Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчёт по лабораторной работе № 3**

Дисциплина: Вычислительная математика

Выполнил студент гр. 3530901/10003 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Я.А. Иванов

(подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Н. Цыган

(подпись)

“27” марта 2023 г.

Санкт-Петербург

2023

Оглавление

[Задание: 2](#__RefHeading___Toc1279_301334972)

[Инструменты: 2](#__RefHeading___Toc1281_301334972)

[Ход выполнения работы: 2](#__RefHeading___Toc1283_301334972)

[Приведение к системе первого порядка: 2](#__RefHeading___Toc2999_2740806723)

[RKF45 3](#__RefHeading___Toc3001_2740806723)

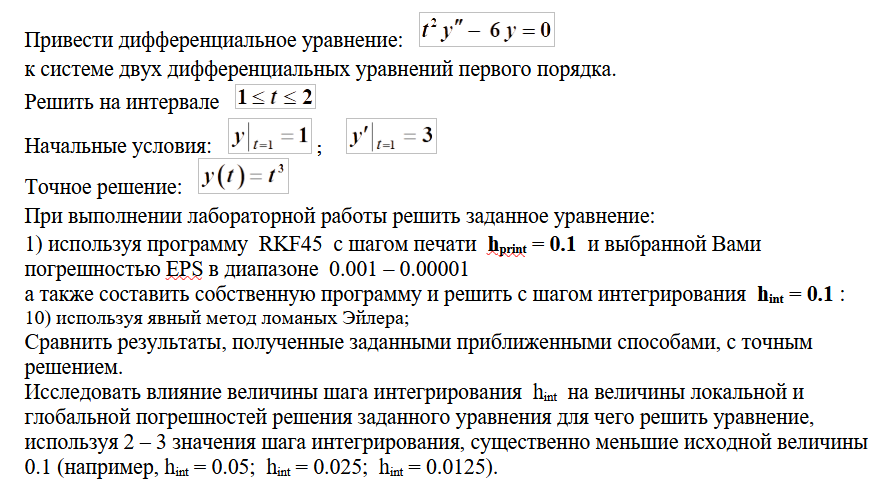
[Метод Эйлера 3](#__RefHeading___Toc3003_2740806723)

[Результаты работы программы: 5](#__RefHeading___Toc3005_2740806723)

[Вывод: 7](#__RefHeading___Toc1291_301334972)

[Ссылки: 7](#__RefHeading___Toc1293_301334972)

# Задание:

**Пункт 1 и 10. EPS=0.00001**

# Инструменты:

Для работы был выбран язык программирования Python версии 3.09 ввиду наличия необходимых библиотек для выполнения поставленной задачи, а именно:

* NumPy – для большей скорости расчетов
* pandas – для красивого вывода в консоль таблицы
* MatplotLib – для вывода графиков
* SciPy – для функций расчета интерполяции и интеграла

# Ход выполнения работы:

## **Приведение к системе первого порядка:**

Для преобразования дифференциального уравнения в систему двух уравнений первого порядка мы вводим новую переменную y1=y и y2=y', а затем находим их производные по t:

y1' = y' = y2

y2' = y'' = t^2 - 6y1

Таким образом, мы получаем систему:

y1' = y2

y2' = t^2 — 6y1

Зададим начальные условия и получившуюся систему диф. Уравнений:

# Функция правой части системы дифференциальных уравнений  
def f(t, y):  
 return [y[1], 6 \* y[0] / t \*\* 2]  
  
  
# Задаем начальные условия  
y0 = [1, 3]  
t0 = 1  
tmax = 2  
eps = 0.00001

## **RKF45**

Функция solve\_rkf45 использует метод RKF45 для решения системы дифференциальных уравнений. Она использует функцию правой части f, задает начальные условия и использует метод set\_integrator для настройки метода RKF45. Она возвращает массив пар из значения t и соответствующего результата работы RKF45.

# Решение системы дифференциальных уравнений методом RKF45  
def solve\_rkf45(h):  
 t = np.arange(t0, tmax + h, h)  
 rk\_integ = ode(f).set\_integrator("dopri5", atol=eps).set\_initial\_value(y0, t[0])  
 X = np.array([y0, \*[rk\_integ.integrate(t[i]) for i in range(1, len(t))]])  
 return t, X[:, 0]

## **Метод Эйлера**

Функция solve\_euler принимает на вход шаг и возвращает данные аналогично с RKF45

# Решение системы дифференциальных уравнений методом Эйлера  
def solve\_euler(h):  
 t = np.arange(t0, tmax + h, h)  
 y = np.zeros\_like(t)  
 y[0] = y0[0]  
 z = np.zeros\_like(t)  
 z[0] = y0[1]  
 for i in range(1, len(t)):  
 y[i] = y[i - 1] + h \* z[i - 1]  
 z[i] = z[i - 1] + h \* 6 \* y[i - 1] / t[i - 1] \*\* 2  
 return t, y

**Обработка результатов работы функции и вычисление погрешностей**

Получаем данные и вычисляем погрешности

x\_values = np.arange(t0, tmax + h\_int, h\_int)  
  
# вычислим значения функций  
rkf45\_values = solve\_rkf45(h\_int)  
euler\_values = solve\_euler(h\_int)  
exact\_values = [t \*\* 3 for t in x\_values]  
  
# вычислим погрешности  
rkf45\_3\_errors = np.abs(rkf45\_values[1] - rkf45\_values[0] \*\* 3)  
euler\_errors = np.abs(euler\_values[1] - euler\_values[0] \*\* 3)

print("RKF45 global err: ",np.sum(rkf45\_3\_errors))  
print("Euler global err: ", np.sum(euler\_errors))

Отобразим таблицу и графики

# вывод таблицы  
 results = pd.DataFrame({  
 'h': euler\_values[0],  
 'Value RKF45': rkf45\_values[1],  
 'Value Euler': euler\_values[1],  
 'Exact': exact\_values,  
 'Error RKF45': rkf45\_3\_errors,  
 'Error Euler': euler\_errors,  
 })  
  
 print(results.to\_string(index=False))  
 # выводим график значений  
 plt.figure(figsize=(15, 4))  
 y = rkf45\_values[1]  
 t = euler\_values[0]  
 print\_graph(t, y, 'RKF45, step=' + str(h\_int), 1, 3)  
  
 t, y = euler\_values  
 print\_graph(t, y, 'Эйлер', 2, 3)  
  
 print\_graph(x\_values, exact\_values, 'Exact solution', 3, 3)  
  
 plt.savefig("Graphs\_h\_" + str(h\_int) + ".jpg")  
 plt.show()  
 # выводим график погрешности  
 plt.figure(figsize=(15, 4))  
  
 y = rkf45\_3\_errors  
 t = rkf45\_values[0]  
 print\_graph(t, y, 'RKF45, step=' + str(h\_int), 1, 2)  
  
 y = euler\_errors  
 t = euler\_values[0]  
 print\_graph(t, y, 'Эйлер', 2, 2)  
  
 plt.savefig("error\_h\_" + str(h\_int) + ".jpg")  
 plt.show()

Выполним перебор для значений h\_list

h\_list = [0.1, 0.05, 0.025, 0.0125]

for h\_int in h\_list:

#далее идет код из пунктов выше

## **Результаты работы программы:**

Для h\_int = 0.1

Таблица:

h\_int = 0.1

h Value RKF45 Value Euler Exact Error RKF45 Error Euler

1.0 1.000 1.000000 1.000 0.000000e+00 0.000000

1.1 1.331 1.300000 1.331 6.254157e-09 0.031000

1.2 1.728 1.660000 1.728 2.818067e-08 0.068000

1.3 2.197 2.084463 2.197 3.931797e-08 0.112537

1.4 2.744 2.578092 2.744 6.818008e-08 0.165908

1.5 3.375 3.145726 3.375 9.921412e-08 0.229274

1.6 4.096 3.792282 4.096 1.329554e-07 0.303718

1.7 4.913 4.522723 4.913 1.698581e-07 0.390277

1.8 5.832 5.342046 5.832 2.103206e-07 0.489954

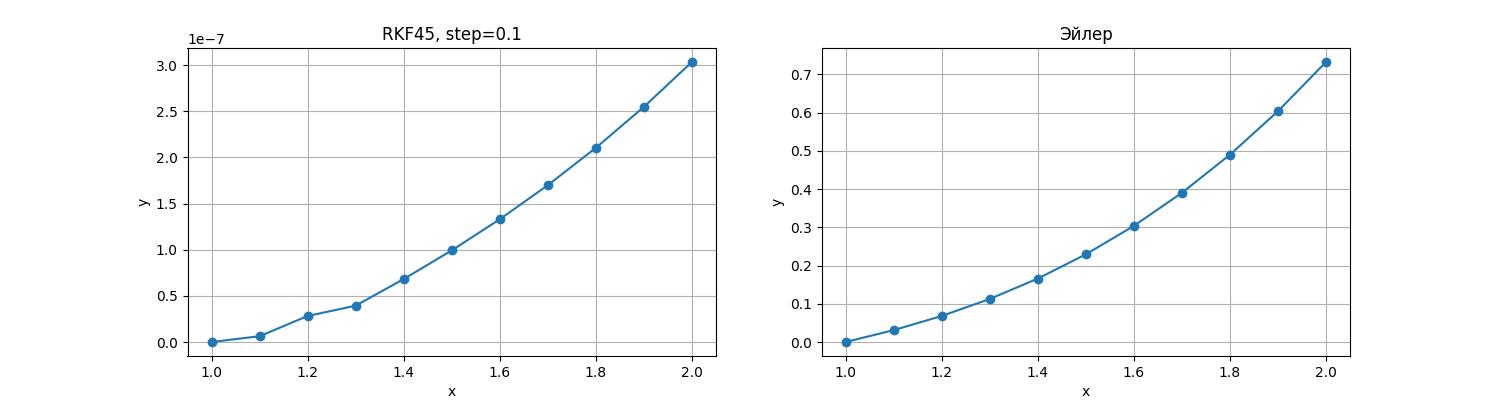
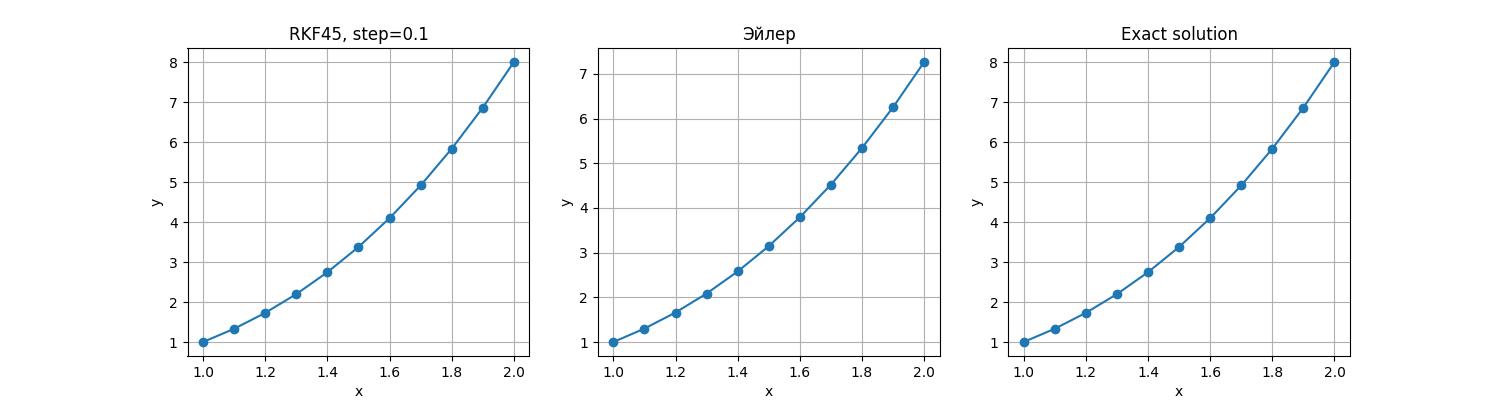
1.9 6.859 6.255266 6.859 2.547031e-07 0.603734

2.0 8.000 7.267413 8.000 3.033378e-07 0.732587

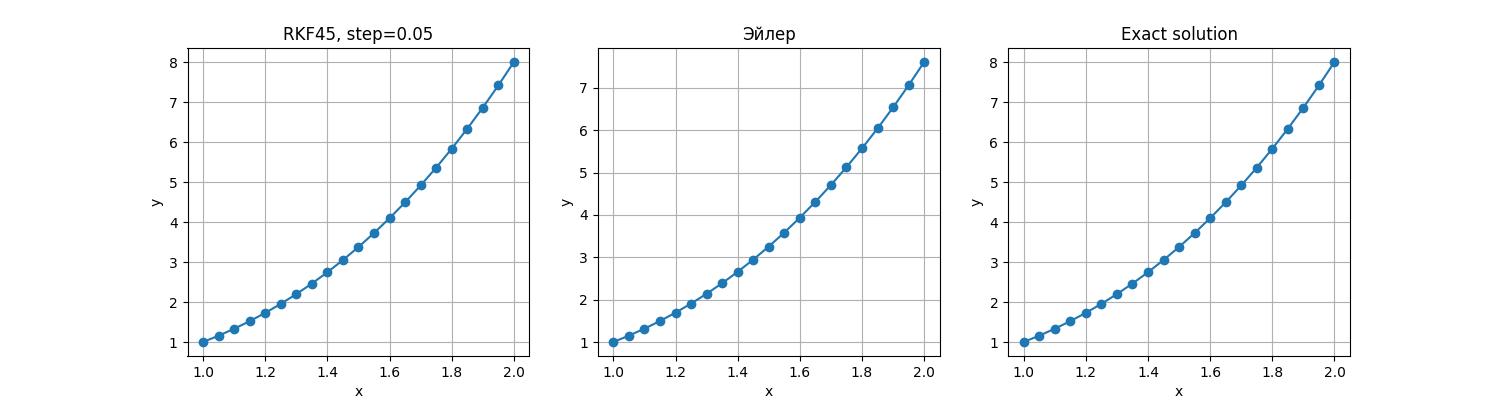
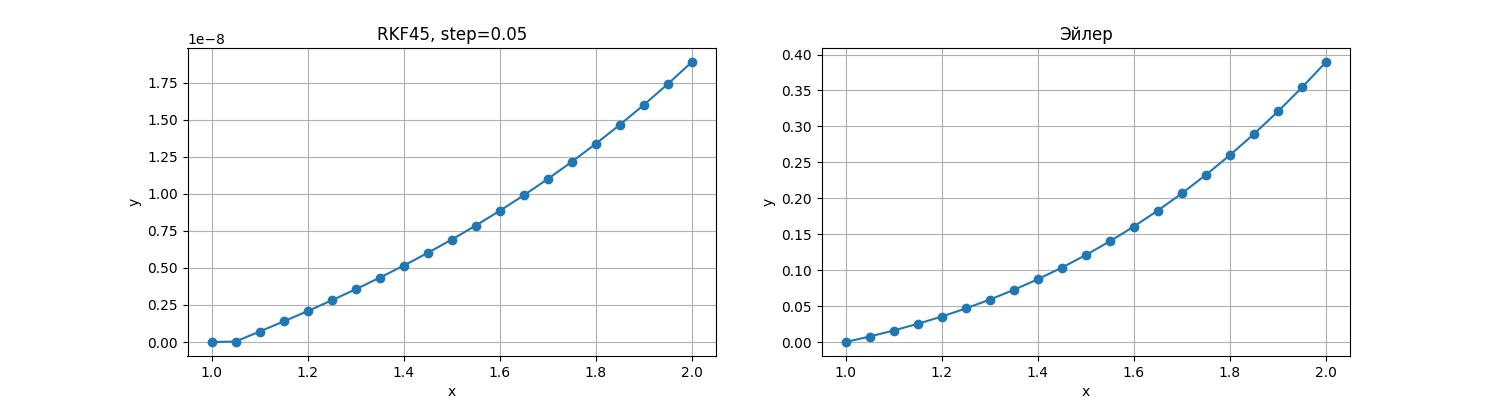
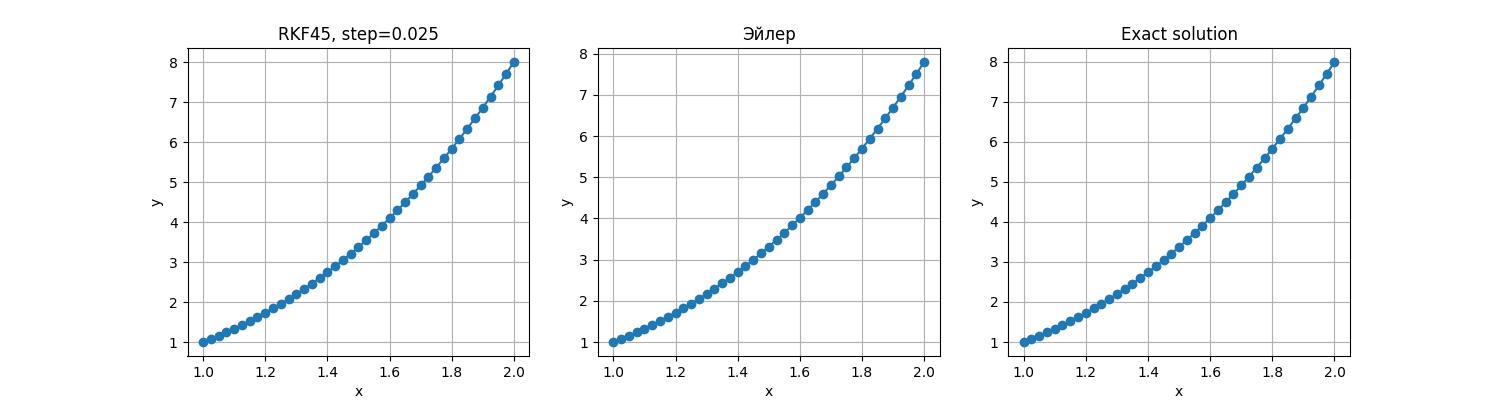
RKF45 global err: 1.3123220110600187e-06

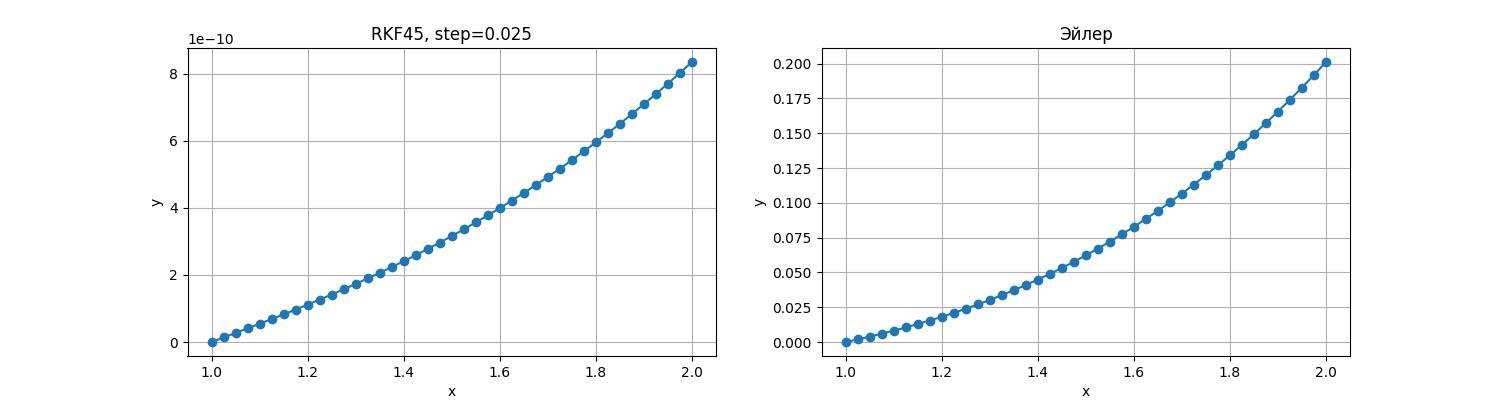
Euler global err: 3.1269889304508016

Графики:

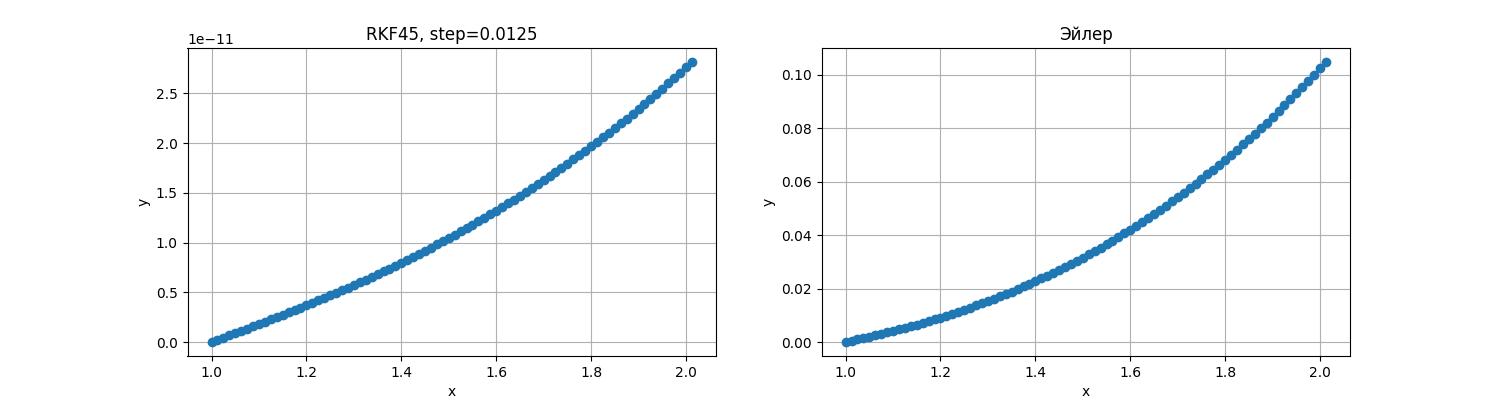
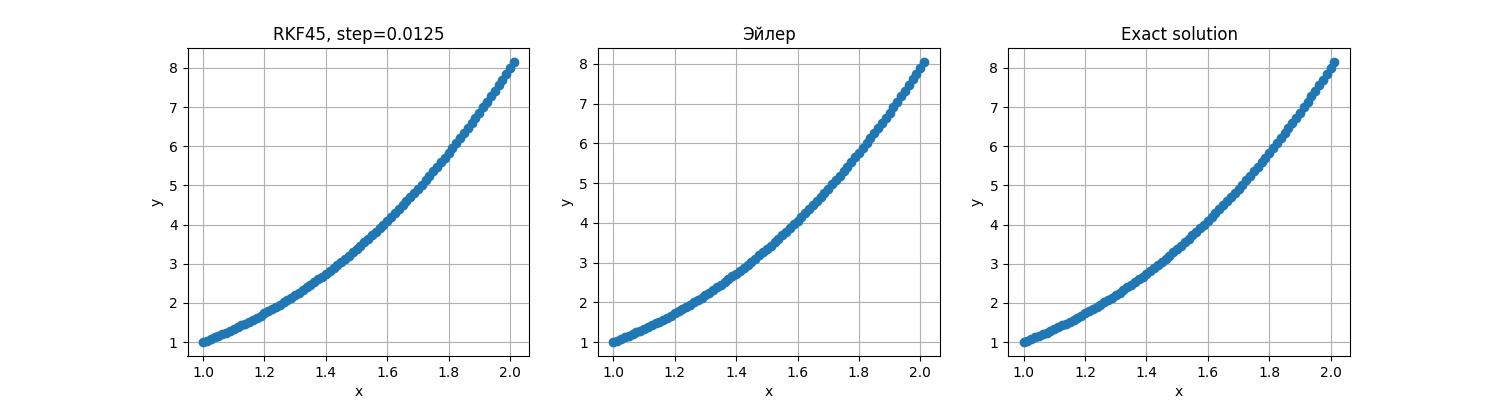


Далее не будем приводить таблицу полностью. Ограничимся лишь графики (вывод таблиц в разделе ссылки)

Для h\_int = 0.05  
Для h\_int = 0.025



Для h\_int = 0.0125



# Вывод:

Из табличных значений можно сделать выводы:

* Локальная погрешностей увеличивается к концу отрезка
* Глобальная погрешность уменьшается у RKF45
* RKF45 имеет сильно меньшую погрешность, чем метод ломанной Эйлера

# Ссылки:

Листин кода на github: [https://github.com/vitaya-para/lab](https://github.com/vitaya-para/lab3_2023/blob/main/main.py)3\_2023/blob/main/main.py

Полные выводы консоли: <https://github.com/vitaya-para/lab3_2023/blob/main/output.txt>