Отчёт по четвёртой лабораторной работе   
по Python

(Python LR4)

Выполнил: Соболь Евгений

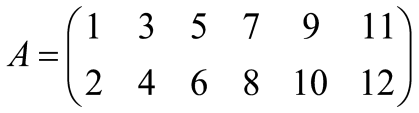
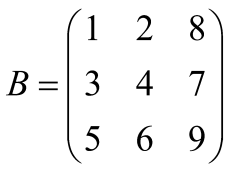
Группа: ФПММ-ИСТ-19-1бзу

**Вариант: 10**

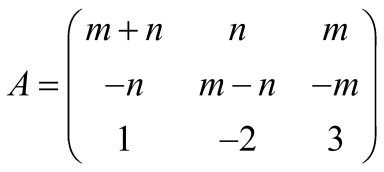
**Задание 1**

Используя модуль NumPy:

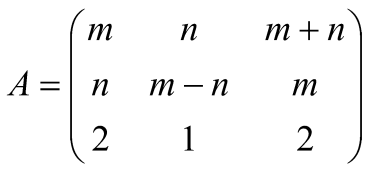
а) Вывести матрицы, транспонированные данным:

; 

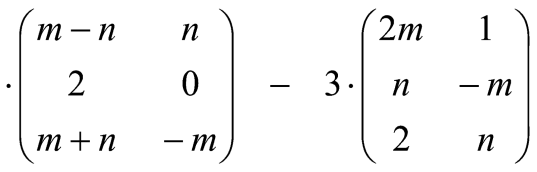
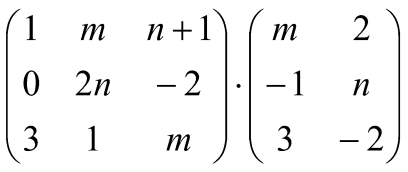
б) Вычислить определитель матрицы (*n, m* – вводятся пользователем):



в) Вывести на экран обратную матрицу к матрицеhttps://lh5.googleusercontent.com/PIbTUi7qRzLXbblsel5MP43L5c9_jnAMQO8He7vNOH0PcMab1RrNcgbJWom9IPChHDEWzUygYV2atNbM43PxLKWJcKA9mumNtfxMfh68vMF9zCRmw3EsEPuMjdtyNw4jbPQSa8I8YnsOqy8BZA и результат проверки выполнение равенства https://lh3.googleusercontent.com/t34aUzHDmG-QLEqmOvttvUtDujE3jDhFBaqItUJLDzg2oTIErmHnonYr9jIzzJiLMSoVnBi1UvgpSAf4Fe1lC_u---5uuEsocHYThbcyC0aKOd-fPcq2w4MyMKDU6EHm0kzRhENFw1ww5INcMQ:



г) Выполнить действия над матрицами:

2; 

**Задание 2**

Написать программу, которая формирует матрицуhttps://lh6.googleusercontent.com/8UQKXeDThtAGUb3IQGAR9pW6qHPURvFYJMLDdYL4kxh6nHsMn_ERnAam4QMVJHCQNAlVcz6osr8SmS8j7Ye79JgwtUpdEMbnakoyAYOtceK-b8627J5P4MZTws0rzhiKok8N1X8cKrlq4cg1ZQ и https://lh6.googleusercontent.com/7TnwzHxA5MJF4P5lfNuDi1PjjP5sVzc0O3TYzuYNbLU5ReFba-oZl4thKHoZ4-hpxbWPNpQtrjuSp134jbl2aWjTl_5aNIkFxmVwrq4TgN_a7GfSMBm8Tv7FUKMeip8LX3GRqQJpZ4NpxOzjTw в следующем виде (*n* – вводится пользователем)

|  |  |
| --- | --- |
| https://lh3.googleusercontent.com/swjftDk6Y-HtNk3CI1qHqEtXmIVxkYXx6-sZNKvt5NtVlO6aS8KBACrdVd5YEJ2lhZs1ih5NF0YvWQS_4Cpdxzmb5YoyymhnDyf9jUoKWGQ3sZi_EAQK-G-_qgFFn9v8E7yhKGWdTTuPyCf9uw | https://lh6.googleusercontent.com/KIuoQtCAnpe-zbUqHqE3qtBXH6ZNwviloaPIFBZCNB7d5l9tSzPFVXA1rMeepVqzpyAfuZqVZ8rAB3T-us_L5qIz-PzA9bMUKeH-jeIMcOtc5Ggjs6GwNz9-1W7rlQBiMWmJYFTW0A6E3qns7w |

**Решение:**

Решение лабораторной оформил в виде класса, в котором методы решают задания.

import numpy as np

from numpy import linalg as la

# Создаём класс

class Lab4:

# Метод для решения задачи 1А

@staticmethod

def task\_1a():

return np.array([[1, 3, 5, 7, 9, 11], [2, 4, 6, 8, 10, 12]]).transpose(), \

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

# Метод для решения задачи 1В

@staticmethod

def task\_1b(n, m):

# Вычисляем и выводим определитель созданной матрицы

return la.det(np.array([[m + n, n, m], [-n, m - n, -m], [1, -2, 3]]))

# Метод для решения задачи 1С

@staticmethod

def task\_1c(n, m):

# Создаём матрицу

a = np.array([[m, n, m + n], [n, m - n, m], [2, 1, 2]])

# Создаём обратную матрицу

b = la.inv(a)

print(b)

# Сравнение произведения матрицы A и B с единичной матрицей

if np.array\_equal(np.dot(a, b), np.array([[1, 0, 0], [0, 1, 0], [0, 0, 1]])):

return True

return False

# Метод для решения задачи 1G

@staticmethod

def task\_1g(n, m):

# Создаём матрицы и вычисляем 1 вырожение и вычисляем 2 выражение

return np.dot(np.array([[m - n, n], [2, 0], [m + n, -m]]), 2) - np.dot(np.array([[2 \* m, 1], [n, -m], [2, n]]),

3), \

np.dot(np.array([[1, m, n + 1], [0, 2 \* n, -2], [3, 1, m]]), np.array([[m, 2], [-1, n], [3, -2]]))

# Метод для решения задачи 2\_1

@staticmethod

def task\_2\_1(n):

# Создаём единичную матрицу размером N x N

m = np.eye(n)

# Строка

for i in range(n):

# Столбец

for j in range(i, n):

m[i][j] = j + 1

return m

# Метод формирующий матрицу n x 2 для задания 2\_2

@staticmethod

def \_\_pol\_spiral\_n(n):

# Создаём единичную матрицу n на n

m = np.eye(n)

# Массив счётчиков

z = np.array([1, 0, 0, 0, 1, 0, -1, -2, 2, 0])

# Цикл заполнения матрицы

while 1:

# Условия остановки цикла

if z[0] > ((1 / 2) \* n + (1 / 2)) \* n:

break

# Заполнение в право по горизонтале

for i in range(z[1], n + z[2]):

m[z[3]][i] = z[0]

z[0] += 1

z[1] += 2

z[3] += 1

# Условия остановки цикла

if z[0] > ((1 / 2) \* n + (1 / 2)) \* n:

break

# Заполнение вниз по вертикали

for i in range(z[4], n + z[5]):

m[i][n + z[6]] = z[0]

z[0] += 1

z[4] += 1

z[5] -= 2

z[6] -= 1

z[9] = n + z[7]

# Условия остановки цикла

if z[0] > ((1 / 2) \* n + (1 / 2)) \* n:

break

# Заполнение по диагонали

for i in range(z[8], n + z[2]):

m[z[9] + z[2]][z[9]] = z[0]

z[0] += 1

z[9] -= 1

z[7] -= 1

z[8] += 2

z[2] -= 1

return m

# Метод формирующий матрицу 2n x 2n для задания 2\_2

@staticmethod

def \_\_pol\_spiral\_2n(matr, n):

# Создаём матрицу n x 2n

matr2 = np.zeros((n, 2 \* n))

# Заполняем матрицу n x 2n

for i in range(len(matr2)):

for j in range(n):

matr2[i][j] = matr[i][j]

for j in range(n):

matr2[i][j + n] = matr[i][j]

# Возвращаем стакнутую матрицу

return np.vstack((matr2, matr2))

# Метод для решения задачи 2\_2

@staticmethod

def task\_2\_2(n):

return Lab4.\_\_pol\_spiral\_n(n), Lab4.\_\_pol\_spiral\_2n(Lab4.\_\_pol\_spiral\_n(n), n)

# Создаём экземпляр класса

L4 = Lab4()

print("Лабораторная 4, Задание 1 А. ")

print("------------------------------")

a, b = L4.task\_1a()

print(a)

print(b, "\n")

print("Лабораторная 4, Задание 1 B. ")

print("------------------------------")

# Вводим n

n1 = int(input("n: "))

# Вводим m

m1 = int(input("m: "))

print(f"Определитель равен:{L4.task\_1b(n1, m1)} \n")

print("Лабораторная 4, Задание 1 C. ")

print("------------------------------")

# Вводим n

n1 = int(input("n: "))

# Вводим m

m1 = int(input("m: "))

if L4.task\_1c(n1, m1):

print("A\*A^-1 = E \n")

else:

print("A\*A^-1 != E \n")

print("Лабораторная 4, Задание 1 G. ")

print("------------------------------")

# Вводим n

n1 = int(input("n: "))

# Вводим m

m1 = int(input("m: "))

a, b = L4.task\_1g(n1, m1)

# Вывод результата 1 выражения

print(a)

# Вывод результата 2 выражения

print(b, "\n")

print("Лабораторная 4, Задание 2.1 ")

print("------------------------------")

# Вводим n

n1 = int(input("n: "))

# Выводим матрицу

print(L4.task\_2\_1(n1), "\n")

print("Лабораторная 4, Задание 2.2 ")

print("------------------------------")

# Вводим n

n1 = int(input("n: "))

a, b = L4.task\_2\_2(n1)

# Вывод матрицы NxN

print(a)

# Вывод матрицы 2Nx2N

print(b, "\n")

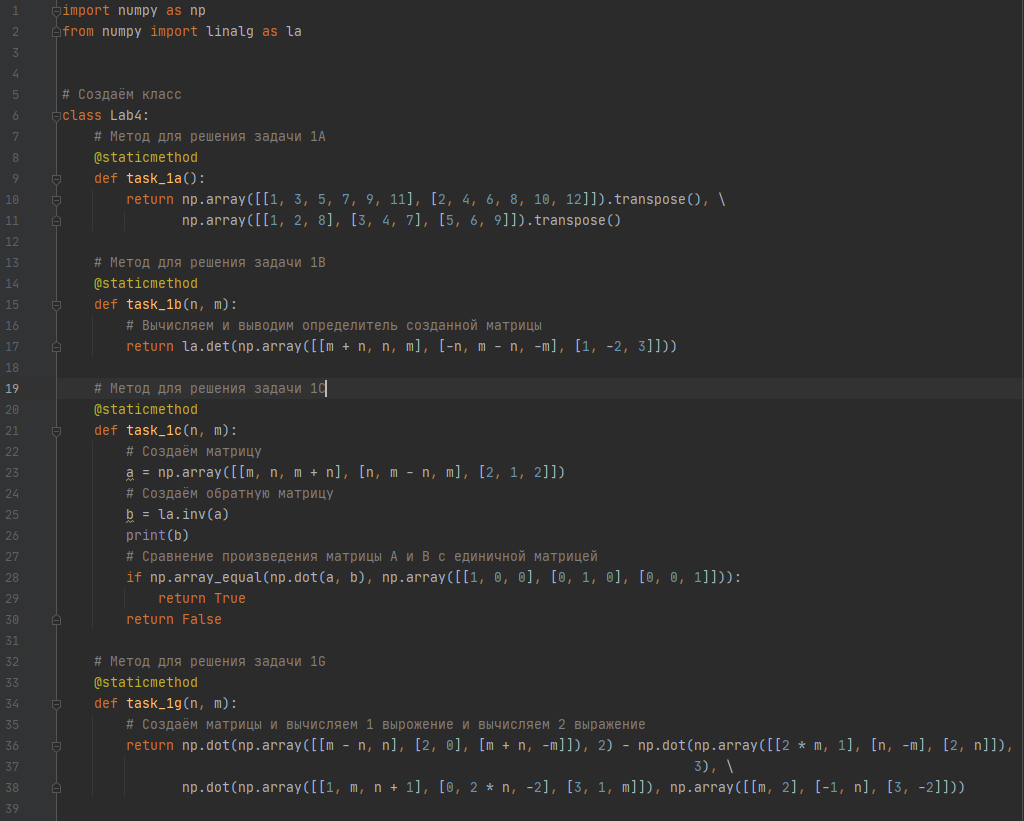


Рисунок -Программный код (1)

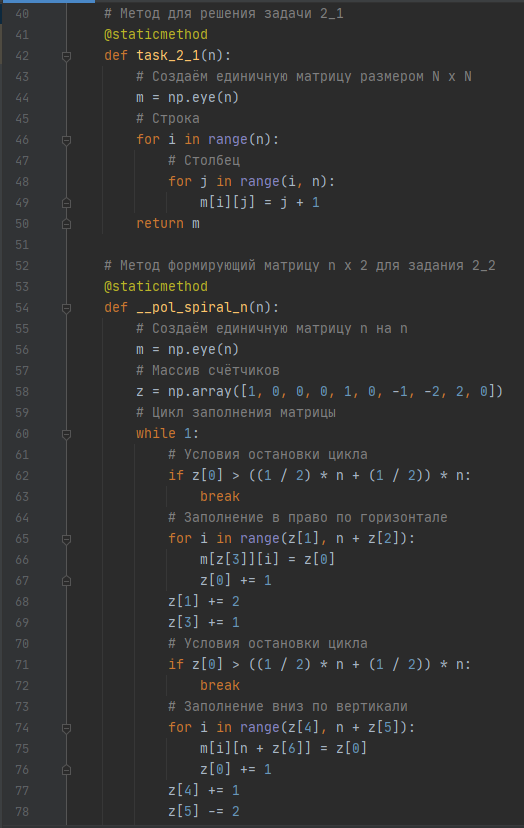


Рисунок - Программный код (2)

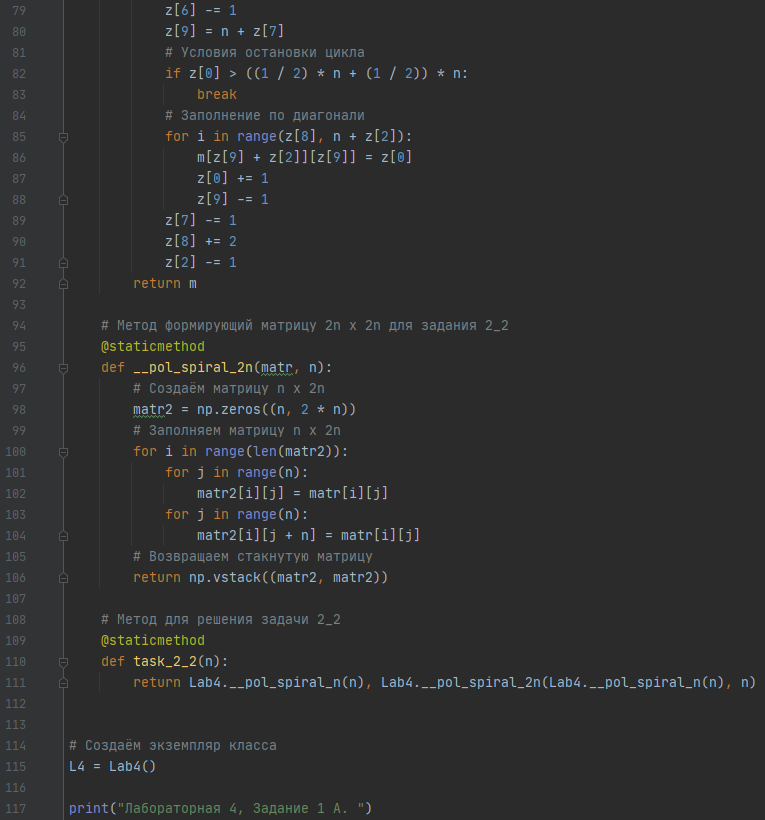


Рисунок - Программный код (3)

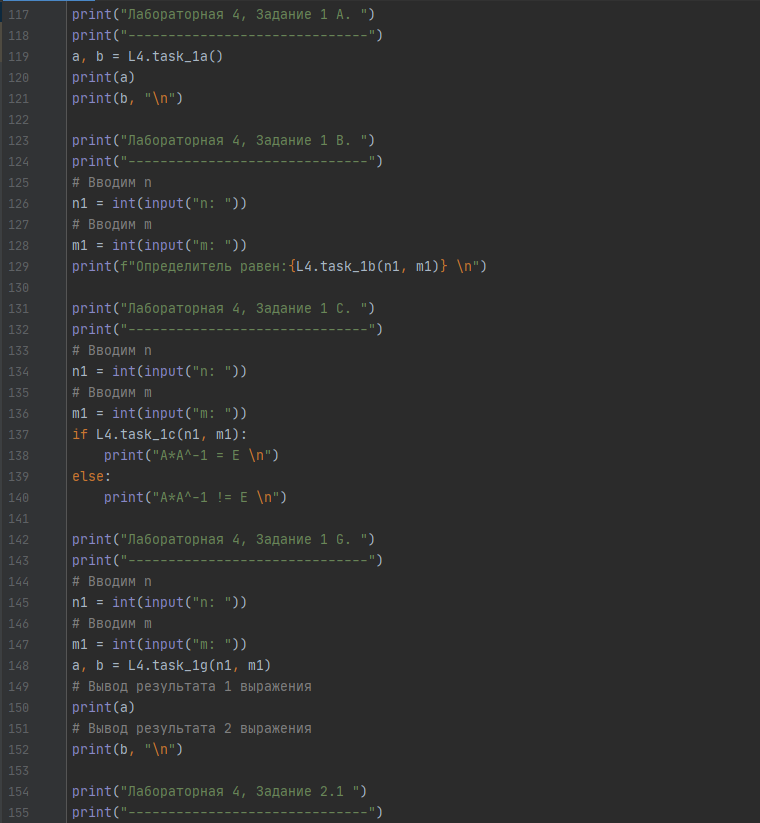


Рисунок - Программный код (4)

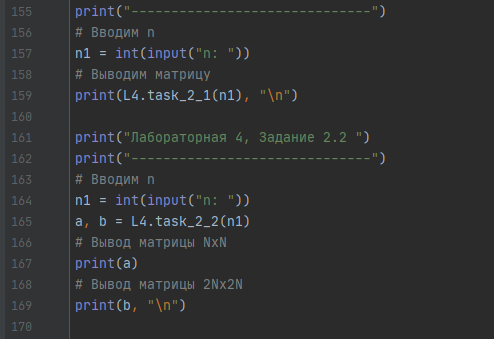


Рисунок - Программный код (5)

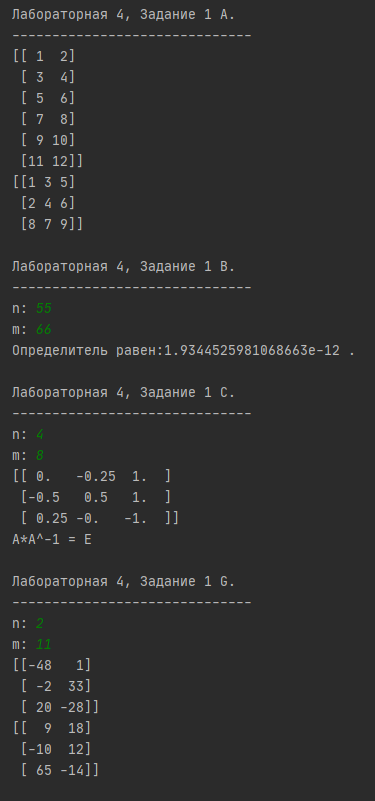


Рисунок - Ввод – вывод (1)

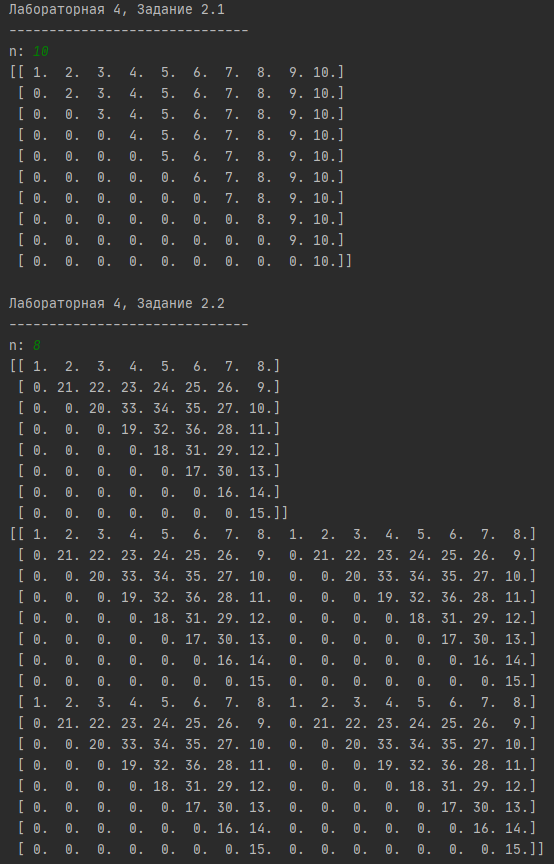


Рисунок - Ввод – вывод (2)

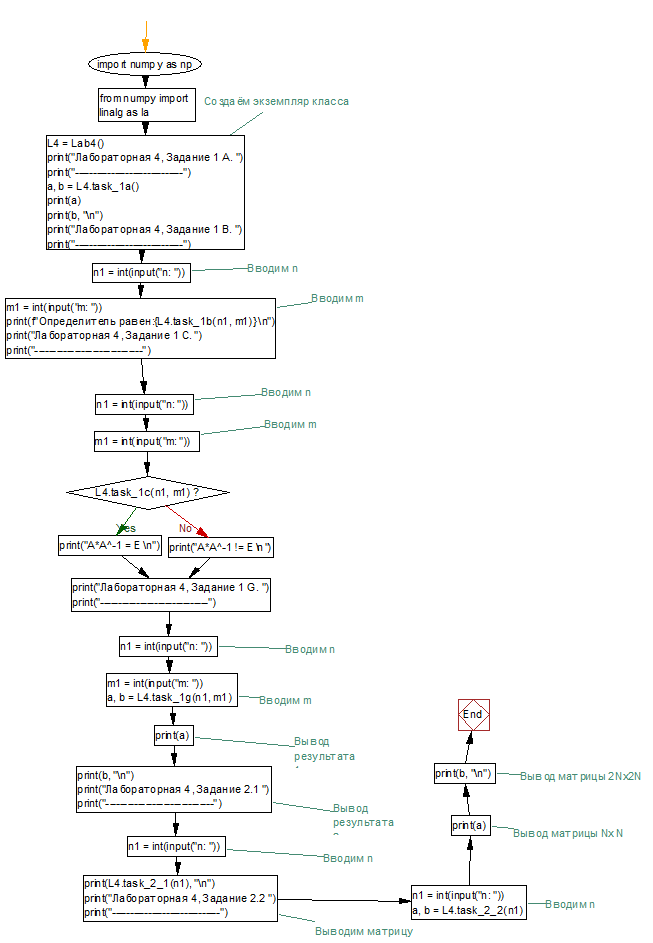


Рисунок -Блок-схема (1)

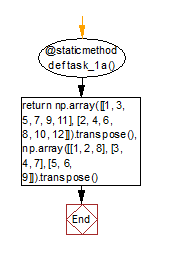


Рисунок -Блок-схема метод задания 1A

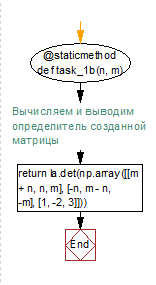


Рисунок - Блок-схема метод задания 1B

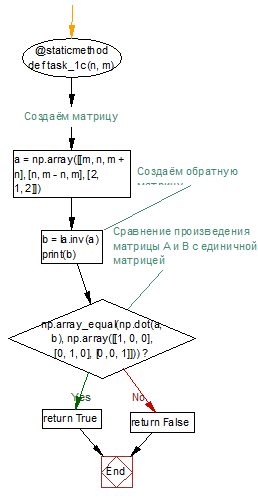


Рисунок - Блок-схема метод задания 1C

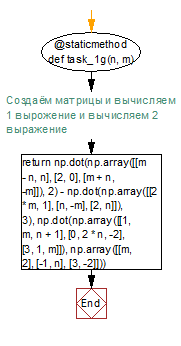


Рисунок - Блок-схема метод задания 1G

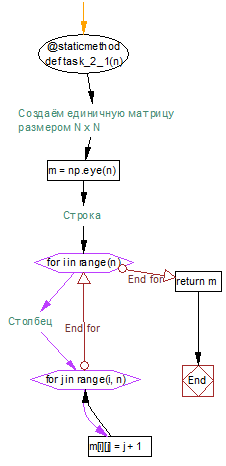


Рисунок - Блок-схема метод задания 2.1

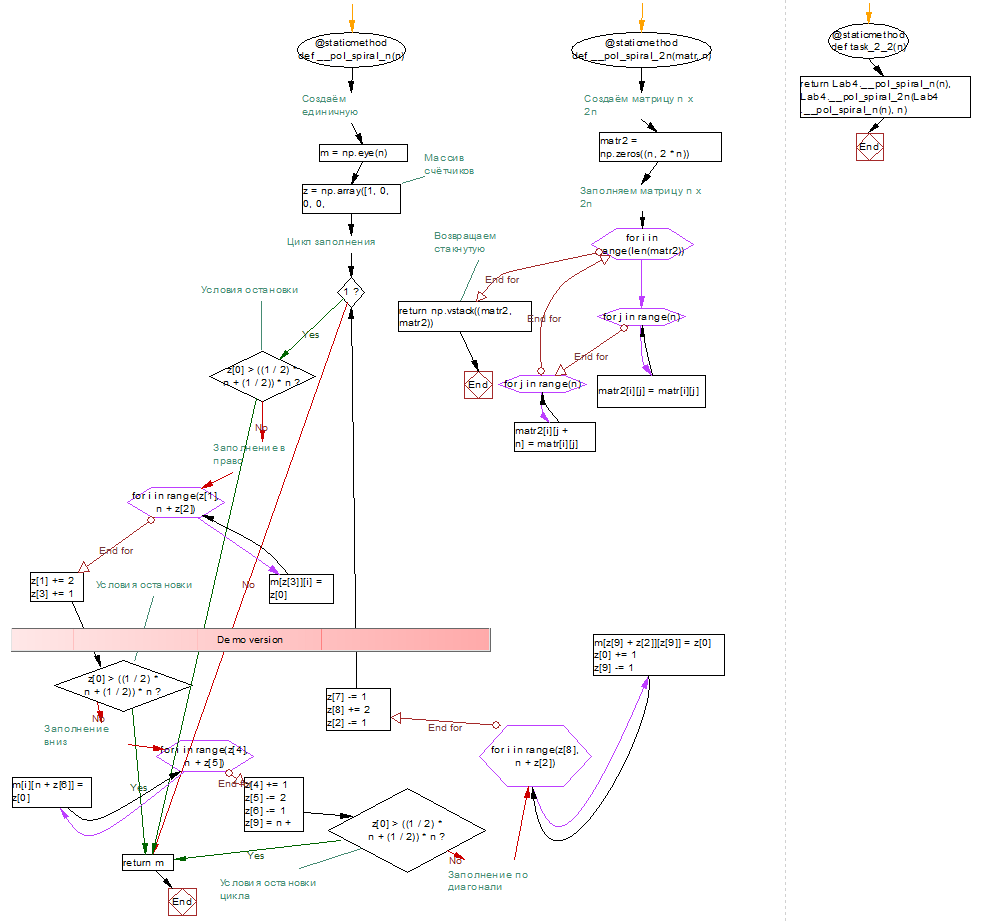


Рисунок - Блок-схема методов задания 2.2