Московский государственный технический университет им. Н.Э. Баумана

Факультет ИУ
Кафедра ИУ5

Курс «Основы информатики»
Отчет по лабораторной работе №6

Выполнил студент группы ИУ5-33Б:
Хасанова К.М.
Подпись и дата:

Проверил преподаватель каф.:
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2024 г

Описание задания

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

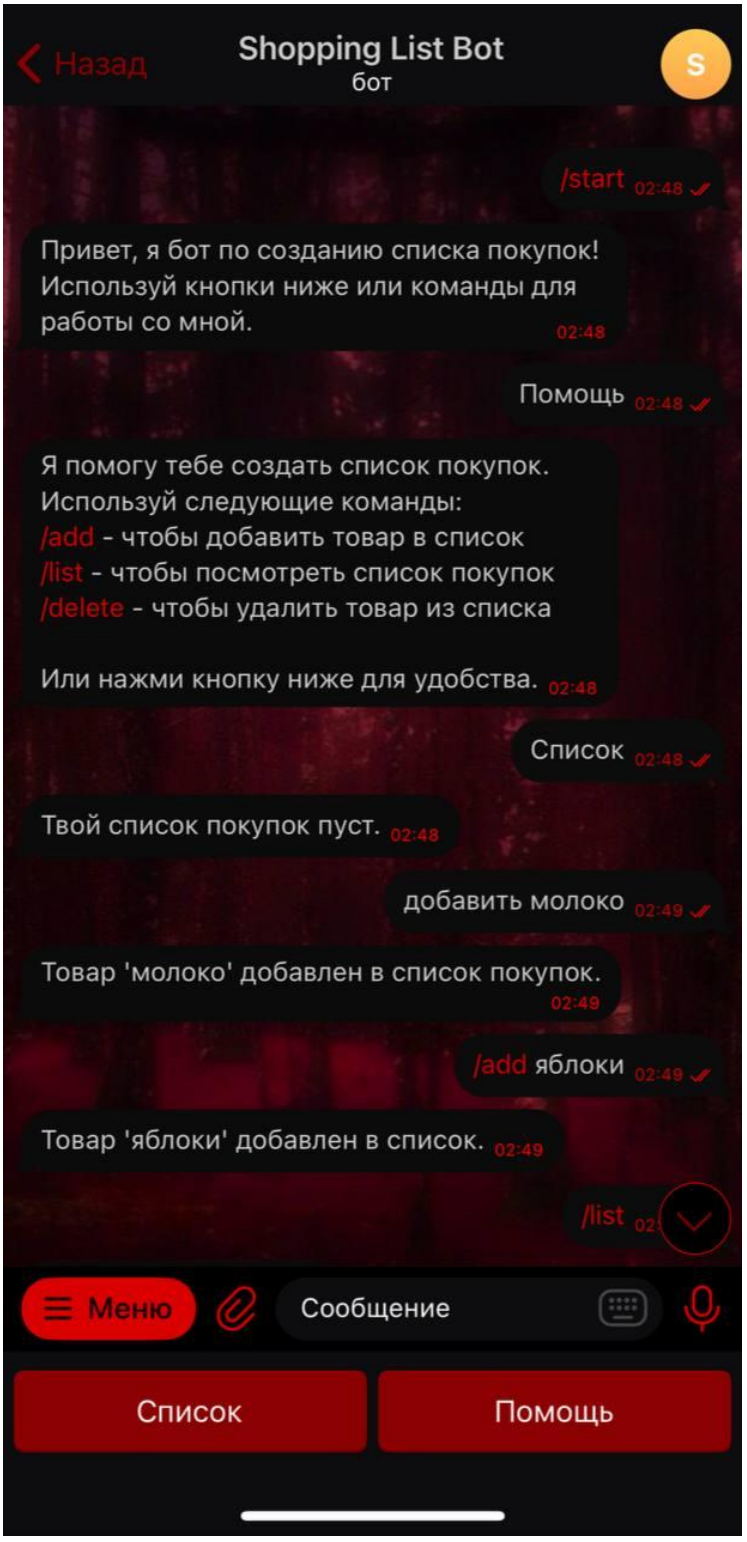
Текст программы

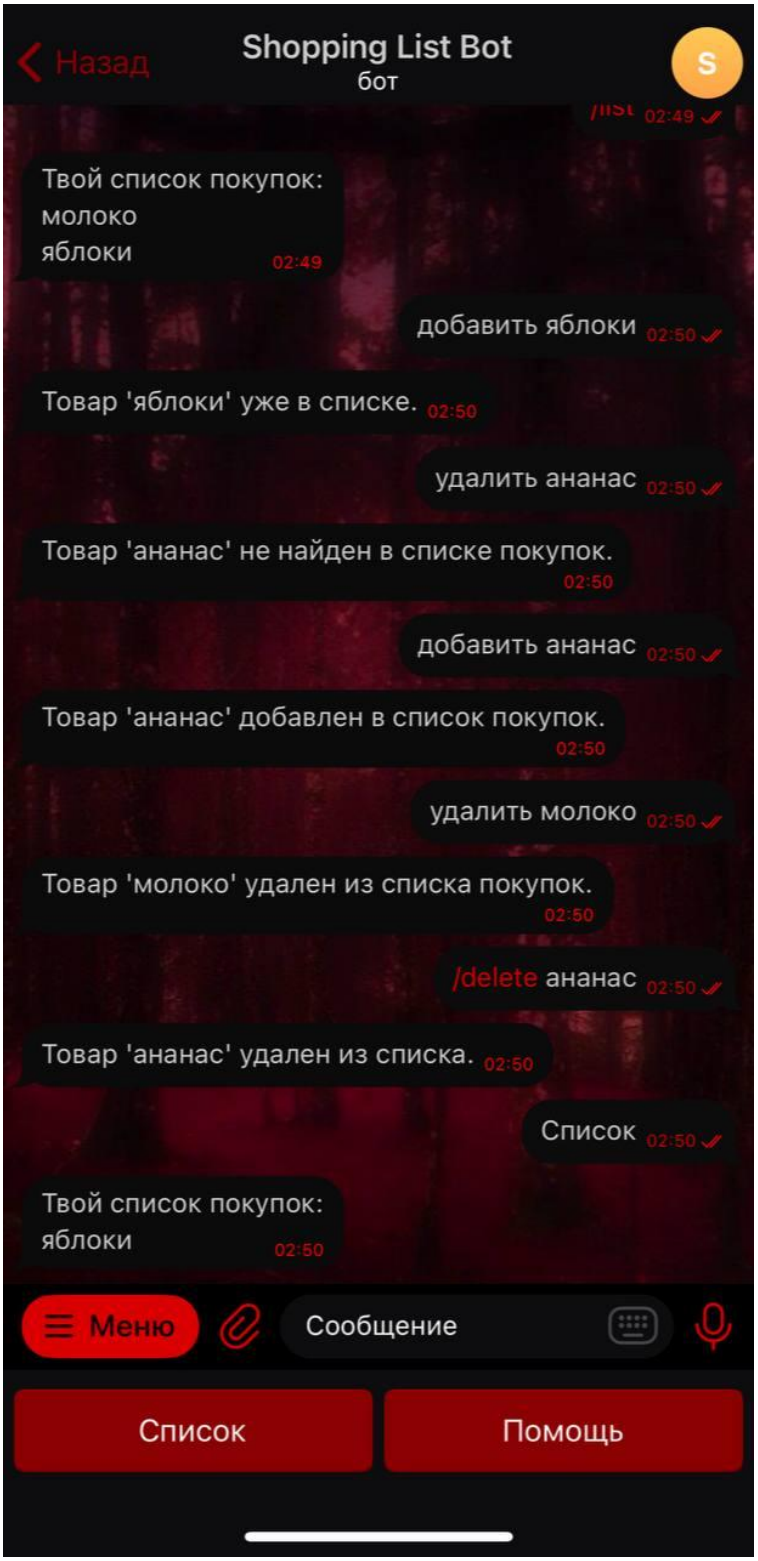
```
from typing import Final
from telegram import Update, ReplyKeyboardMarkup, KeyboardButton
from telegram.ext import Application, CommandHandler, MessageHandler, filters,
CallbackContext, ContextTypes

# Константы
TOKEN: Final = '8100881734:AAGrPktorJRVC8689RPiE_c71iXJWfBqxsA'
BOT_USERNAME: Final = '@ssshoppinglist_bot'

shopping_list = []


# Главное меню с кнопками
def main_menu():
    keyboard = [
        [KeyboardButton("Список"), KeyboardButton("Помощь")]  # Две кнопки: "Список" и
"Помощь"
    ]
    return ReplyKeyboardMarkup(keyboard, resize_keyboard=True)


# Команда /start
async def start_command(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await update.message.reply_text(
        'Привет, я бот по созданию списка покупок! Используй кнопки ниже или команды
для работы со мной.',
        reply_markup=main_menu()
    )


# Команда /help
async def help_command(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await update.message.reply text(
        'Я помогу тебе создать список покупок. Используй следующие команды:\n'
        '/add - чтобы добавить товар в список\n'
        '/list - чтобы посмотреть список покупок\n'
        '/delete - чтобы удалить товар из списка\n\n'
        'Или нажми кнопку ниже для удобства.',
        reply_markup=main_menu()
    )


# Команда /add - добавление товара в список
async def add_command(update: Update, context: CallbackContext) -> None:
    item = " ".join(context.args)
    if item:
        if item not in shopping_list:  # Проверяем, если товар уже в списке
            shopping_list.append(item)
            await update.message.reply_text(f"Товар '{item}' добавлен в список.")
        else:
            await update.message.reply_text(f"Товар '{item}' уже в списке.")
    else:
        await update.message.reply_text("Пожалуйста, укажи товар, который нужно
добавить.")


# Команда /list - показать список покупок
```

```
async def list_command(update: Update, context: CallbackContext) -> None:
    if shopping_list:
        items = "\n".join(shopping_list)
        await update.message.reply_text(f"Твой список покупок:\n{items}")
    else:
        await update.message.reply_text("Твой список покупок пуст.")


# Команда /delete - удаление товара из списка
async def delete_command(update: Update, context: CallbackContext) -> None:
    item = " ".join(context.args)
    if item:
        if item in shopping_list:
            shopping_list.remove(item)
            await update.message.reply_text(f"Товар '{item}' удален из списка.")
        else:
            await update.message.reply_text(f"Товар '{item}' не найден в списке.")
    else:
        await update.message.reply text("Пожалуйста, укажи товар, который нужно
удалить.")


# Обработчик общего сообщения с проверкой на добавление или удаление
async def handle message(update: Update, context: ContextTypes.DEFAULT TYPE):
    text = update.message.text.lower()

    if "добавить" in text:  # Если в сообщении есть слово "добавить"
        item = text.replace("добавить", "").strip()  # Убираем "добавить" и пробелы
        if item:  # Если после слова "добавить" что-то есть
            if item not in shopping_list:  # Проверяем, не добавлен ли товар уже
                shopping_list.append(item)
                await update.message.reply_text(f"Товар '{item}' добавлен в список
покупок.")
            else:
                await update.message.reply_text(f"Товар '{item}' уже в списке.")
        else:
            await update.message.reply_text("Пожалуйста, укажи товар, который нужно
добавить.")

    elif "удалить" in text:  # Если в сообщении есть слово "удалить"
        item = text.replace("удалить", "").strip()  # Убираем "удалить" и пробелы
        if item:  # Если после слова "удалить" что-то есть
            if item in shopping_list:
                shopping_list.remove(item)
                await update.message.reply_text(f"Товар '{item}' удален из списка
покупок.")
            else:
                await update.message.reply_text(f"Товар '{item}' не найден в списке
покупок.")
        else:
            await update.message.reply_text("Пожалуйста, укажи товар, который нужно
удалить.")

    elif text == "список":  # Обработка нажатия на кнопку "Список"
        if shopping_list:
            items = "\n".join(shopping_list)
            await update.message.reply_text(f"Твой список покупок:\n{items}",
reply_markup=main_menu())
        else:
            await update.message.reply_text("Твой список покупок пуст.",
reply_markup=main_menu())

    elif text == "помощь":  # Обработка нажатия на кнопку "Помощь"
        await help_command(update, context)

    elif "пасхалко" in text:  # Если в тексте есть слово "пасхалка"
        # Отправляем картинку
        await update.message.reply_photo(r"C:\Users\Администратор\Pictures\photo.jpg")
```

```
    elif "почему" in text:  # Если в тексте есть слово "почему"
        # Отправляем стикер
        await
update.message.reply_sticker("CAACAgIAAxkBAAELGFpnZynBO_KQoMTWP3LSOqSJ8YBT8wACt2MAAmEEO
UviHR61N3ecUDYE")

    else:
        await update.message.reply_text(
            'Если хочешь пополнить список или удалить что-то - напиши вначале
"добавить" или "удалить".',
            reply_markup=main_menu()
        )  # Ответ на любое другое сообщение


# Обработчик ошибок
async def error(update: Update, context: ContextTypes.DEFAULT_TYPE):
    print(f'Update {update} caused error {context.error}')


# Запуск приложения
if __name__ == '__main__':
    print('Starting bot...')

    # Создаем приложение для Telegram бота
    app = Application.builder().token(TOKEN).build()

    # Добавляем команды
    app.add_handler(CommandHandler('start', start_command))
    app.add_handler(CommandHandler('help', help_command))
    app.add_handler(CommandHandler("add", add_command))
    app.add_handler(CommandHandler("list", list_command))
    app.add_handler(CommandHandler("delete", delete_command))

    # Общий обработчик сообщений для текста
    app.add_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, handle_message))

    # Обработчик ошибок
    app.add_error_handler(error)

    # Запуск polling
    print('Polling...')
    app.run_polling(poll_interval=3)
```

Выполнение программы


Назад
Shopping List Bot
бот
S
/start 02:48
Привет, я бот по созданию списка покупок! Используй кнопки ниже или команды для работы со мной. 02:48
Помощь 02:48
Я помогу тебе создать список покупок. Используй следующие команды:
/add - чтобы добавить товар в список
/list - чтобы посмотреть список покупок
/delete - чтобы удалить товар из списка
Или нажми кнопку ниже для удобства. 02:48
Список 02:48
Твой список покупок пуст. 02:48
добавить молоко 02:49
Товар 'молоко' добавлен в список покупок. 02:49
/add яблоки 02:49
Товар 'яблоки' добавлен в список. 02:49
/list 02
Меню
Сообщение
Список
Помощь


Назад
Shopping List Bot
бот
S
/list 02:49
Твой список покупок:
молоко
яблоки
02:49
добавить яблоки 02:50
Товар 'яблоки' уже в списке. 02:50
удалить ананас 02:50
Товар 'ананас' не найден в списке покупок.
02:50
добавить ананас 02:50
Товар 'ананас' добавлен в список покупок.
02:50
удалить молоко 02:50
Товар 'молоко' удален из списка покупок.
02:50
/delete ананас 02:50
Товар 'ананас' удален из списка. 02:50
Список 02:50
Твой список покупок:
яблоки
02:50
Меню
Сообщение
Список
Помощь