**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс

«**Парадигмы и конструкции языков программирования**»

**Отчет по лабораторной работе №5**

**«**[**Разработка простого бота для Telegram с использованием языка Python**](https://github.com/ugapanyuk/BKIT_2021/wiki/lab_bot1)**»**

|  |  |  |
| --- | --- | --- |
| **Выполнил:** |  | **Проверил:** |
| студент группы ИУ5-31Б |  | преподаватель каф. ИУ5 |
| Санников Н.А. |  |  |
|  |  |  |

Москва, 2024 г.

**Описание задания**

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

**Текст программы**

**#bot\_v1.py**

from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup

from telegram.ext import Application, CommandHandler, CallbackQueryHandler, ContextTypes

TOKEN = "токен удалён в целях безопасности"

# Функция, которая вызывается, когда пользователь вводит команду /start

async def start(update: Update, context: ContextTypes.DEFAULT\_TYPE) -> None:

# создаем список с одной кнопкой

keyboard = [

[InlineKeyboardButton("Нажми меня!", callback\_data='button\_pressed')]

]

# оборачиваем кнопку в InlineKeyboardMarkup для отображения пользователю

reply\_markup = InlineKeyboardMarkup(keyboard)

# отправляем сообщение с кнопкой пользователю

await update.message.reply\_text('Привет! Это мой первый бот с кнопками.', reply\_markup=reply\_markup)

# функция, которая вызывается, когда пользователь нажимает на кнопку

async def button\_handler(update: Update, context: ContextTypes.DEFAULT\_TYPE) -> None:

# получаем инфу о нажатии на кнопку

query = update.callback\_query

# штука для тг, чтобы не было ошибок

await query.answer()

# меняем прошлое сообщение на нужный текст при нажатии на кнопку

await query.edit\_message\_text(text="Вы нажали на кнопку!")

def main() -> None:

# инициализация бота

application = Application.builder().token(TOKEN).build()

# Регистрируем обработчик команды /start

application.add\_handler(CommandHandler("start", start))

# Регистрируем обработчик нажатия на кнопку

application.add\_handler(CallbackQueryHandler(button\_handler))

# Запускаем бота и начинаем polling (циклично докапываться до тг, есть ли новые сообщения?)

application.run\_polling()

if \_\_name\_\_ == '\_\_main\_\_':

main()

**#bot\_v2.py**

from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup

from telegram.ext import Application, CommandHandler, CallbackQueryHandler, ContextTypes

TOKEN = "токен удалён в целях безопасности"

# Функция, которая вызывается, когда пользователь вводит команду /start

async def start(update: Update, context: ContextTypes.DEFAULT\_TYPE) -> None:

# создаем список с одной кнопкой 1\_1

keyboard = [

[InlineKeyboardButton("Нажми меня!", callback\_data='button\_pressed')]

]

# оборачиваем кнопку в InlineKeyboardMarkup для отображения пользователю

reply\_markup = InlineKeyboardMarkup(keyboard)

# отправляем сообщение с кнопкой 1\_1 пользователю

await update.message.reply\_text('Привет! Это мой первый бот с кнопками.', reply\_markup=reply\_markup)

# функция, которая вызывается, когда пользователь нажимает на кнопку 1\_1

async def button\_handler(update: Update, context: ContextTypes.DEFAULT\_TYPE) -> None:

# получаем инфу о нажатии на кнопку 1\_1

query = update.callback\_query

# штука для тг, чтобы не было ошибок

await query.answer()

# создаем новый список кнопок

new\_keyboard = [

[InlineKeyboardButton("Новая кнопка 1", callback\_data='new\_button\_1')],

[InlineKeyboardButton("Новая кнопка 2", callback\_data='new\_button\_2')]

]

# оборачиваем новый список в InlineKeyboardMarkup

reply\_markup = InlineKeyboardMarkup(new\_keyboard)

# заменяем сообщение на новое с другим списком кнопок

await query.edit\_message\_text(text="Выберите новую кнопку:", reply\_markup=reply\_markup)

# функция для обработки нажатия на кнопку 2\_1

async def new\_button\_1\_handler(update: Update, context: ContextTypes.DEFAULT\_TYPE) -> None:

query = update.callback\_query

await query.answer()

await query.edit\_message\_text(text="Вы нажали на Новую кнопку 1!")

# функция для обработки нажатия на кнопку 2\_2

async def new\_button\_2\_handler(update: Update, context: ContextTypes.DEFAULT\_TYPE) -> None:

query = update.callback\_query

await query.answer()

await query.edit\_message\_text(text="Вы нажали на Новую кнопку 2!")

def main() -> None:

# инициализация бота

application = Application.builder().token(TOKEN).build()

# Регистрируем обработчик команды /start

application.add\_handler(CommandHandler("start", start))

# Регистрируем обработчик нажатия на кнопку 1\_1

application.add\_handler(CallbackQueryHandler(button\_handler, pattern='button\_pressed'))

# Регистрируем обработчик нажатия на кнопку 2\_1

application.add\_handler(CallbackQueryHandler(new\_button\_1\_handler, pattern='new\_button\_1'))

# Регистрируем обработчик нажатия на кнопку 2\_2

application.add\_handler(CallbackQueryHandler(new\_button\_2\_handler, pattern='new\_button\_2'))

# Запускаем бота и начинаем polling (циклично докапываться до тг, есть ли новые сообщения?)

application.run\_polling()

if \_\_name\_\_ == '\_\_main\_\_':

main()

from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup

**Экранные формы**

**с примерами выполнения программы**

**работа бота в режиме 1**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

\*\*нажатие на кнопку изменяет сообщение на следующее состояние, например здесь выведется информация, что кнопка была нажата:

Изображение выглядит как текст, снимок экрана, Шрифт

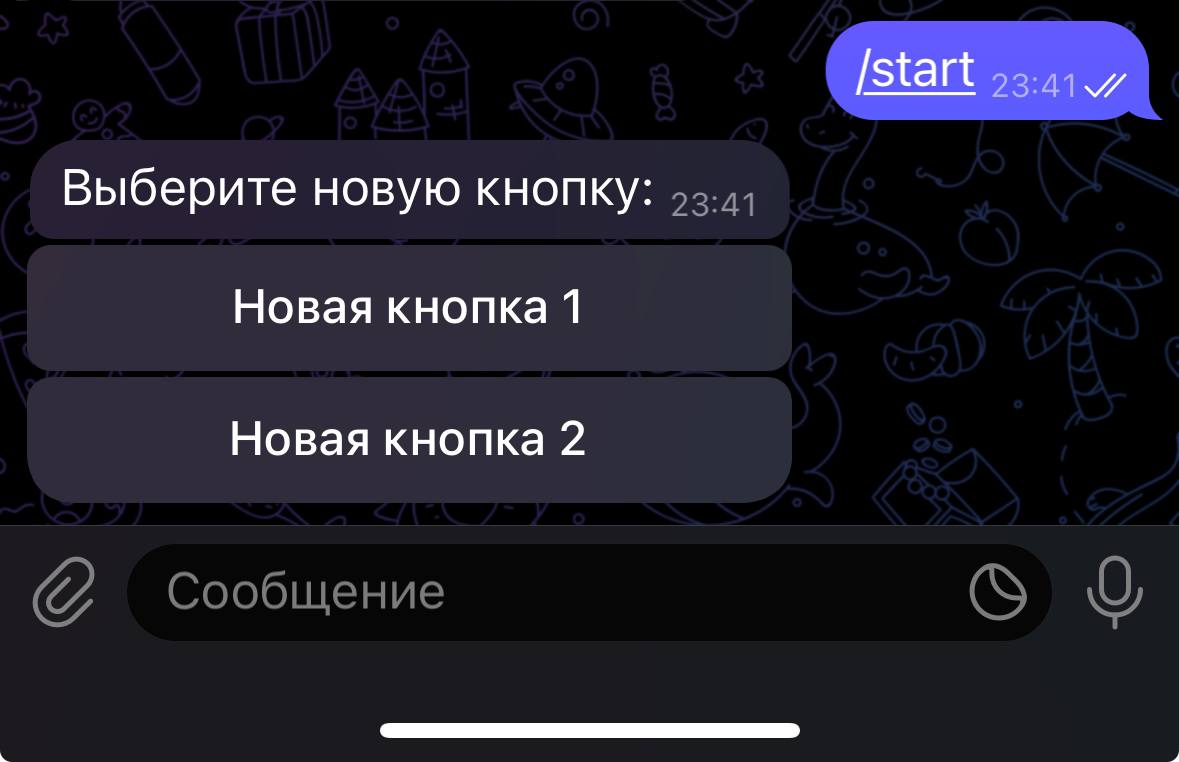
Автоматически созданное описание

**работа бота в режиме 2**

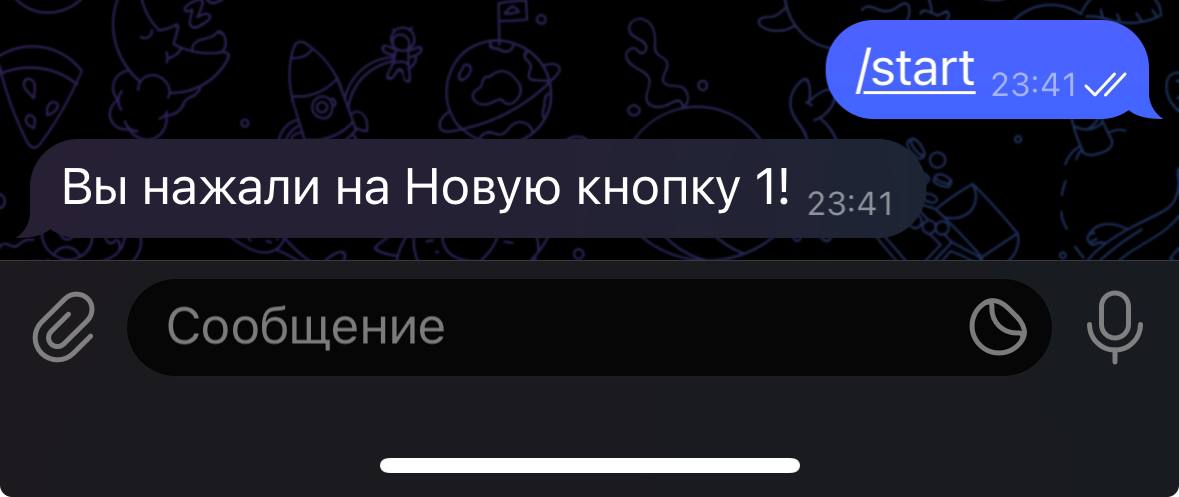
Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

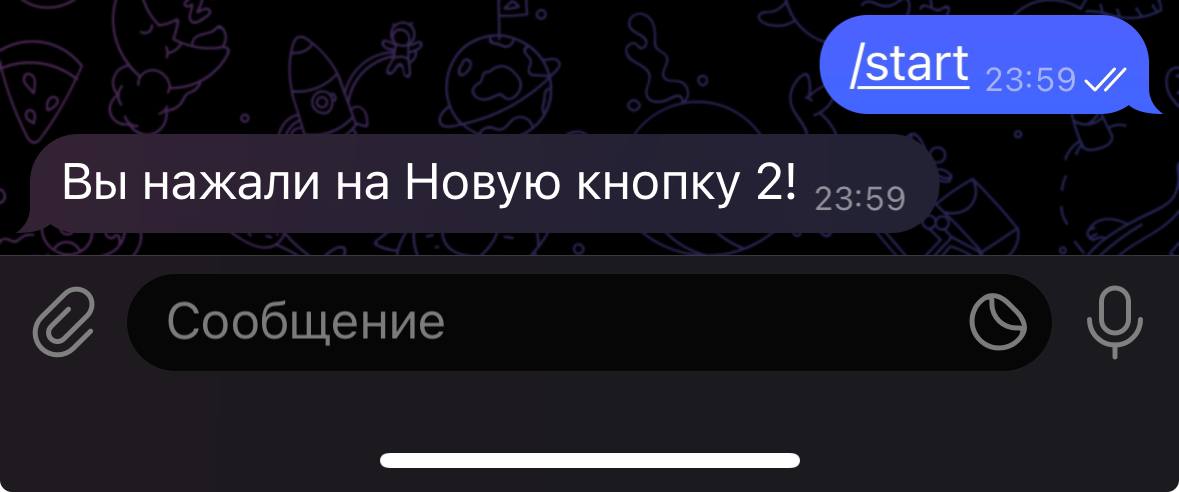
\*\*нажатие на кнопку изменяет сообщение на следующие две кнопки:



\*\*нажатие на любую кнопку изменяет сообщение на следующее состояние, например если нажать на первую кнопку, выведется информация, что новая кнопка 1 была нажата:



\*\*аналогично и для второй кнопки выведется информация о нажатии новой кнопки 2:



**Заключение**

В заключение хочу уточнить, что бот, показанный выше, не имеет практической пользы на данном этапе разработки, а служит лишь «фундаментом», на котором можно реализовать необходимую функциональность.

В ходе выполнения данной лабораторной работы были приобретены базовые навыки разработки Telegram-ботов с использованием языка Python и библиотеки python-telegram-bot. Бот был успешно реализован с использованием кнопок, что позволяет пользователю взаимодействовать с ним через интерфейс. Также был продемонстрирован механизм обработки нажатий на кнопки и изменения сообщений в ответ на действия пользователя.

Разработка подобных ботов может быть полезна для создания различных автоматизированных решений, таких как опросы, навигация по информации, а также для интеграции с внешними системами. Полученные знания могут служить основой для создания более сложных ботов, предоставляющих дополнительные возможности и удобные интерфейсы для взаимодействия с пользователем.