Last task

1:  
import numpy as np

import matplotlib.pyplot as plt

import scipy.linalg as lag

import math

x = int(input("number: "))

print("Number's square: ", x\*\*2)

print("Number's cube: ",   x\*\*3)

theta = float(input("angle θ : "))

type = input("Write Radians or Degrees: ")

if type == 'Radians':

    print(math.cos(theta),math.sin(theta))

else:

    print(math.cos(math.radians(theta)),math.sin(math.radians(theta)))

meshpoints = np.linspace(-1,1,500)

print("53th element of meshPoints: ", meshpoints[52])

plt.plot(meshpoints,(np.sin(2\*math.pi\*meshpoints)))

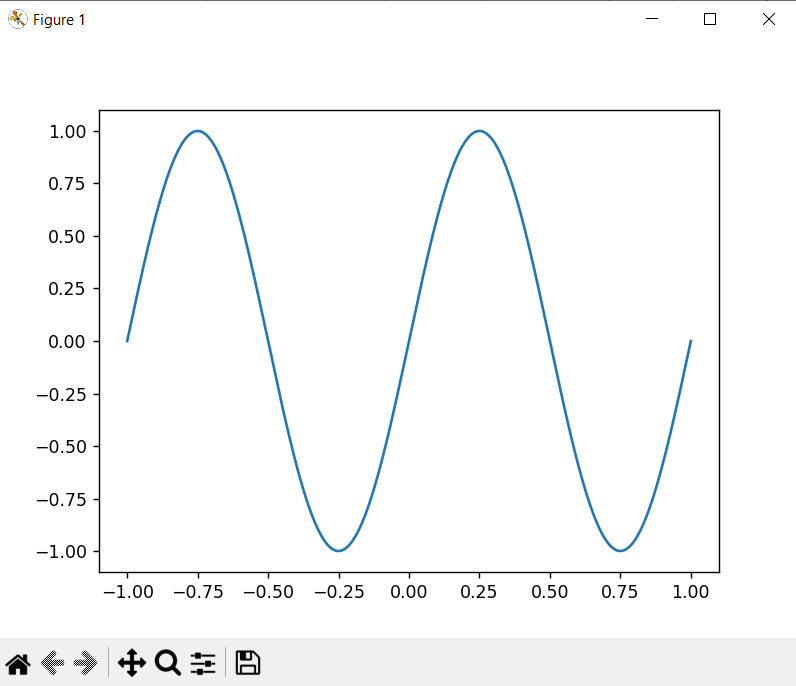
plt.savefig("1picture")

plt.show()

Output:

Изображение выглядит как текст

Автоматически созданное описание



2.py

import numpy as np

import matplotlib.pyplot as plt

import scipy.linalg as la

from math import \*

vec1 = np.array([ -1., 4., -9.])

mat1 = np.array([[ 1., 3., 5.], [7., -9., 2.], [4., 6., 8. ]])

vec2 = np.pi/4\*vec1

print(vec2)

vec2 = np.cos(vec2)

print(vec2)

vec3 = vec1 + 2\*vec2

print(vec3)

length = la.norm(vec3)

print(length)

vec4= mat1.dot(vec3)

print(vec4)

mat2 = mat1.T

print(mat2)

det = la.det(mat1)

print("det:", det)

trace = sum(np.diag(mat1))

print('trace:', trace)

min\_element = np.min(vec1)

print("min\_in\_vector", min\_element)

coeff, = np.where(np.isclose(vec1, min\_element))

print("j in vector[j]=min", coeff)

min\_element\_in\_matrix = np.min(mat1)

print('min\_element\_in\_matrix', min\_element\_in\_matrix)

i,j = np.where(np.isclose(mat1, min\_element\_in\_matrix))

print("i and j of min", i, j)

A=np.array([[17, 24, 1, 8, 15],

[23, 5, 7, 14, 16],

[ 4, 6, 13, 20, 22],

[10, 12, 19, 21, 3],

[11, 18, 25, 2, 9]])

B = A.T

sum\_row = np.array([sum(A[0]),sum(A[1]),sum(A[2]),sum(A[3]),sum(A[4])])

sum\_column = np.array([sum(B[0]),sum(B[1]),sum(B[2]),sum(B[3]),sum(B[4])])

sum\_d1=sum(np.diag(A))

sum\_d2 =sum(np.diag(np.fliplr(A)))

arr = np.array([sum\_d1,sum\_d2,np.min(sum\_row),np.max(sum\_row),np.min(sum\_column),np.max(sum\_column)])

for i in arr:

    print(i)

if np.min(arr)==np.max(arr):

    print("yes")

else:

    print("No")

M = np.array([np.random.rand(10) for i in range(10)])

#print(M)

MUL = M[:5, :5]

MUR = M[5:10, :5]

MLL = M[:5, 5:10]

MLR = M[5:10, 5:10]

print(MUL)

print(MUR)

print(MLL)

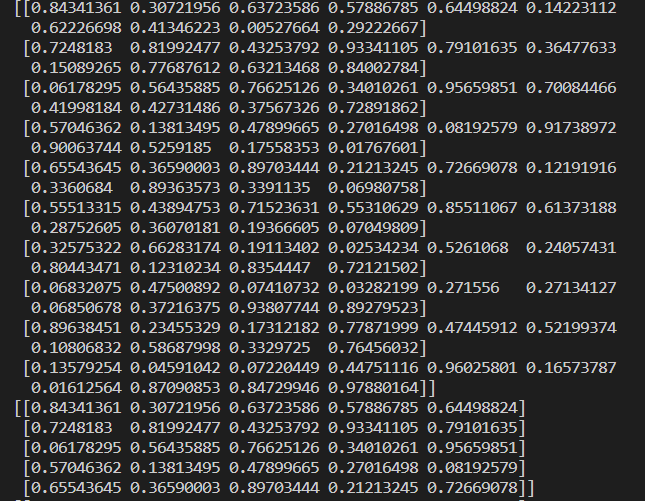
print(MLR)

OUTPUT:

Изображение выглядит как текст

Автоматически созданное описание

10 by 10 matrix:



5by5 parts:

Изображение выглядит как текст, табличка

Автоматически созданное описание

3py:

import numpy as np

import matplotlib.pyplot as plt

f = lambda x: np.e\*\*(-x/10)\*np.sin(np.pi\*x)

g = lambda x: x\*np.e\*\*(-x/3)

x = np.arange(0, 10.5, 0.5)

x0 = np.linspace(0,10,1000)

y\_f = f(x0)

y\_g = g(x0)

plt.plot(x0, y\_f, label = "f(x)", color = "blue")

plt.plot(x0, y\_g, label = "g(x)", color = "red")

plt.xlabel("x")

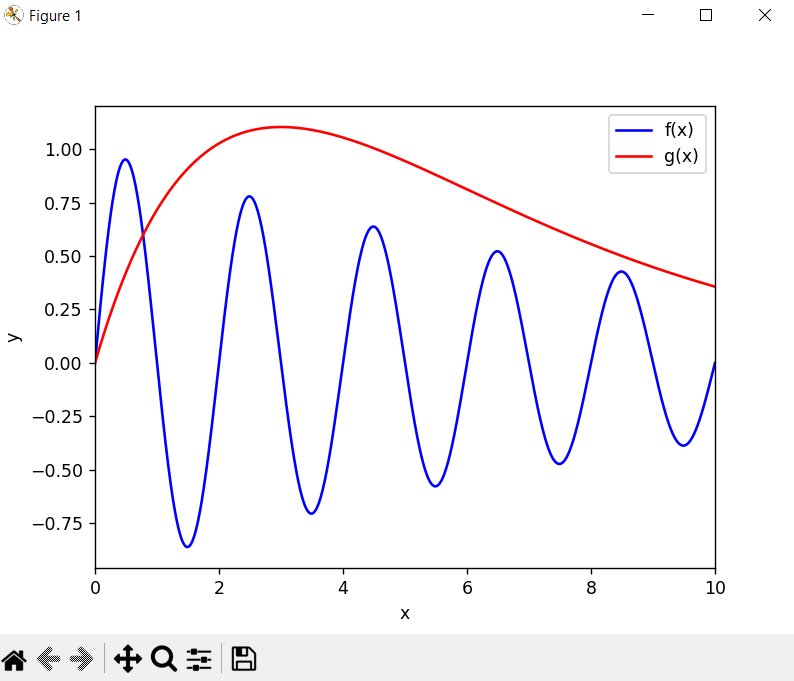
plt.ylabel("y")

plt.xlim(0, 10)

plt.legend()

plt.savefig("3py.jpg")

plt.show()



3b py:

import numpy as np

import matplotlib.pyplot as plt

fig = plt.figure()

ax1 = fig.add\_subplot(311)

ax2 = fig.add\_subplot(312)

ax3 = fig.add\_subplot(313)

def r(r0, angle):

    return r0 + np.cos(angle)

def x(r, angle):

    return r\*np.cos(angle)

def y(r, angle):

    return r\*np.sin(angle)

angle\_arr = np.arange(0, 2\*np.pi, 0.01)

r1 = 0.8

r2 = 1.0

r3 = 1.2

x1, x2, x3, y1,y2, y3  = [], [], [], [], [], []

for i in range(len(angle\_arr)):

    x1.append(x(r(r1, angle\_arr[i]), angle\_arr[i]))

    x2.append(x(r(r2, angle\_arr[i]), angle\_arr[i]))

    x3.append(x(r(r3, angle\_arr[i]), angle\_arr[i]))

    y1.append(y(r(r1, angle\_arr[i]), angle\_arr[i]))

    y2.append(y(r(r2, angle\_arr[i]), angle\_arr[i]))

    y3.append(y(r(r3, angle\_arr[i]), angle\_arr[i]))

ax1.plot(x1, y1)

ax2.plot(x2, y2)

ax3.plot(x3, y3)

plt.savefig("3b.pdf")

plt.show()

