МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет "Информатика и системы управления"
Кафедра "Системы обработки информации и управления"

Дисциплина "Парадигмы и конструкции языков программирования"

Отчет по лабораторной работе №6
"Разработка простого бота для Telegram с использованием языка Python"

Выполнил:
Студент группы ИУ5-35Б
Королев М.О.
Преподаватель:
Гапанюк Ю.Е.

Москва 2025

Задание:

Разработайте простого бота для Telegram. Бот должен использовать функциональность создания кнопок

https://github.com/koromax/Korolev-PCPL-Labs-2025/tree/main/lab5

Листинг кода:

```
# bot.py
from PIL import Image
from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup, InputMediaPhoto
from telegram.ext import Application, CommandHandler, MessageHandler, filters, CallbackContext,
CallbackQueryHandler

with open("lab5/token.txt") as f:
    TOKEN = f.read()

import io
import sys
import os
sys.path.append(os.path.join(os.path.dirname(__file__), "..", "..", "pixelsort"))
from corruption2 import corrupt_image

async def start(update: Update, context: CallbackContext) -> None:
    await update.message.reply_text(
        "AWAITING TO RECEIVE AN IMAGE..."
    )

async def destroy_photo(update: Update, context: CallbackContext) -> None:
    try:
        photo_file = await update.message.photo[-1].get_file()
        photo_bytes = await photo_file.download_as_bytearray()
        context.user_data['photo_bytes'] = photo_bytes

        caption = update.message.caption
        if caption and 0 <= int(caption) <= 100:
            processed_bytes = corrupt_image(photo_bytes, tameness=int(caption))
        else:
            processed_bytes = corrupt_image(photo_bytes)

        keyboard = [[InlineKeyboardButton("IM NOT SATISFIED", callback_data='re-destroy')]]
        await update.message.reply_photo(
            photo=processed_bytes,
            caption="THANK YOU FOR YOUR CONTENT",
            reply_markup=InlineKeyboardMarkup(keyboard)
        )
    except Exception as e:
        print(e)
        await update.message.reply_text("SOMETHING WENT WRONG")

async def button_click(update: Update, context: CallbackContext):
    query = update.callback_query
    await query.answer()

    if query.data == 're-destroy':
        try:
            photo_bytes = context.user_data.get('photo_bytes')
            if not photo_bytes:
                await query.edit_message_text("NOTHING REMAINS")
                return

            processed_bytes = corrupt_image(photo_bytes)

            await query.edit_message_media(
                media=InputMediaPhoto(processed_bytes,
                                      caption="IF YOU WISH..."),
```

```
                reply_markup=InlineKeyboardMarkup([
                    [InlineKeyboardButton("IM NOT SATISFIED", callback_data='re-destroy')]
                ])
            )
        except Exception as e:
            print(e)
            await query.edit_message_text("NAH")

def main() -> None:
    application = Application.builder().token(TOKEN).build()

    application.add_handler(CommandHandler("start", start))
    application.add_handler(MessageHandler(filters.PHOTO, destroy_photo))
    application.add_handler(CallbackQueryHandler(button_click))
    application.add_handler(MessageHandler(
        filters.TEXT & ~filters.COMMAND,
        lambda update, context: update.message.reply_text("THIS IS NOT INCLUDED IN MY PROGRAMMING")
    ))

    print("Brace for impact")
    application.run_polling(allowed_updates=Update.ALL_TYPES)

if __name__ == '__main__':
    main()
```

Результат выполнения программы: