МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«Санкт–Петербургский государственный университет
аэрокосмического приборостроения»

Кафедра №43 «Компьютерных технологий и программной инженерии»

ОТЧЁТ ПО ПРАКТИКЕ
ЗАЩИЩЁН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

Ст. преподаватель		С.А. Рогачев
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЁТ ПО ПРАКТИКЕ

вид практики производственная

тип практики технологическая (проектно-технологическая)

на тему индивидуального задания Программа для анализа текстов на предмет наличия определённых слов или фраз

выполнен Бобрович Николаем Сергеевичем
фамилия, имя, отчество обучающегося в творительном падеже

по направлению подготовки 02.03.03 Математическое администрирование и обеспечение информационных систем
код наименование направления
наименование направления

направленности 02.00.00 Системный анализ в информационных технологиях
код наименование направленности
наименование направленности

Обучающийся группы № 4136	13.07.2023	Бобрович Н. С.
номер	подпись, дата	инициалы, фамилия

Санкт–Петербург 2023

1. Оглавление.

2. Цель работы в соответствии с вариантом.

Программа для анализа текстов на предмет наличия определённых слов или фраз

3. Исходные данные и их описание.

Для поставленной задачи моими исходными данными будут:

1) Параметры (вводимые пользователем), необходимые для понимания какого формата текст нужно анализировать, а именно:
    1. Пользователю необходимо выбрать находится ли нужный ему текст в файле или нет;
    2. Если текст находится в файле, то необходимо, чтобы пользователь расположение этого файла;
    3. Если текст не находится в файле, то необходимо, чтобы пользователь ввёл его в соответствующем окне;
2) Текст, который нужно проверить;
3) Последовательность символов, на наличие которой нужно проверить полученный текст.

4. Теоретический раздел.

Программа разрабатывалась на основе уже имеющейся у меня программы генерации ИУЛов, написанной на языке Python, поэтому при написании этой программы был использован тот же язык программирования.

Был составлен примерный алгоритм выполнения программы, он представлен на примерной блок-схеме (не судите строго), предстваленной ниже:

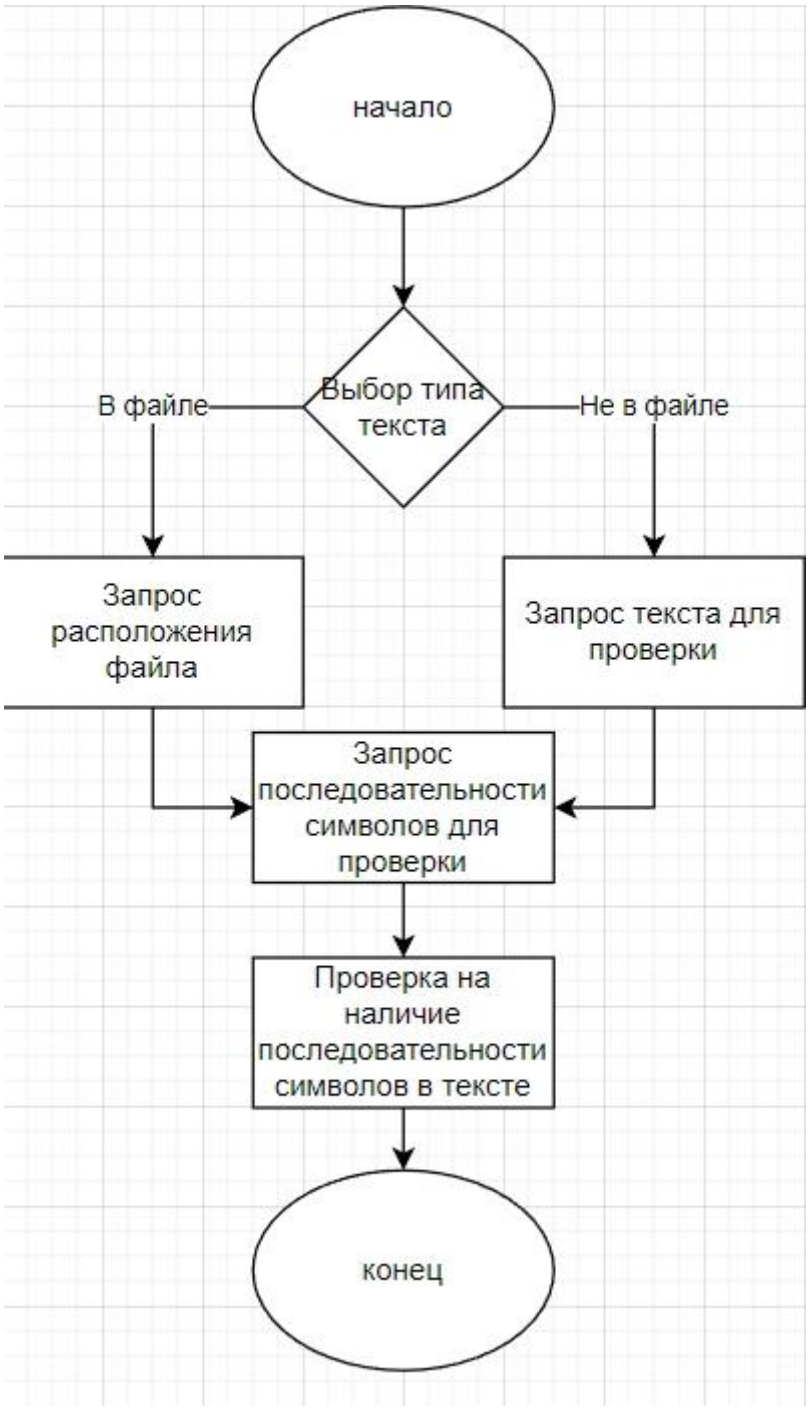


Рис. 1: Примерная блок-схема программы.

После составления блок-схемы был начат выбор метода поиска последовательности символов в тексте. Мной был выбран прямой метод метода поиска последовательности символов в тексте, так как его преимуществами являются:

1. Простота реализации: Прямой метод поиска подстроки является одним из самых простых алгоритмов поиска. Он основан на линейном переборе символов в тексте и сравнении каждого символа с соответствующим символом подстроки.

2. Низкая сложность: В прямом методе нет сложных вычислительных операций или структур данных. Это позволяет достичь лучшей производительности в случаях, когда обработка текста или строк происходит на маломасштабных данных.

3. Применимость к коротким и простым подстрокам: Прямой метод хорошо работает с короткими и простыми подстроками. Можно быстро найти все вхождения подстроки в строке без необходимости использовать сложные алгоритмы.

4. Прогрессивность поиска: Прямой метод позволяет найти все вхождения подстроки в тексте один за другим. После нахождения первого вхождения, можно продолжить поиск остальных вхождений, начиная с последнего найденного символа.

5. Адаптивность к изменяющимся данным: Поскольку прямой метод не требует предварительной обработки текста или подстроки, он может работать с изменяющимися данными. Если подстрока или текст изменяются, нет необходимости перестраивать или переиндексировать структуры данных.

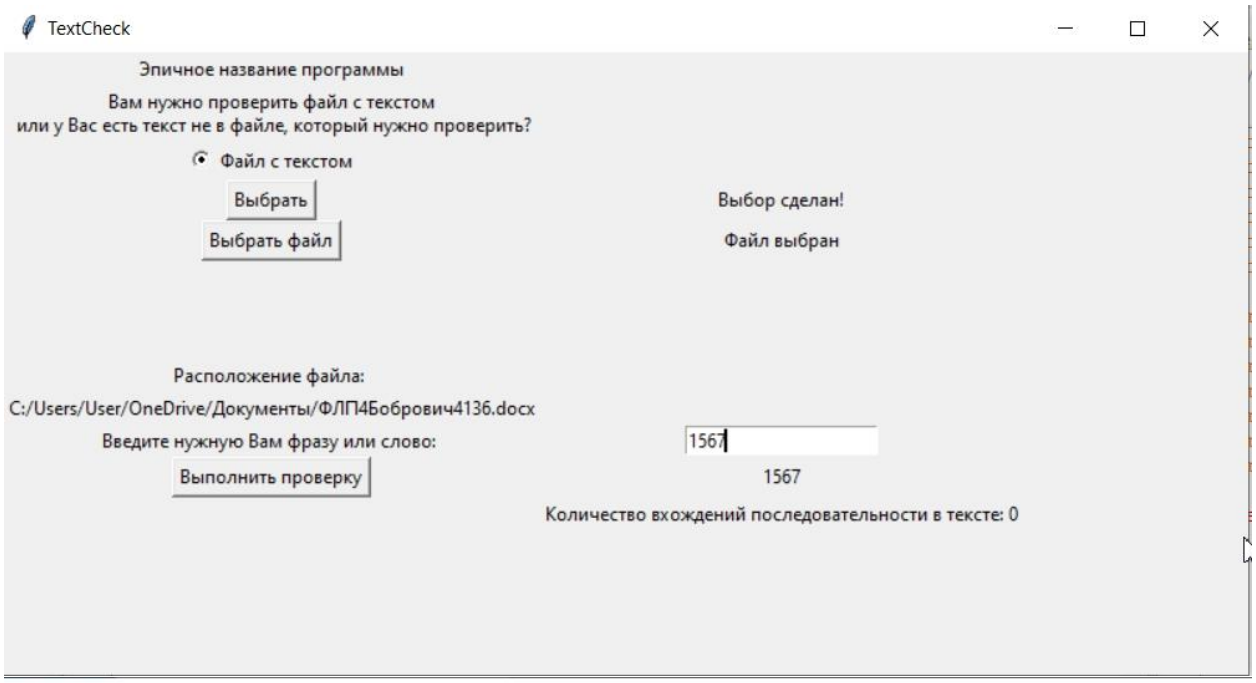
После выбора метода поиска последовательности символов в тексте я приступил к практической части задания.

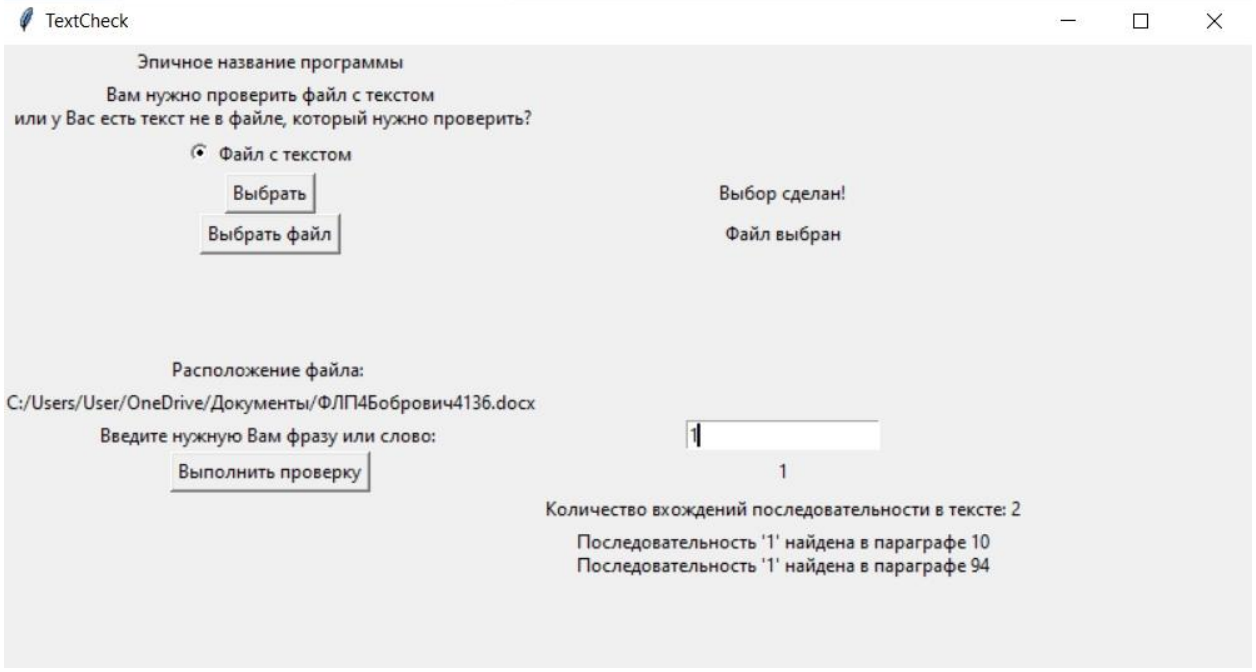
5. Практический раздел.

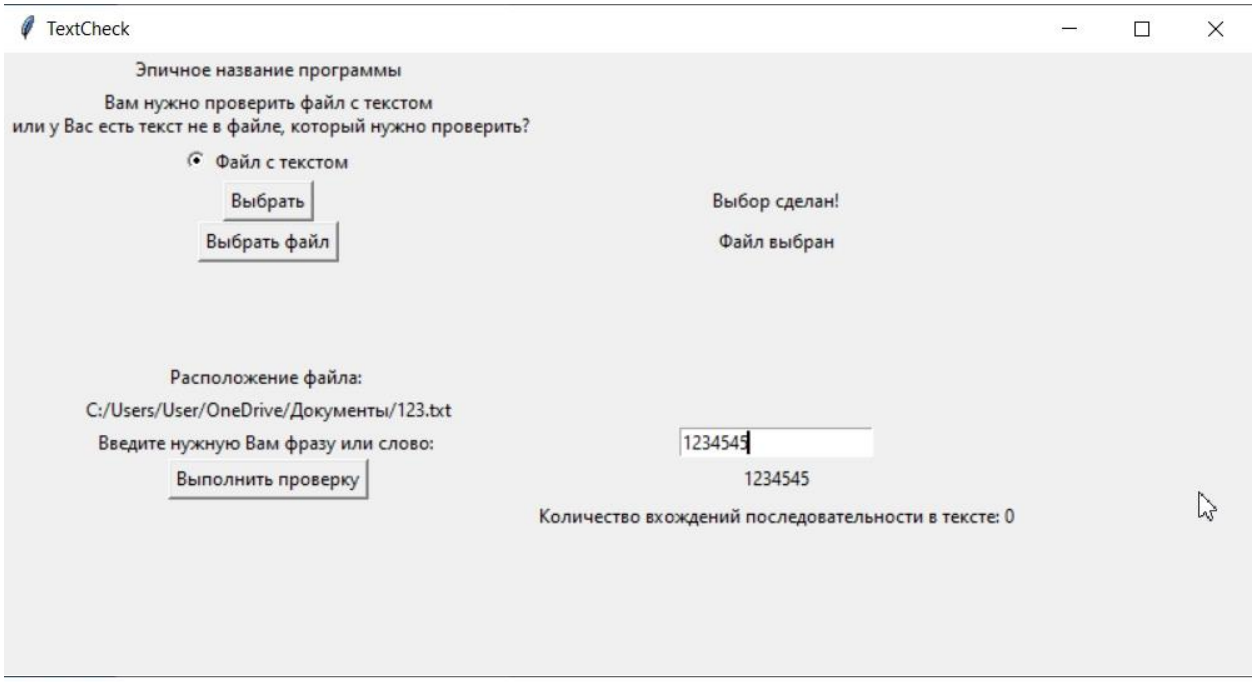
Мной был написан код, представленный в приложении 1, с помощью которого осуществлялись соответствующие условию и моим представлениям реализации этого условия задачи.

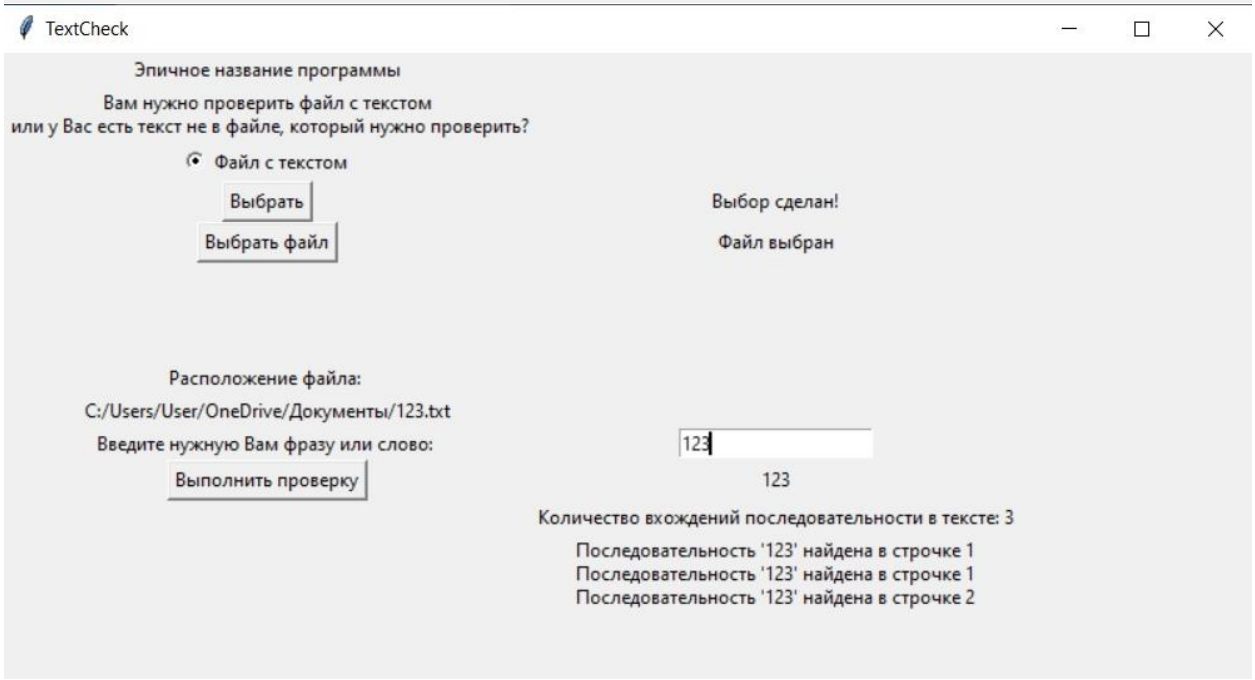
6. Результаты.

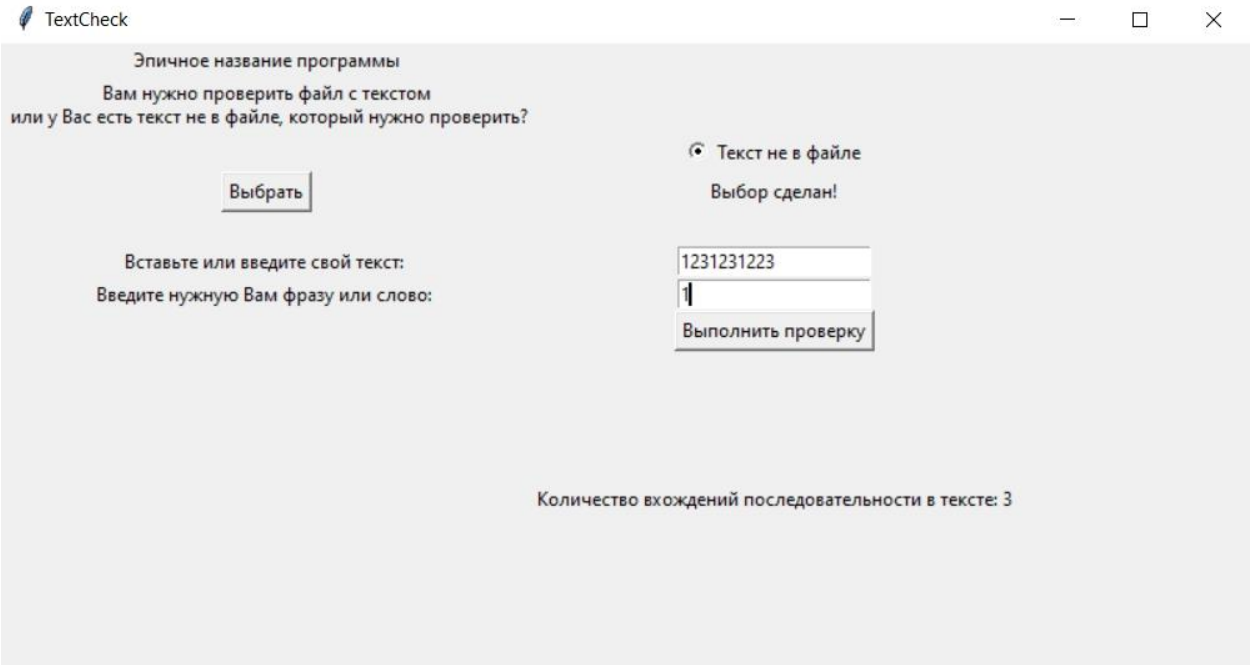
Результатом данной программы является сообщение о количестве найденных последовательностей символов в заданном тексте, а также (только в работе с текстом из файлов) расположение этих последовательностей, как показано в изображениях ниже.


TextCheck
Эпичное название программы
Вам нужно проверить файл с текстом
или у Вас есть текст не в файле, который нужно проверить?
Файл с текстом
Выбрать
Выбор сделан!
Выбрать файл
Файл выбран
Расположение файла:
C:/Users/User/OneDrive/Документы/ФЛП4Боброvич4136.docx
Введите нужную Вам фразу или слово:
1567
Выполнить проверку
1567
Количество вхождений последовательности в тексте: 0


TextCheck
Эпичное название программы
Вам нужно проверить файл с текстом
или у Вас есть текст не в файле, который нужно проверить?
Файл с текстом
Выбрать
Выбор сделан!
Выбрать файл
Файл выбран
Расположение файла:
C:/Users/User/OneDrive/Документы/ФЛП4Боброvич4136.docx
Введите нужную Вам фразу или слово:
1
Выполнить проверку
1
Количество вхождений последовательности в тексте: 2
Последовательность '1' найдена в параграфе 10
Последовательность '1' найдена в параграфе 94


TextCheck
Эпичное название программы
Вам нужно проверить файл с текстом
или у Вас есть текст не в файле, который нужно проверить?
Файл с текстом
Выбрать
Выбор сделан!
Выбрать файл
Файл выбран
Расположение файла:
C:/Users/User/OneDrive/Документы/123.txt
Введите нужную Вам фразу или слово:
1234545
Выполнить проверку
1234545
Количество вхождений последовательности в тексте: 0


TextCheck
Эпичное название программы
Вам нужно проверить файл с текстом
или у Вас есть текст не в файле, который нужно проверить?
Файл с текстом
Выбрать
Выбор сделан!
Выбрать файл
Файл выбран
Расположение файла:
C:/Users/User/OneDrive/Документы/123.txt
Введите нужную Вам фразу или слово:
123
Выполнить проверку
123
Количество вхождений последовательности в тексте: 3
Последовательность '123' найдена в строчке 1
Последовательность '123' найдена в строчке 1
Последовательность '123' найдена в строчке 2



7. Выводы.

В заключении хочется сказать, что благодаря проекту я не узнал практически ничего нового, зато повторил хорошо забытое старое, а, значит, стал немного умнее. И это прекрасно. Также хочется пожелать счастья, здоровья и семейного благополучия всем, кто это читает.

8. Список литературы.

1) Код: тайный язык информатики. Чарльз Петцольд.

2) Игра в имитацию. Алан Тьюринг.

3) Компиляторы. Джеффри Д. Ульман, Альфред Ахо, Моника С. Лам, Рави Сети.

4) Путь программиста. Джон Сонмез.

5) Кодеры за работой. Размышления о ремесле программиста. Питер Сейбел.

6) Джоэл о программировании. Джоэл Спольски.

9. Приложения.

Приложение 1:
Листинг PP1.py:

```
import tkinter as tk
import sys
import pathlib
import os
import docx
import docx2pdf

from tkinter import messagebox as mb
from tkinter import filedialog
from tkinter import *
from os import path
from docx import Document
```

```
from docx2pdf import convert
from PyPDF2 import PdfReader

#sys.path.append('../../')

def f_count0(text, sequence):
    count = 0
    for i in range(len(text) - len(sequence) + 1):
        if text[i:i+len(sequence)] == sequence:
            count += 1
    return count

def f_count1(file_path, sequence):
    count = 0
    o = ""
    with open(file_path, "r") as file:
        lines = file.readlines()
        for j, line in enumerate(lines):
            line = line.rstrip("\n")
            for i in range(len(line) - len(sequence) + 1):
                if line[i:i+len(sequence)] == sequence:
                    count += 1
                    o += f"Последовательность '{sequence}' найдена в строчке {j + 1}"
+ "\n"
    lbl14_0.configure(text= o)
    return count

def f_count2(file_path, sequence):
    count = 0
    o = ""
    doc = Document(file_path)
    paragraphs = doc.paragraphs
    for i, paragraph in enumerate(paragraphs):
        text = paragraph.text
        if sequence in text:
            count += 1
            o += f"Последовательность '{sequence}' найдена в параграфе {i + 1}" +
"\n"
    lbl14_0.configure(text= o)
    return count

def fbtn1():
    if selected1.get() == 1:
        lbl3.configure(text="Выбор сделан!")
        btn2 = Button(window, text="Выбрать файл", command = fbtn2)
```

```
        btn2.grid(column=1, row=5)
        rad2.grid_remove()

    if selected1.get() == 2:
        lbl3.configure(text="Выбор сделан!")
        lbl4_1.configure(text="Вставьте или введите свой текст: ")
        entry0.grid(column=2, row=6)
        lbl13.configure(text="Введите нужную Вам фразу или слово: ")
        entry00.grid(column=2, row=7)
        rad1.grid_remove()
        btn5 = Button(window, text="Выполнить проверку", command=fbtn5)
        btn5.grid(column=2, row=8)

def fbtn2():
    window0 = tk.Tk()
    window0.title("Выбор файла")
    window0.geometry('300x200')
    folder_path = StringVar()
    lbl0 = Label(master=window0,textvariable=folder_path)
    lbl0.grid(column=1, row=5)
    lbl = Label(window0)
    lbl.grid(column=1, row=0)
    btn2 = Button(window0, text="Выбрать файл", command=browse_button)
    btn2.grid(column=1, row=0)
    window0.mainloop()

def fbtn4_0():
    btn4 = Button(window, text="Выполнить проверку", command=fbtn4_0)
    btn4.grid(column=1, row=12)
    global folder_path
    folder_path.get()
    paths = []
    folder = folder_path.get()
    t = entry.get()
    lbl9['text'] = t
    with open (fn(folder), 'r') as file:
        lbl11.configure(text= "Количество вхождений последовательности в
тексте: " + str(f_count1(fn(folder_path.get()), t)))

def fbtn4_1():
    btn4 = Button(window, text="Выполнить проверку", command=fbtn4_1)
    btn4.grid(column=1, row=12)
    global folder_path
    folder_path.get()
    paths = []
```

```
    folder = folder_path.get()
    t = entry.get()
    lbl9['text'] = t
    doc = docx.Document(fn(folder))
    ftext = []
    for para in doc.paragraphs:
        ftext.append(para.text)
    lbl11.configure(text= "Количество вхождений последовательности в тексте: " + str(f_count2(fn(folder_path.get()), t)))

def fbtn5():
    s = entry00.get()
    t = entry0.get()
    lbl11.configure(text= "Количество вхождений последовательности в тексте: " + str(f_count0(t, s)))

def fn(filename):
    parts = filename.split("'")
    if len(parts) > 1:
        return parts[1]
    else:
        return ''

def browse_button():
    filename = filedialog.askopenfile()
    global folder_path
    if (filename != ''):
        lbl4_0.configure(text="Расположение файла: ")

        folder_path.set(filename)
        folder = folder_path.get()
        lbl5.configure(text=fn(folder))

        lbl0.configure(text="Файл выбран")

        entry.grid(column=2, row=11)
        lbl8.configure(text="Введите нужную Вам фразу или слово: ")

        f, s = os.path.splitext(fn(folder))

        if s == ".txt":
            btn4 = Button(window, text="Выполнить проверку", command=fbtn4_0)
            btn4.grid(column=1, row=12)
        if s == ".docx":
            btn4 = Button(window, text="Выполнить проверку", command=fbtn4_1)
```

```
            btn4.grid(column=1, row=12)
    else:
        lbl0.configure(text="Файл не выбран!")

window = tk.Tk()
window.title("TextCheck")
window.geometry('800x400')

folder_path = StringVar()
s = StringVar()
selected1 = IntVar()
selected2 = IntVar()

entry = Entry()
entry0 = Entry()
entry00 = Entry()

lbl1 = Label(window)
lbl2 = Label(window)
lbl1.configure(text="Эпичное название программы")
lbl2.configure(text="Вам нужно проверить файл с текстом\n или у Вас есть текст не в файле, который нужно проверить?")
lbl1.grid(column=1, row=1)
lbl2.grid(column=1, row=2)

rad1 = Radiobutton(window, text='Файл с текстом', value=1, variable=selected1)
rad2 = Radiobutton(window, text='Текст не в файле', value=2, variable=selected1)
rad1.grid(column=1, row=3)
rad2.grid(column=2, row=3)

btn1 = Button(window, text="Выбрать", command = fbtn1)
btn1.grid(column=1, row=4)

lbl3 = Label(window)
lbl3.grid(column=2, row=4)

lbl4_0 = Label(window)
lbl4_1 = Label(window)
lbl4_0.grid(column=1, row=9)
lbl4_1.grid(column=1, row=6)

lbl0 = Label(window)
lbl0.grid(column=2, row=5)
```

```
lbl5 = Label(window)
lbl6 = Label(window)
lbl5.grid(column=1, row=10)
lbl6.grid(column=1, row=8)

lbl7 = Label(window)
lbl7.grid(column=2, row=10)

lbl8 = Label(window)
lbl8.grid(column=1, row=11)

lbl9 = Label(window)
lbl9.grid(column=2, row=12)

lbl10 = Label(window)
lbl10.grid(column=1, row=13)

lbl11 = Label(window)
lbl11.grid(column=2, row=13)

lbl12 = Label(window)
lbl12.grid(column=2, row=9)

lbl13 = Label(window)
lbl13.grid(column=1, row=7)

lbl14_0 = Label(window)
lbl14_0.grid(column=2, row=14)

lbl14_1 = Label(window)
lbl14_1.grid(column=1, row=14)

window.mainloop()

#проверка, если нужно
#print('Готово!')
```