Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и
информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

Представление Графической Информации
Лабораторная работа №8

Выполнил:
Студент IV курса ИВТ,
группы ИП-713
Михеев Никита Алексеевич

Работу проверил:
доцент кафедры ПМиК
Перцев И.В.

Новосибирск 2020 г.

Задание

Вывести на экран PCX файл.

Результат работы

В результате работы была реализована программа на языке Python версии 3.9. Сначала считывается изображение и считывается длина и ширина изображения из заголовка PCX-файла (первые 128 байт файла). Далее считывается цветовая палитра из последних 768 байт файла, откуда 768 = 256 цветов * 3 цвета, т.е. на каждый цвет отводится по байту. Затем идет считывание файла, сначала идет проверка старших битов, если биты равны единицам, то мы берем оставшиеся 6 бит, которые содержат значения длины серии, и заполняем в отдельный массив для хранения цветов из нашей палитры, которая была создана ранее. Иначе байт считается за цвет и так же добавляется в массив. После того, как мы прошли по всему файлу идет запуск графического окна и попиксельное заполнение изображение на основании имеющегося у нас массива.

Рис. 1 – демонстрация работы программы.

Листинг

Lab8.py:

```
import io

import pygame
from PIL import Image
```

```
image = Image.open("200001.PCX")
image_bytes = io.BytesIO()
image.save(image_bytes, "PCX")
image_bytes = bytes(image_bytes.getvalue())

# Constants
FPS = 30
W = int.from_bytes(image_bytes[8:10], "little") + 1
H = int.from_bytes(image_bytes[10:12], "little") + 1

pygame.init()
pygame.display.set_caption("Lab8")
sc = pygame.display.set_mode((W, H))
clock = pygame.time.Clock()


def main():
    color_palette = [
        (image_bytes[i], image_bytes[i + 1], image_bytes[i + 2])
        for i in range(len(image_bytes) - 768, len(image_bytes), 3)
    ]

    i = 128
    decoded_image = []
    while i < len(image_bytes) - 768:
        if (image_bytes[i] & 0xC0) == 0xC0:
            row = image_bytes[i] & 0x3F
            for _ in range(row):
                decoded_image.append(color_palette[image_bytes[i + 1]])
            i += 2
        else:
            decoded_image.append(color_palette[image_bytes[i + 1]])
            i += 1

    while True:
        clock.tick(FPS)

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                exit(0)

        for i in range(H):
            for j in range(W):
                # print(decoded_image[i * W + j])
                pygame.draw.line(sc, decoded_image[i * W + j], (j, i), (j,
i))
        pygame.display.update()


if __name__ == "__main__":
    exit(main())
```