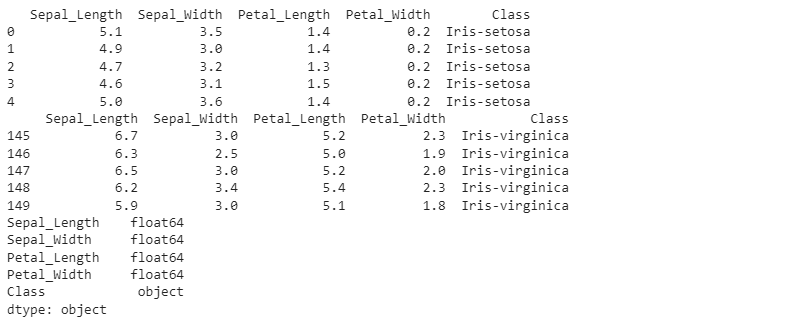
col\_names = ['Sepal\_Length','Sepal\_Width','Petal\_Length','Petal\_Width','Class']

iris =  pd.read\_csv(csv\_url, names = col\_names)

print(iris.head())

print(iris.tail())

print(iris.dtypes)



A DataFrame

csv\_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

iris = pd.read\_csv(csv\_url, header = None)

col\_names = ['Sepal\_Length','Sepal\_Width','Petal\_Length','Petal\_Width','Species']

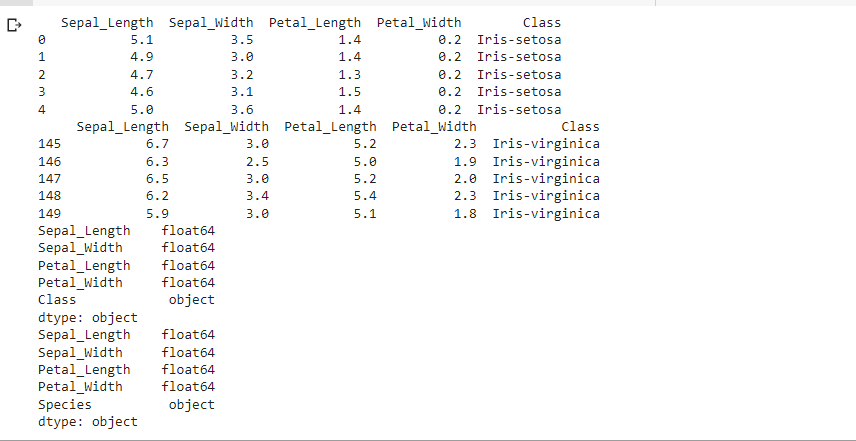
# read in the dataset from the UCI Machine Learning Repository link and specify column names to use

# save as iris\_df

iris =  pd.read\_csv(csv\_url, names = col\_names)

# The columns of the resulting DataFrame have different dtypes.

iris.dtypes



### **Viewing Data**

iris.head()

# View the index of the DataFrame

iris.index

RangeIndex(start=0, stop=150, step=1)

# View the columns of the DataFrame

iris.columns

Index(['Sepal\_Length', 'Sepal\_Width', 'Petal\_Length', 'Petal\_Width',

'Species'],

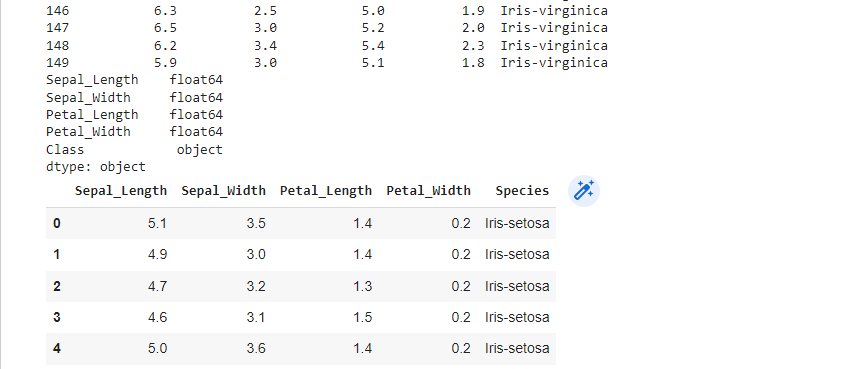
dtype='object')

# sorting by an axis

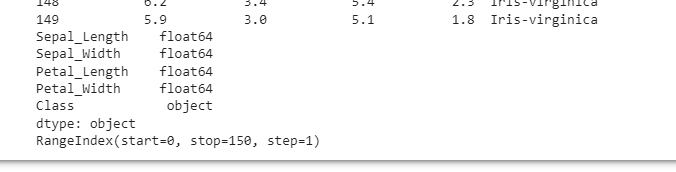
iris.sort\_index(axis=1, ascending=False).head(10)

# sorting by values

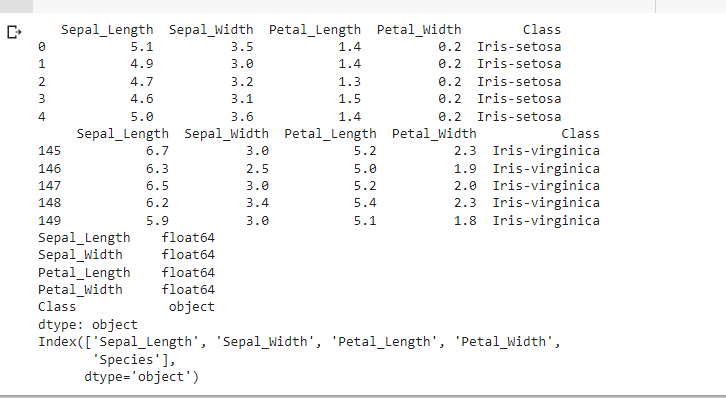
iris.sort\_values(by='Petal\_Width').head(10)



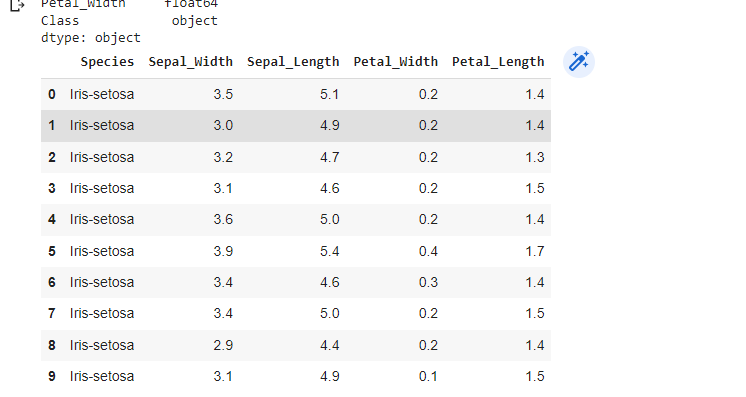
iris.head()



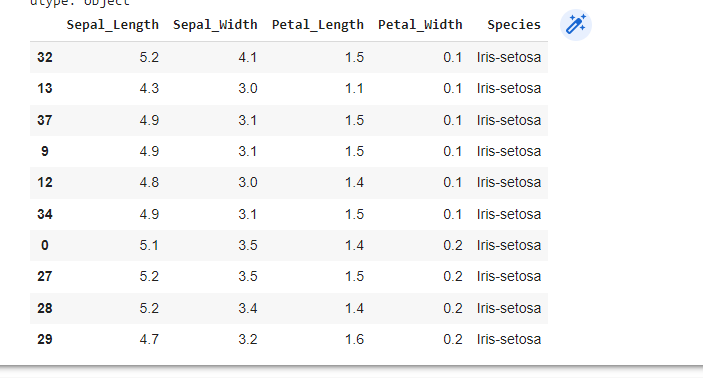
iris.index



iris.sort\_index(axis=1, ascending=False).head(10)



iris.sort\_values(by='Petal\_Width').head(10)



### **Selection**

#### Getting