**ГБОУ «Президентский ФМЛ №239»**

**Визуализация физического явления, называемого акустической левитацией.**

*Годовой проект по информатике*

Работу выполнил

ученик 10-2 класса

Соколовский Александр

Учитель Информатики:

Филатов Владимир Константинович

Санкт-Петербург

2018

1. **Оглавление**
2. Оглавление……………………………………………………………………….2
3. Постановка задачи……………………………………………………………….3

2.1) Формулировка задачи……………………………………………………....3

2.2) Пояснение……………………………………………………………………3

1. Выбранный язык программирования…………………………………………...3
2. Уточнение исходных и выходных данных……………………………………..3

4.1) Исходные данные……………………………………………………………3

4.2) Выходные данные…………………………………………………….……..3

1. Математическая (физическая) модель……………………………………….…3
2. Анализ используемой структуры данных………………………………...……4
3. Метод решения…………………………………………………………..………4
4. Комментированный листинг………………………………………..…………..4
5. Пример работы программы………………………………………..……………5
6. Трудности во время проектирования и разработки…………...………………6
7. **Постановка задачи**
   1. *Формулировка задачи*
8. Создать визуализацию некого физического явления, называемого акустической левитацией.
9. Изучить новый язык программирования, который будет полезен в будущем (был выбран язык Python так как он широко используется физиками в их профессиональной деятельности для иллюстрации своих экспериментов и вычислений и для создания статей)
   1. *Пояснение*

Задать два синусоида (две стоячие волны); изобразить предмет, местонахождение которого соответствует физическим законам и реальным экспериментальным наблюдениям; график интерференции.

1. **Выбранный язык программирования.**

**Python** — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Основными преимуществами данного языка являются:

1. Широко используется физиками в различных целях.
2. Довольно прост в изучении, особенно на начальном этапе.
3. Особенности синтаксиса стимулируют программиста писать хорошо читаемый код.
4. Множество полезных библиотек и расширений языка можно легко использовать в своих проектах благодаря предельно унифицированному механизму импорта и программным интерфейсам.
5. **Уточнение исходных и выходных данных**
   1. *Исходные данные*

* На вход программе подается значение частоты излучаемых волн из которого находится длина волны и тд.
  1. *Выходные данные*
* Результатом работы является график, изображающий действие волн на предмет и на друг друга.

1. **Математическая (физическая) модель**

Итак, как уже было сказано, на вход программе подается значение частоты, из которого высчитывается длина волны (по формуле, связывающей частоту, скорость звука и длину волны: f=c/λ, λ - длина волны).

Далее по формуле (c/(fr\*2))\*(round(fr/50)) задается предел по х (где здесь и далее: с - скорость звука, а fr – частота).

По формуле sin((fr/c)\*2\*pi\*x) задаются синусоиды.

1. **Анализ используемой структуры данных**

Так как в программе не используется ни большое количество переменных, ни строки, то не задаются массивы или матрицы для хранения величин, а лишь хранятся как простейшие типы данных (int, float).

1. **Метод решения**
2. Получить значение величины частоты.
3. Задать пределы допустимых значений по осям x и y.
4. Задать функции и координаты объекта.
5. Отредактировать внешний вид всей картины.
6. Задать функцию отвечающую за видимость\невидимость объектов и функций.
7. Вывести на экран.
8. **Комментированный листинг**

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.widgets import CheckButtons

fr = float(input())/100 # frequency(частота)

c = 343 # speed of sound (скорость звука)

x = np.arange(0.0, (c/(fr\*2))\*(round(fr/50)) + 0.002, 0.001)

x1=np.arange((c/(fr\*2))\*(round(fr/50))/2 + c\*(round(fr/50)%2-1)/(fr\*4)-0.01,

(c/(fr\*2))\*(round(fr/50))/2 + c\*(round(fr/50)%2 - 1)/(fr\*4) + 0.01, 0.001)

# задаются пределы по осям

s0 = np.sin((fr/c)\*2\*np.pi\*x) # задаются функции для синусоидов,

s1 = np.sin(-(fr/c)\*2\*np.pi\*x) объекта и интерференции.

s2 = np.sin(0\*np.pi\*x1)

s3 = (np.sin(((fr/c)\*2\*np.pi) \* x))\*\*2

fig, ax = plt.subplots() # задаются основные параметры объектов (цвет, толщина, их обозначение в легенде и

изначальная видимость\невидимость)

l0, = ax.plot(x, s0, lw=2, color='r', label='1st radiant')

l1, = ax.plot(x, s1, lw=2, color='r', label='2nd radiant')

l2, = ax.plot(x1, s2, 'go', visible=False, lw=20, color='b', label='object')

l3, = ax.plot(x, s3, visible=False, lw=4, color='g', label='3nd radiant')

plt.subplots\_adjust(left=0.2) # смешение графика, чтобы влезали названия объектов

lines = [l0, l1, l2, l3] # создание окошка-легенды с функцией видимости\невидимости

rax = plt.axes([0.01, 0.4, 0.1, 0.15])

labels = [str(line.get\_label()) for line in lines]

visibility = [line.get\_visible() for line in lines]

check = CheckButtons(rax, labels, visibility)

def func(label): # метод для прорисовки объекта, при нажатии

index = labels.index(label)

lines[index].set\_visible(not lines[index].get\_visible())

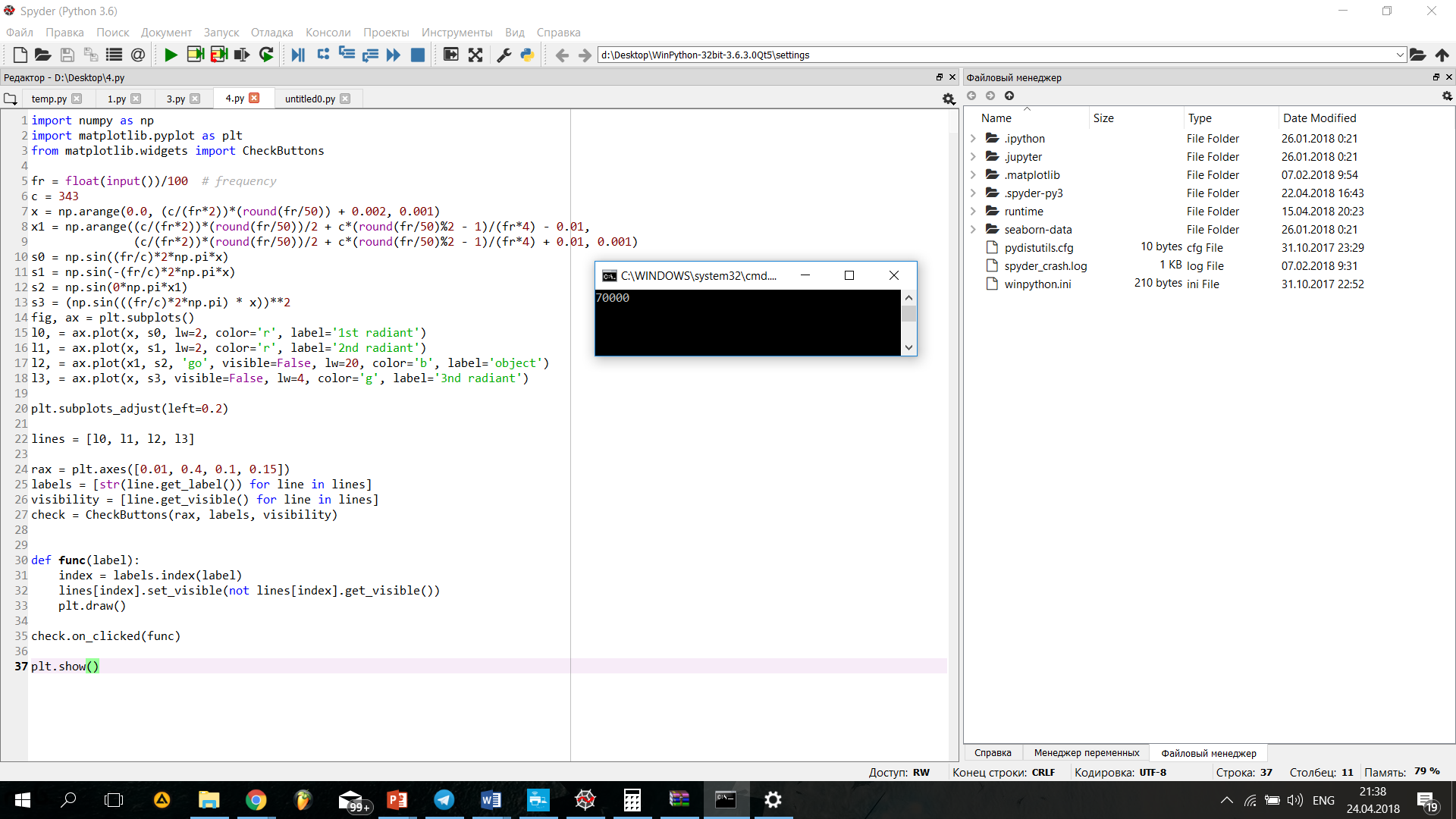
plt.draw()

check.on\_clicked(func)

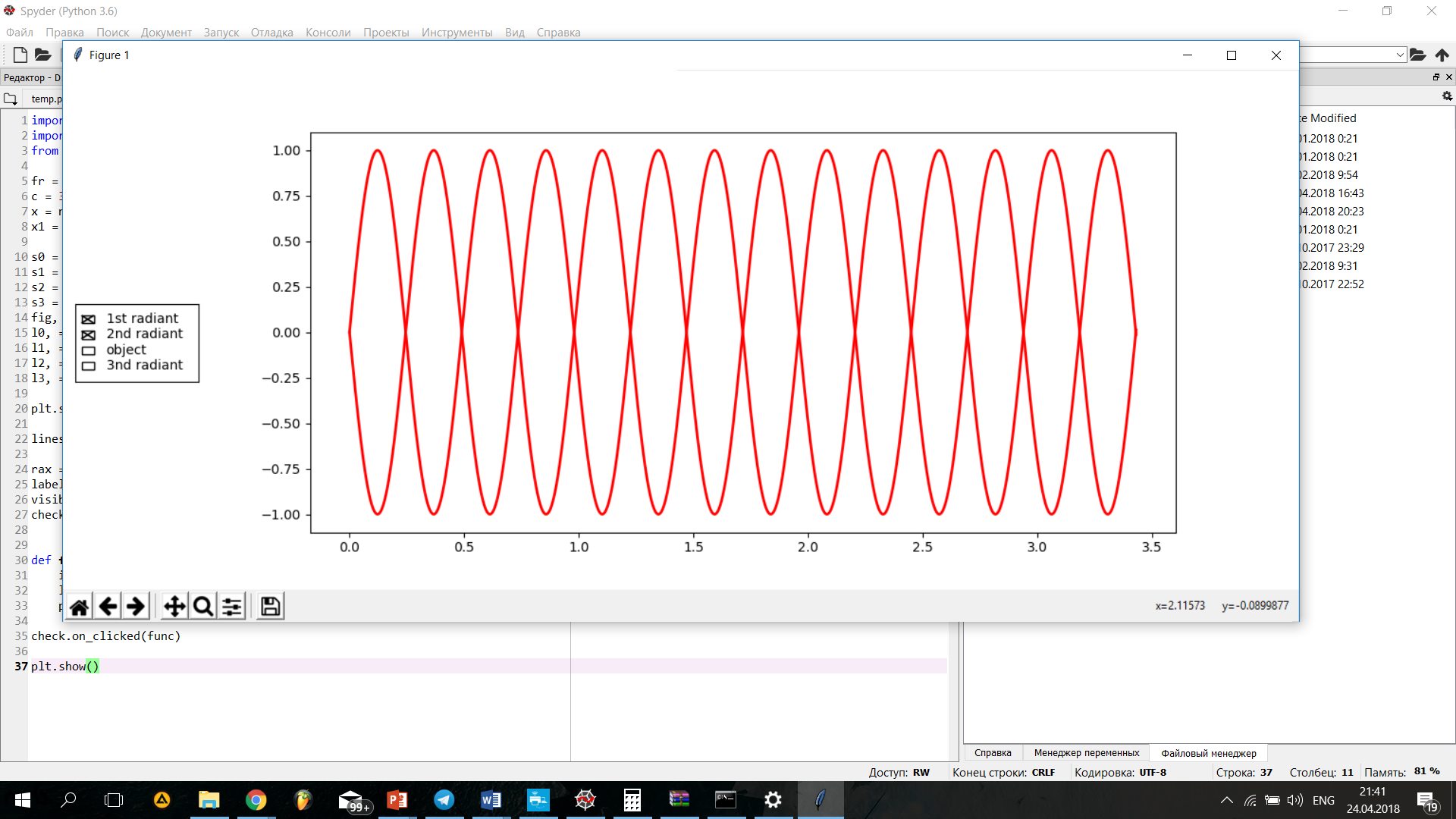
plt.show()

1. **Пример работы программы**

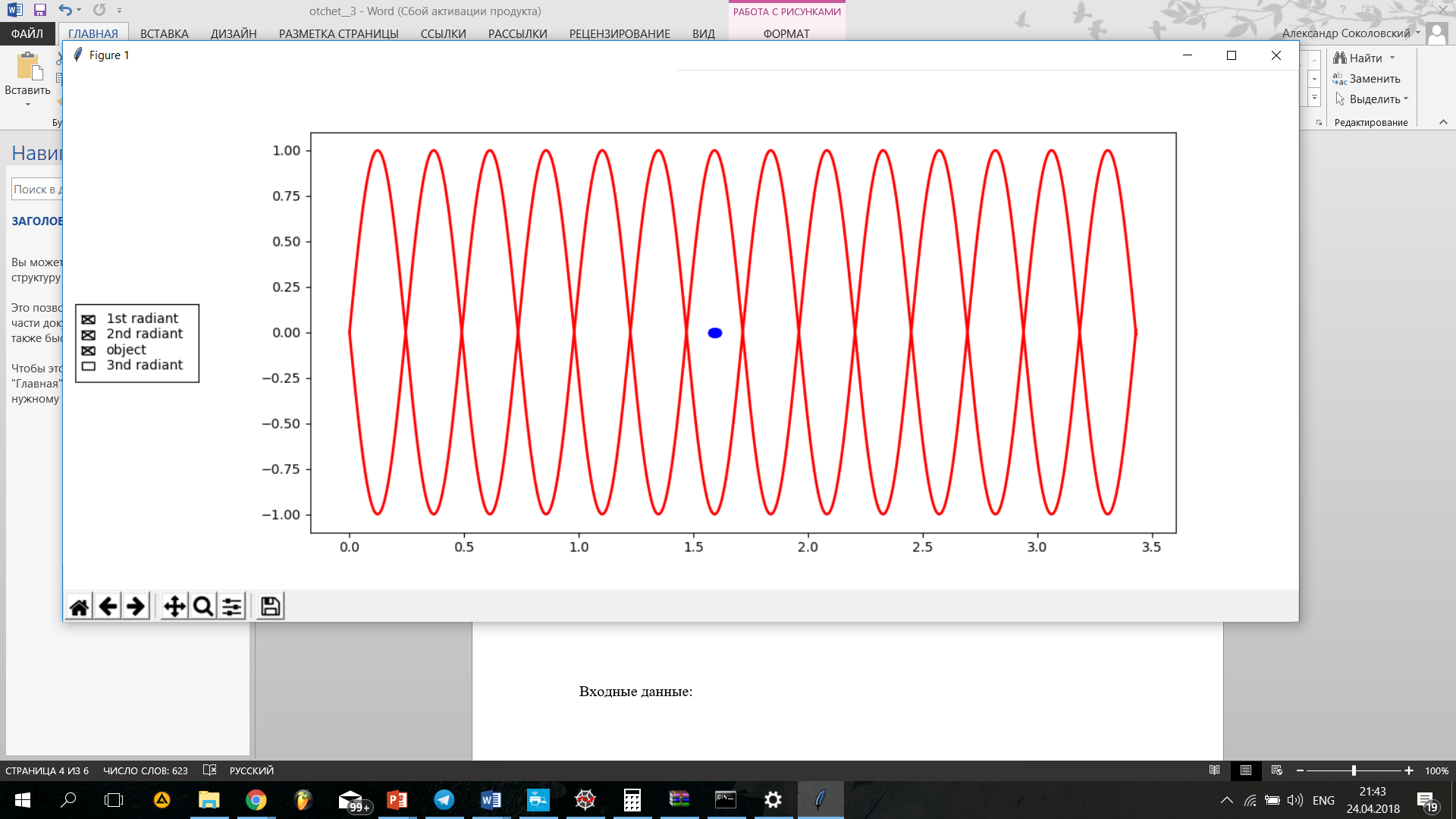
Входные данные:



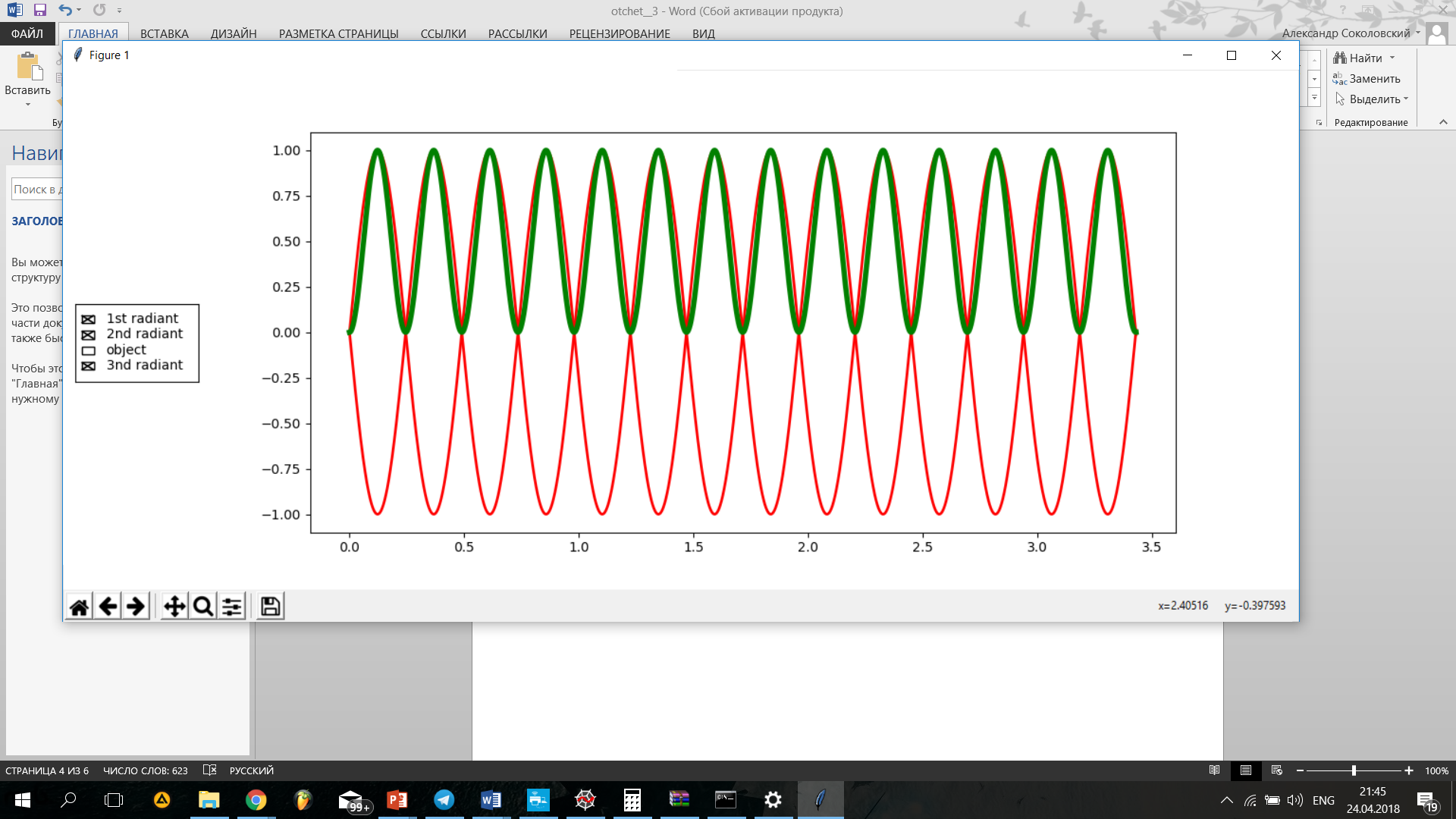
Результат решения:



**Изначальный вид графика.**



**Добавление обрисовки объекта.**



**Добавление обрисовки графика интерференции.**

1. **Трудности во время проектирования и разработки**
2. Выбор программной среды.
3. Выбор наиболее подходящей библиотеки для создания графика, удовлетворяющего установленным требованиям.
4. Сложность одновременного изображения графического объекта и графика.
5. Трудности постановки физического эксперимента.