**Lab Cycle 2**

1.)Create a three dimensional array specifying float data type and print it.

import numpy as np

print("Athira Anil-21MCA013")

arr1=np.array([

[[2,3,4],[7,4,5],[55,9,9]],

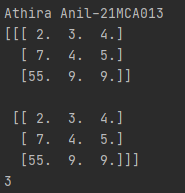
[[2, 3, 4], [7, 4, 5], [55, 9, 9]]

],dtype=float)

print(arr1)

print(arr1.ndim)

output



2). Create a 2 dimensional array (2X3) with elements belonging to complex data

type and print it. Also display

a. the no: of rows and columns

b. dimension of an array

c. reshape the same array to 3X2

import numpy as np

print("Athira Anil : 21MCA013")

x=np.array([[1,3,5],[6,8,9]],dtype=complex)

print(x)

rows, columns = x.shape

print("Rows = ",rows)

print("Columns = ", columns)

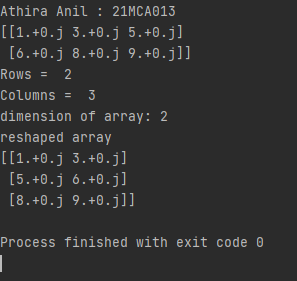
print("dimension of array:",x.ndim)

print("reshaped array")

newarr = x.reshape(3, 2)

print(newarr)

output



3). Familiarize with the functions to create

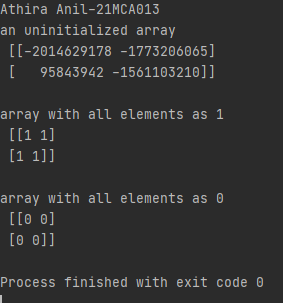
a) an uninitialized array

b) array with all elements as 1,

c) all elements as 0

print("Athira Anil-21MCA013")  
import numpy as np  
arr=np.empty([2,2],dtype="int")  
print("an uninitialized array\n",arr)  
arr=np.ones([2,2],dtype="int")  
print("\narray with all elements as 1\n",arr)  
arr=np.zeros([2,2],dtype="int")  
print("\narray with all elements as 0\n",arr)

output



4). Create an one dimensional array using the arrange function containing 10 elements.

Display

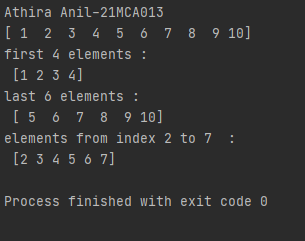
a. First 4 elements

b. Last 6 elements

c. Elements from index 2 to 7

print("Athira Anil-21MCA013")  
import numpy as np  
arr=np.arange(start=1,stop=11,step=1,dtype="int")  
print(arr)  
print("first 4 elements :\n",arr[:4])  
print("last 6 elements :\n",arr[-6:])  
print("elements from index 2 to 7 :\n",arr[1:7])

output



5) Create an 1D array with arange containing first 15 even numbers as elements

a. Elements from index 2 to 8 with step 2(also demonstrate the same

using slice function)

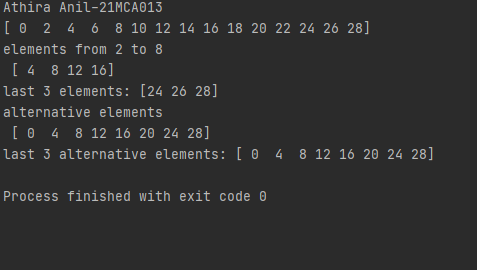
b. Last 3 elements of the array using negative index

c. Alternate elements of the array

d. Display the last 3 alternate elements

print("Athira Anil-21MCA013")  
import numpy as np  
ar=np.arange(start=0,stop=30,step=2)  
print(ar)  
print("elements from 2 to 8\n",ar[2:9:2])  
print("last 3 elements:",ar[-3:])  
print("alternative elements\n",ar[0:30:2])  
print("last 3 alternative elements:",ar[:30:2])

output



6). Create a 2 Dimensional array with 4 rows and 4 columns.

a. Display all elements excluding the first row

b. Display all elements excluding the last column

c. Display the elements of 1 st and 2 nd column in 2 nd and 3 rd row

d. Display the elements of 2 nd and 3 rd column

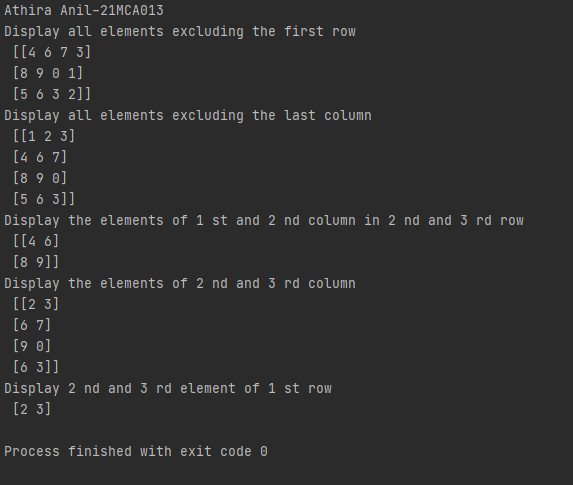
e. Display 2 nd and 3 rd element of 1 st row

f. Display the elements from indices 4 to 10 in descending order(use

–values)

print("Athira Anil-21MCA013")  
  
import numpy as np  
ar=np.array([[1,2,3,4],  
  
 [4,6,7,3],  
  
 [8,9,0,1],  
  
 [5,6,3,2]  
  
 ])  
print("Display all elements excluding the first row\n",ar[1:4])  
  
print("Display all elements excluding the last column\n",ar[:,0:3])  
  
print("Display the elements of 1 st and 2 nd column in 2 nd and 3 rd row\n",ar[1:3,0:2])  
  
print("Display the elements of 2 nd and 3 rd column\n",ar[:,1:3])  
  
print("Display 2 nd and 3 rd element of 1 st row\n",ar[0,1:3])

output



7. Create two 2D arrays using array object and

a. Add the 2 matrices and print it

b. Subtract 2 matrices

c. Multiply the individual elements of matrix

d. Divide the elements of the matrices

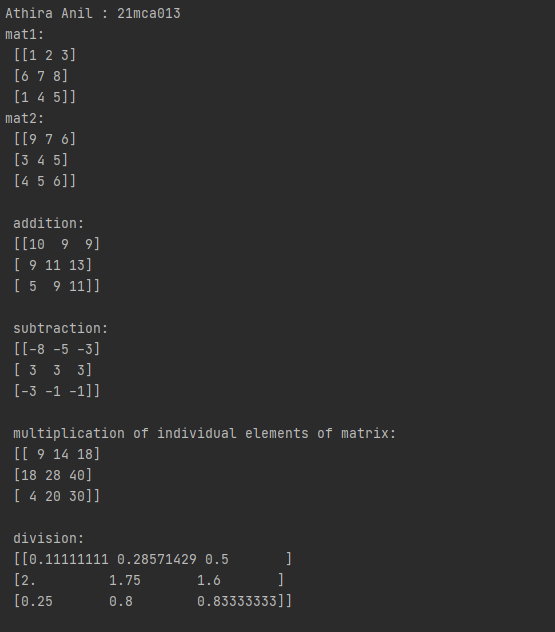
e. Perform matrix multiplication

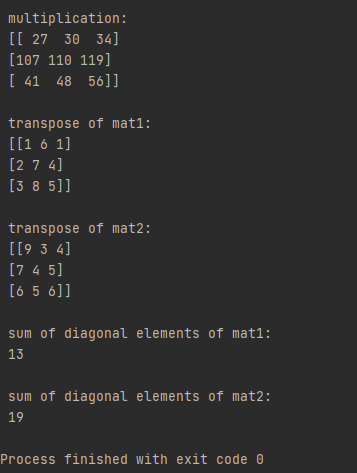
f. Display transpose of the matrix

g. Sum of diagonal elements of a matrix

print("Athira Anil : 21mca013")  
import numpy as np  
mat1=np.array([[1,2,3],[6,7,8],[1,4,5]])  
mat2=np.array([[9,7,6],[3,4,5],[4,5,6]])  
print("mat1:\n",mat1)  
print("mat2:\n",mat2)  
mat3=np.add(mat1,mat2)  
print("\n addition:\n",mat3)  
mat3=np.subtract(mat1,mat2)  
print("\n subtraction:\n",mat3)  
mat3=np.multiply(mat1,mat2)  
print("\n multiplication of individual elements of matrix:\n",mat3)  
mat3=np.divide(mat1,mat2)  
print("\n division:\n",mat3)  
mat4=np.matmul(mat1,mat2)  
print("\n multiplication:\n",mat4)  
print("\n transpose of mat1:\n",np.transpose(mat1));  
print("\n transpose of mat2:\n",np.transpose(mat2));  
print("\n sum of diagonal elements of mat1:\n",np.trace(mat1));  
print("\n sum of diagonal elements of mat2:\n",np.trace(mat2));

output

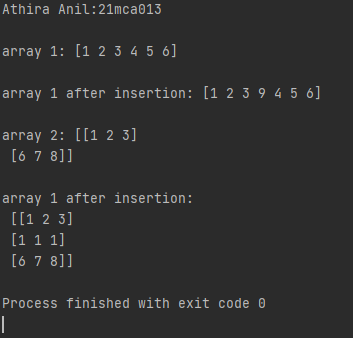




8. Demonstrate the use of insert() function in 1D and 2D array

print("Athira Anil:21mca013")  
import numpy as np  
arr1=np.array([1,2,3,4,5,6])  
print("\narray 1:", arr1)  
print("\narray 1 after insertion:",np.insert(arr1,3,9))  
arr2=np.array([[1,2,3],[6,7,8]])  
print("\narray 2:",arr2)  
print("\narray 1 after insertion:\n",np.insert(arr2, 1, np.array((1, 1, 1)), 0))

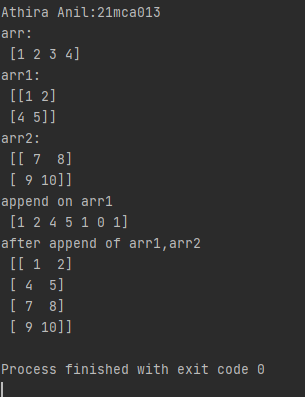
output



10. Demonstrate the use of append() function in 1D and 2D array.

print("Athira Anil:21mca013")  
import numpy as np  
arr=np.array([1,2,3,4])  
arr1=np.array([  
[1,2],  
[4,5]  
])  
arr2=np.array([  
[7,8],  
[9,10]  
])  
print('arr:\n',arr)  
print("arr1:\n",arr1)  
print("arr2:\n",arr2)  
arr3=np.append(arr1,arr2, axis=0)  
arr4=np.append(arr1,[1,0,1])  
print('append on arr1\n',arr4)  
print("after append of arr1,arr2\n",arr3)

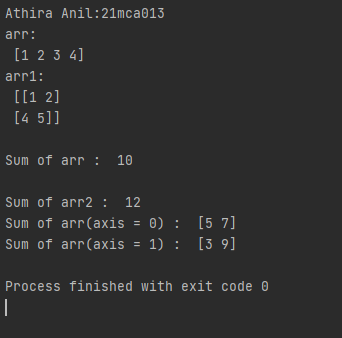
output



11.Demonstarte the use of sum() function in 1D and 2D array.

print("Athira Anil:21mca013")  
import numpy as np  
arr=np.array([1,2,3,4])  
arr1=np.array([  
[1,2],  
[4,5]  
])  
print("arr:\n",arr)  
print("arr1:\n",arr1)  
print("\nSum of arr : ", np.sum(arr))  
print("\nSum of arr2 : ", np.sum(arr1))  
print("Sum of arr(axis = 0) : ", np.sum(arr1, axis = 0))  
print("Sum of arr(axis = 1) : ", np.sum(arr1, axis = 1))

output



PART-2

1. Create a square matrix with random integer values(use randint()) and use

appropriate functions to find:

i) inverse

ii) rank of matrix

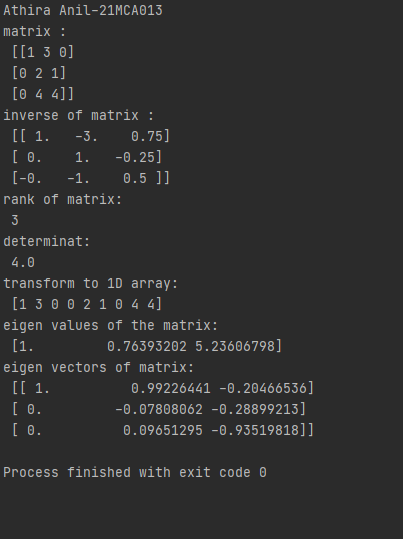
iii) Determinant

iv) transform matrix into 1D array

v) eigen values and vectors

print("Athira Anil-21MCA012")  
import numpy as np  
mat=np.random.randint(7,size=(3,3))  
print('matrix :\n',mat)  
print('inverse of matrix :\n',np.linalg.inv(mat))  
print('rank of matrix:\n',np.linalg.matrix\_rank(mat))  
print('determinat:\n',np.linalg.det(mat))  
print('transform to 1D array:\n',mat.flatten(order='c'))  
  
v,vr=np.linalg.eig(mat)  
print('eigen values of the matrix:\n',v)  
print('eigen vectors of matrix:\n',vr)

output



2. Create a matrix X with suitable rows and columns

i) Display the cube of each element of the matrix using different methods

(use multiply(), \*, power(),\*\*)

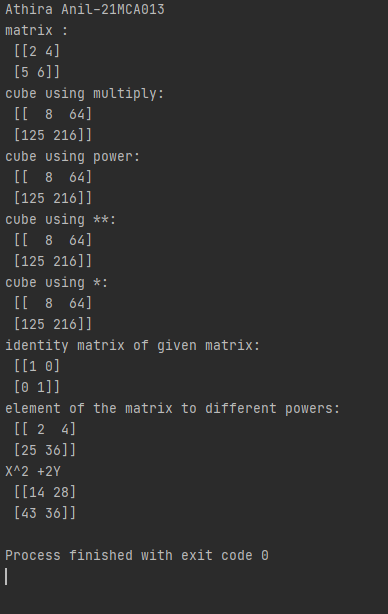
ii) Display identity matrix of the given square matrix.

iii) Display each element of the matrix to different powers.

iv) Create a matrix Y with same dimension as X and perform the operation X 2 +2Y

print("Athira Anil-21MCA013")  
import numpy as np  
  
mat=np.array([  
 [2,4],  
 [5,6]  
])  
print('matrix :\n',mat)  
print('cube using multiply:\n',np.multiply(mat,np.multiply(mat,mat)))  
print('cube using power:\n',np.power(mat,3))  
print('cube using \*\*:\n',mat\*\*3)  
print('cube using \*:\n',mat\*mat\*mat)  
print('identity matrix of given matrix:\n',np.identity(2,dtype=int))  
print('element of the matrix to different powers:\n',np.power(mat,[[1,1],[2,2]]))  
  
y=np.array([  
 [5,6],  
 [9,0]  
])  
  
print('X^2 +2Y\n',(mat\*\*2)+(2\*y))

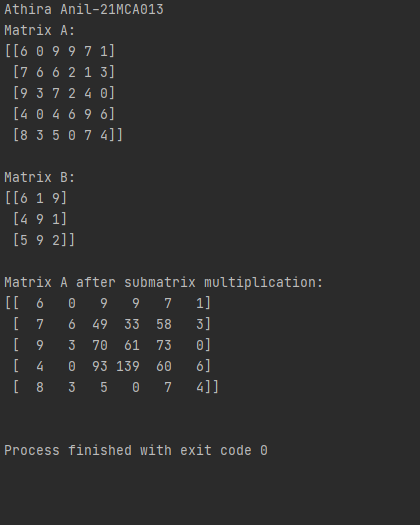
Output



3. Multiply a matrix with a submatrix of another matrix and replace the same in larger matrix.

print("Athira Anil-21MCA012")  
import numpy as np  
A = np.random.randint(0, 10, size=(5,6))  
B = np.random.randint(0, 10, size=(3,3))  
print("Matrix A:\n{}\n".format(A))  
print("Matrix B:\n{}\n".format(B))  
C = A[1:4,2:5] @ B  
A[1:4,2:5] = C  
print("Matrix A after submatrix multiplication:\n{}\n".format(A))

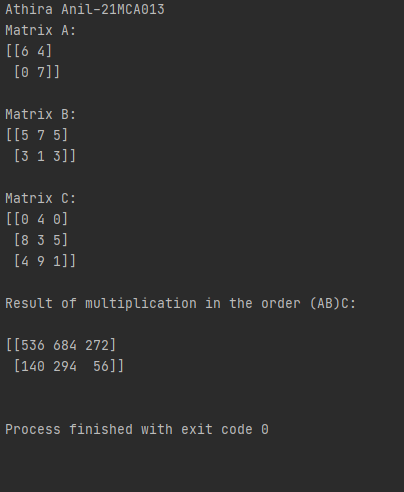
output



4. Given 3 Matrices A, B and C. Write a program to perform matrix multiplication of the 3 matrices.

print("Athira Anil-21MCA013")  
import numpy as np  
A = np.random.randint(0, 10, size=(2,2))  
B = np.random.randint(0, 10, size=(2,3))  
C = np.random.randint(0, 10, size=(3,3))  
print("Matrix A:\n{}\n".format(A))  
print("Matrix B:\n{}\n".format(B))  
print("Matrix C:\n{}\n".format(C))  
D = np.matmul(np.matmul(A,B), C)  
print("Result of multiplication in the order (AB)C:\n\n{}\n".format(D))

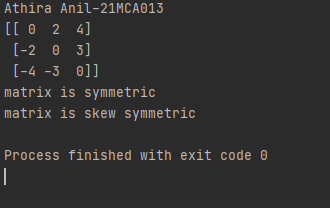
output



5. Write a program to check whether given matrix is symmetric or Skew Symmetric.

print("Athira Anil-21MCA013")  
import numpy as np  
  
  
A=np.matrix([[0,2,4],  
 [-2,0,3],  
 [-4,-3,0]])  
print(A)  
B=A.transpose()  
if A.all() == B.all():  
 print("matrix is symmetric ")  
else:  
 print("not symmetric")  
  
if np.allclose(-A,B)==True :  
 print("matrix is skew symmetric")  
else:  
 print("not skew symmetric")

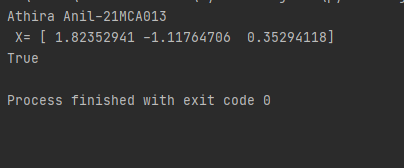
output



6. Write a program to find out the value of X using solve(), given A and b as above

print("Athira Anil-21MCA013")  
import numpy as np  
a = np.array([  
 [1, 2,4],  
 [3, 5,6],  
 [7, 8, 9]  
 ])  
b = np.array([1, 2, 7])  
  
x = np.linalg.solve(a, b)  
print(" X=",x)  
  
print(np.allclose(np.dot(a, x), b))

output



7. Write a program to perform the SVD of a given matrix. Also reconstruct the given matrix from the 3 matrices obtained after performing SVD.

print("Athira Anil-21MCA013")  
import numpy as np  
A = np.array([[1, 2], [3, 4], [5, 6]])  
print(A)  
# Singular-value decomposition  
U, s, VT = np.linalg.svd(A)  
# create m x n Sigma matrix  
print("\nU:",U)  
print("\ns:",s)  
print("\nVT",VT)  
Sigma = np.zeros((A.shape[0], A.shape[1]))  
# populate Sigma with n x n diagonal matrix  
Sigma[:A.shape[1], :A.shape[1]] = np.diag(s)  
# reconstruct matrix  
B = U.dot(Sigma.dot(VT))  
print("\n",B)

output

