Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение   
высшего образования**  
**«Новгородский государственный университет имени Ярослава Мудрого» (НовГУ)**

Великий Новгород

**Методы защиты информации.**

Лабораторная работа №1.

Отчёт.

Студент гр. 3091

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ильин Д. А.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Жгун Т. В.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

г. Великий Новгород

-2024-

Оглавление

[План работы 3](#_Toc185332637)

[Теоретическое положение 4](#_Toc185332638)

[Алгоритм работы программы 5](#_Toc185332639)

[Приложения А 6](#_Toc185332640)

[Приложение Б 9](#_Toc185332641)

# План работы

1. Реализовать инструмент, обеспечивающий частотный анализ файлов графического (\*.bmp, 3 канала по 8 бит), и текстового формата (\*.txt, 8 бит). Кодировка символов – ASCII представлена в таблице № 1. Все форматы представлены числами в диапазоне [0, 255].
2. Анализ файла представляет собою гистограмму текста и таблицу встречаемости символов. Все гистограммы должны быть реализованы для одного диапазона [0, 255].
3. Проанализировать распределение символов для графических файлов различного свойства (картинка с преобладанием одного цвета, картинка в графическом редакторе, фото, …) и текстов разного свойства (текст учебника, статья в СМИ, литературный текст, переписка в соцсетях, …). Объем текстового файла не менее 2 тысяч символов.

# Теоретическое положение

Частотный анализ представляет собой методику, применяемую в криптографии и лингвистике для изучения частотности появления символов, букв или слов в тексте. Данный метод основывается на предположении, что в любом языке определенные символы или их сочетания встречаются с разной частотой.

Одной из возможностей программы станет расчет энтропии для заданного файла. Энтропия, согласно теории информации, отражает среднее количество информации в сообщении и характеризует степень неопределенности, связанную с вероятностным распределением возможных исходов, которая рассчитывается по формуле:

Если энтропия низкая, система считается предсказуемой. Например, текст с частыми повторами одних и тех же слов имеет низкую энтропию, так как его информационное разнообразие минимально.

Наоборот, высокая энтропия свидетельствует о большой неопределенности. В тексте, содержащем множество уникальных слов с примерно равной частотой, предсказать следующее слово сложнее, что указывает на высокий уровень энтропии.

# Алгоритм работы программы

Программа представляет собой графическое приложение для анализа текстовых и графических файлов. Ее основная цель — построение гистограммы частот символов или значений пикселей и вычисление энтропии файла. Пользователь взаимодействует с приложением через удобный графический интерфейс на базе библиотеки Tkinter.

После запуска приложения перед пользователем открывается окно с кнопкой для выбора файла и областью для вывода результатов. Нажав на кнопку, пользователь открывает системный менеджер файлов и выбирает текстовый файл (.txt ) или графический файл (.png, .jpg, .jpeg, .bmp).

Когда файл выбран, программа анализирует его тип. Для текстовых файлов считываются символы, которые преобразуются в числовые значения в кодировке cp1251, после чего строится массив частот символов. Для графических файлов изображение конвертируется в массив яркостей пикселей, а затем также вычисляются частоты.

На основе этих данных программа вычисляет энтропию файла, используя вероятности появления каждого символа или пикселя. Энтропия отражает степень неупорядоченности или случайности содержимого файла.

Результаты анализа визуализируются в приложении. Построенная гистограмма показывает распределение символов или яркостей пикселей, а значение энтропии выводится текстом. Интерфейс позволяет пользователю повторять анализ, выбирая новые файлы.

Если возникает ошибка, например, файл имеет неподдерживаемый формат или поврежден, программа уведомляет об этом с помощью всплывающего сообщения. Программа продолжает работать до тех пор, пока пользователь не закроет окно.

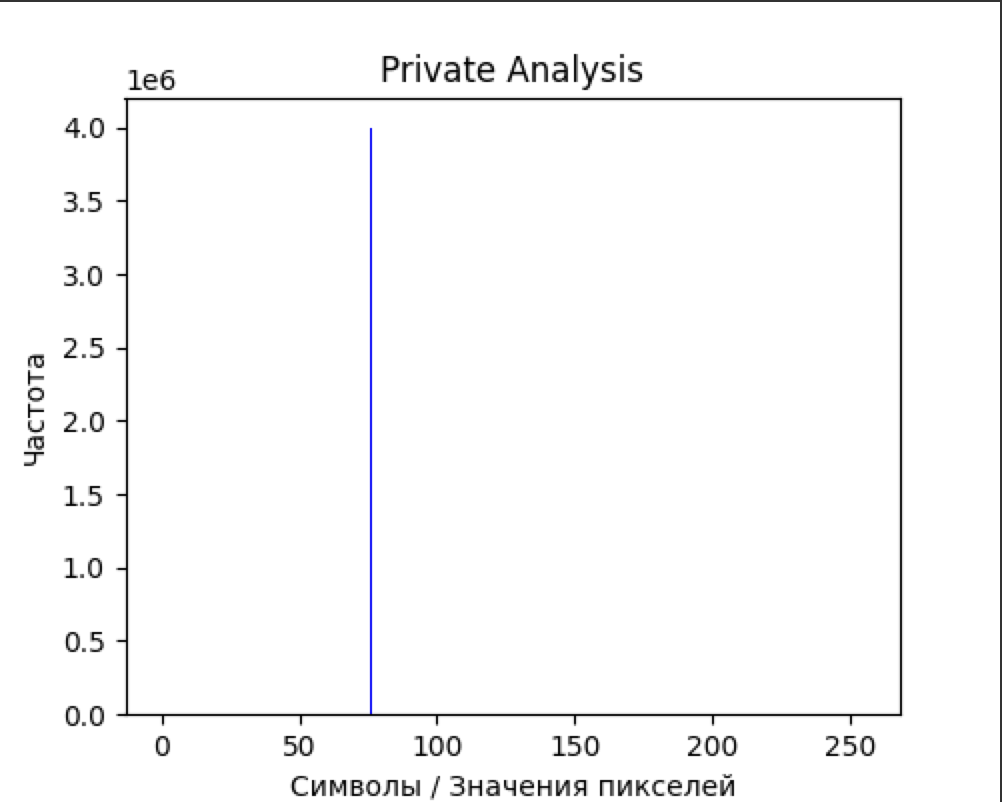
Для демонстрации в приложения A данного отчета предоставлены примеры работы программы. В приложение Б расположен исходный код приложения.

# Приложения А

Изображение выглядит как красный, Карминный цвет

Автоматически созданное описание

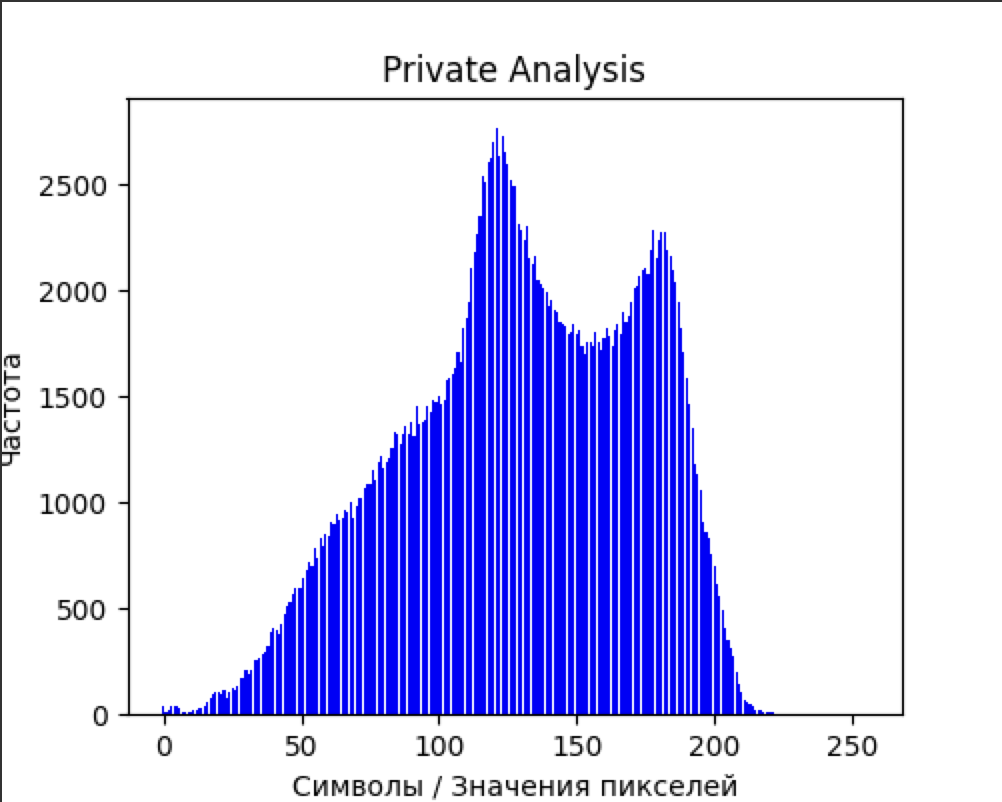
Изображение red.bmp



Изображение выглядит как примат, картина, обезьяна, живая природа

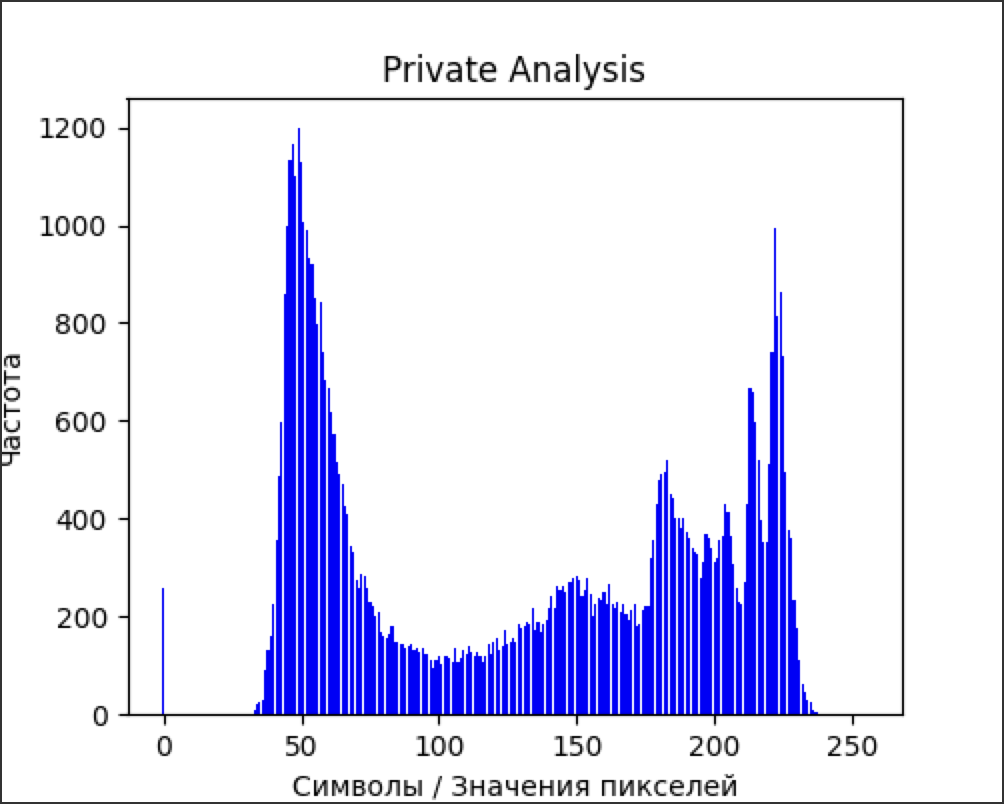
Автоматически созданное описание

Изображение mandrill.bmp





Изображение tree.bmp



# Приложение Б

import tkinter as tk

from tkinter import filedialog, messagebox

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

import numpy as np

from PIL import Image

# Функции для анализа

def privateAnalysisText(path: str, encode="cp1251") -> np.ndarray:

privateAnalysis = np.zeros(shape=256, dtype=int)

with open(path, 'r', encoding=encode) as f:

text = f.read()

for char in text:

privateAnalysis[char.encode(encode)[0]] += 1

return privateAnalysis

def privateAnalysisPhoto(path: str) -> np.ndarray:

privateAnalysis = np.zeros(shape=256, dtype=int)

img = Image.open(path).convert('L')

dataPixelIn1D = np.array(img).reshape(-1)

for pixel in dataPixelIn1D:

privateAnalysis[pixel] += 1

return privateAnalysis

def entropy(privateAnalysis):

total = sum(privateAnalysis)

probabilities = privateAnalysis / total

entropy = -np.nansum(probabilities \* np.log2(probabilities, where=privateAnalysis != 0))

return entropy

# Класс приложения на Tkinter

class PrivateAnalysisApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Private Analysis Application")

self.root.geometry("600x600")

# Метки и кнопки

self.label = tk.Label(root, text="Выберите файл для анализа", font=("Arial", 16))

self.label.pack(pady=10)

self.btn\_select\_file = tk.Button(root, text="Выбрать файл", command=self.load\_file, font=("Arial", 12))

self.btn\_select\_file.pack(pady=5)

# Гистограмма

self.figure = plt.Figure(figsize=(5, 4), dpi=100)

self.ax = self.figure.add\_subplot(111)

self.canvas = FigureCanvasTkAgg(self.figure, root)

self.canvas.get\_tk\_widget().pack(padx=10, pady=10)

# Энтропия

self.entropy\_label = tk.Label(root, text="", font=("Arial", 14))

self.entropy\_label.pack(pady=10)

def load\_file(self):

file\_path = filedialog.askopenfilename()

if not file\_path:

messagebox.showwarning("Ошибка", "Файл не выбран")

return

try:

# Определяем, текст это или картинка

if file\_path.endswith(('.txt')):

analysis = privateAnalysisText(file\_path)

elif file\_path.endswith(('.png', '.jpg', '.jpeg', '.bmp')):

analysis = privateAnalysisPhoto(file\_path)

else:

messagebox.showerror("Ошибка", "Неподдерживаемый формат файла")

return

# Вычисляем энтропию

ent = entropy(analysis)

self.entropy\_label.config(text=f"Энтропия файла: {ent:.4f}")

# Строим график

self.ax.clear()

self.ax.bar(range(256), analysis, color='blue')

self.ax.set\_title("Private Analysis")

self.ax.set\_xlabel("Символы / Значения пикселей")

self.ax.set\_ylabel("Частота")

self.canvas.draw()

except Exception as e:

messagebox.showerror("Ошибка", f"Произошла ошибка: {e}")

# Точка входа в приложение

if \_\_name\_\_ == "\_\_main\_\_":

root = tk.Tk()

app = PrivateAnalysisApp(root)

root.mainloop()