**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,**

**СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА» (СПбГУТ)**

Факультет: Инфокоммуникационных сетей и систем Кафедра: Защищенных систем связи

Дисциплина: Защищенные операционные системы

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6.1

Графический интерфейс

*(тема отчета)*

Направление/специальность подготовки

10.03.01 Информационная безопасность

*(код и наименование направления/специальности)*

Студент:

Емельянов Н.Р. ИКБ-32

*(Ф.И.О., № группы) (подпись)*

Михайлов И.Д. ИКБ-32

*(Ф.И.О., № группы) (подпись)*

Преподаватель:

Асс. Смирнов Д.Н.

*(Должность, Ф.И.О. преподавателя) (подпись)*

Санкт-Петербург 2024

**Цель**

Написать приложение для визуализации данных с Matplotlib

**Задачи**

Основные задачи на лабораторную работу:

1. Написать программу для визуализации данных из CSV-файла с помощью библиотеки Matplotlib
2. Создать графический интерфейс с использованием PySimpleGUI, позволяющий пользователю выбирать тип графика и вводить данные.
3. Реализовать функции для построения выбранного графика на основе введенных данных.
4. Добавить возможность сохранения графика в файл различных форматов

**Ход выполнения**

Шаги выполнения лабораторной работы:

1. Импортируем библиотеки: PySimpleGUI, Matplotlib и CSV;
2. Создаем функцию is\_number, которая проверяет, является ли входная строка числом;
3. Реализуем функцию convert\_to\_normal\_format:
   1. Функция принимает путь к CSV файлу;
   2. Читает данные из файла;
   3. Конвертирует числа во float;
   4. Форматирует данные для построения графика;
4. Вводим функцию display\_graphic:
   1. принимает тип графика (линейный, рассеянный, столбчатый);
   2. создает график с помощью Matplotlib и отображает его;
5. При помощи PySimpleGUI создаем макет окна с элементами для выбора файла, кнопкой загрузки значений и кнопками для выбора типа графика
6. Создаем цикл обработки событий, в теле которого определяется состояние (доступна или нет) кнопок в зависимости от того, загружен ли файл, анализируется событие, которое выбирает пользователь (тип графика) и выводится на экран соответствующий график при помощи функции display\_graphic()

**Листинг программы**

import PySimpleGUI as sg

import matplotlib.pyplot as plt

import csv

def is\_number(input\_str):

try:

float(input\_str)

return True

except ValueError:

return False

def convert\_to\_normal\_format(file\_path):

list\_of\_floats = []

with open(file\_path, "r") as file:

csv\_reader = csv.reader(file)

data\_list = []

for row in csv\_reader:

data\_list.append(row)

for row in data\_list:

for item in row:

if is\_number(item):

list\_of\_floats.append(float(item))

x = []

y = []

for i in range(1, len(data\_list)):

x.append(data\_list[i][0])

y.append(int(data\_list[i][1]))

xlabel = data\_list[0][0]

ylabel = data\_list[0][1]

return x, y, xlabel, ylabel

def display\_graphic(plt\_type):

plt.figure(figsize=(11, 8))

if plt\_type == 'plot':

plt.plot(x, y, label=xlabel)

elif plt\_type == 'scatter':

plt.scatter(x, y, label=xlabel)

elif plt\_type == 'bar':

plt.bar(x, y, label=xlabel)

plt.xlabel(xlabel)

plt.ylabel(ylabel)

plt.title(ylabel)

plt.legend()

plt.show()

sg.theme("GreenMono")

layout = [

[sg.T("")],

[

sg.Text("Выберите файл: "),

sg.Input(),

sg.FileBrowse("Обзор", key="-IN-", file\_types=(("CSV Files", "\*.csv"),)),

],

[sg.Button("Загрузить значения")],

[sg.Button("Линейный", disabled=True), sg.Button("Scatterplot", disabled=True), sg.Button("Столбчатый", disabled=True)]

]

window = sg.Window("Визуализация CSV", layout, size=(600, 200))

while True:

event, values = window.read()

if event == sg.WIN\_CLOSED or event == "Exit":

break

elif event == "Загрузить значения":

csv\_file\_path = values["-IN-"]

x, y, xlabel, ylabel = convert\_to\_normal\_format(csv\_file\_path)

window['Загрузить значения'].update(disabled=True)

window['Линейный'].update(disabled=False)

window['Scatterplot'].update(disabled=False)

window['Столбчатый'].update(disabled=False)

elif event == "Линейный" and x and y:

display\_graphic('plot')

elif event == "Scatterplot" and x and y:

display\_graphic('scatter')

elif event == "Столбчатый" and x and y:

display\_graphic('bar')

На рисунках 1, 2 представлено окно с элементами для выбора файла, кнопкой загрузки значений и кнопками для выбора типа графика:

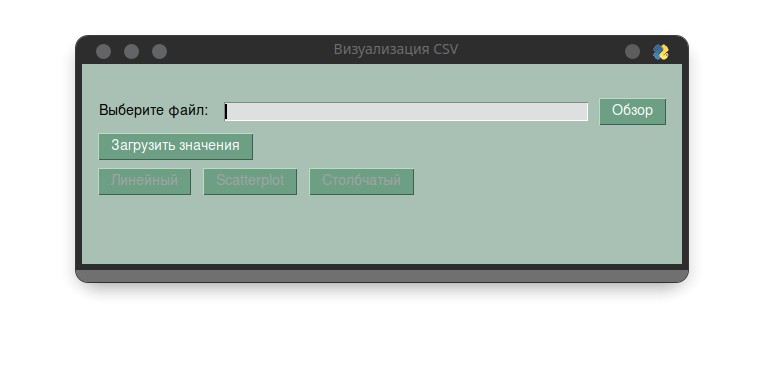
****

Рис. 1 – Окно (не указан путь к файлу)

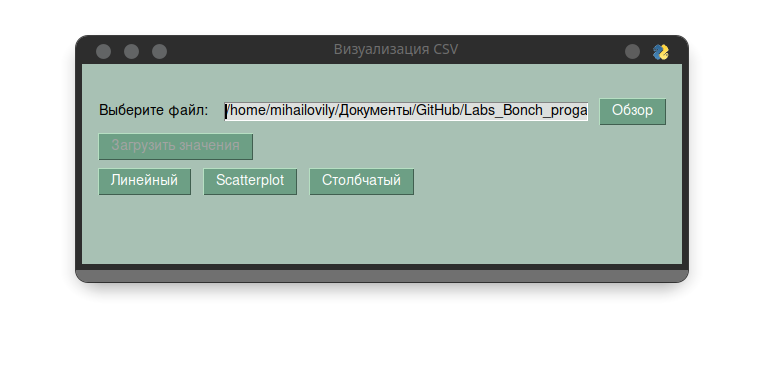


Рис. 2 – Окно (указан путь к файлу)

На рисунках 3, 4, 5 представлены графики линейного, рассеянного и столбчатого графиков соответственно:

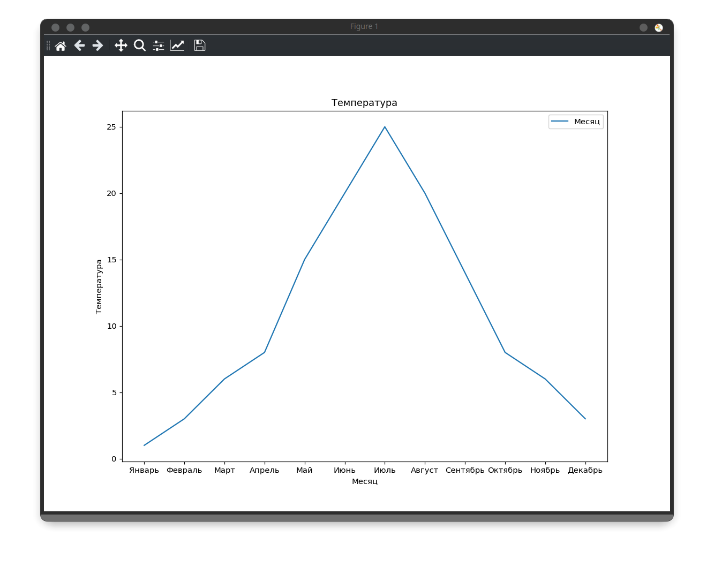


Рис. 3 – график линейного типа

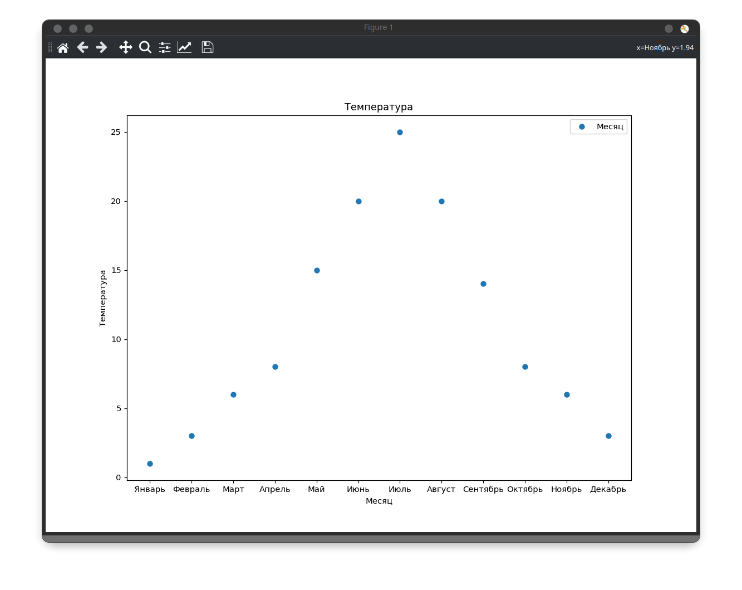


Рис. 4 – график рассеянного типа

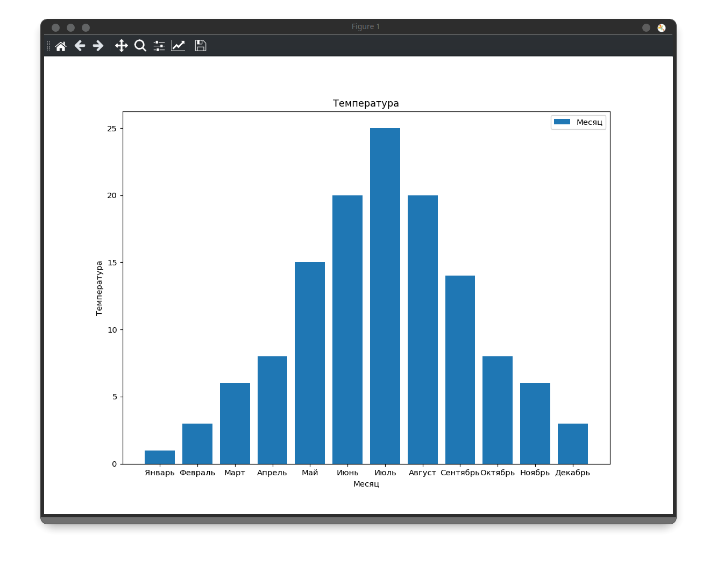


Рис. 5 – график столбчатого типа

**Вывод**

В ходе выполнения лабораторной работы было написано приложение для визуализации данных с Matplotlib Это было реализовано при помощи следующих шагов:

1. Импортировали необходимые библиотеки (PySimpleGUI, Matplotlib, CSV);
2. Создали проверку на входные данные от пользователя
3. Реализовали функцию, которая форматирует данные файла в необходимый формат для построения графика;
4. Создали GUI для работы с пользователем и реализовали цикл обработки событий;