

**Московский государственный технический
университет им. Н.Э. Баумана**

**Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»**

Курс «ПиКЯП»

Отчет по лабораторной работе №6

«Разработка телеграмм бота на основе конечного автомата»

Выполнил:

студент группы ИУ5-36Б

Илюхин И. Д.

Подпись и дата: 18.12.2024

Проверил:

преподаватель каф. ИУ5

Нардид А. Н.

Подпись и дата:

Москва, 2024 г.

Цель лабораторной работы: изучение разработки ботов в Telegram.

Задание:

Разработайте бота для Telegram. Бот должен реализовывать конечный автомат из трех состояний.

Код программы:

```
from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup
from telegram.ext import (
    Application, CommandHandler, CallbackQueryHandler, ContextTypes,
    ConversationHandler
)

TOKEN = "7991963753:AAGo8ts_7wmrtfitWhVd4aTKwh992bRCXjY" # Замените на ваш токен
STATE1, STATE2, STATE3 = range(3)

async def start(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    context.user_data['button_history'] = [] # Инициализируем историю
    keyboard = [
        [
            InlineKeyboardButton("лада веста", callback_data="state2_button1"),
            InlineKeyboardButton("лада гранта", callback_data="state2_button2"),
            InlineKeyboardButton("лада нива", callback_data="state2_button3")
        ]
    ]
    reply_markup = InlineKeyboardMarkup(keyboard)
    await update.message.reply_text(
        "Выберете автомобиль:", reply_markup=reply_markup
    )
    return STATE1

async def state1_handler(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    query = update.callback_query
    await query.answer()
    context.user_data['button_history'].append(query.data) # Добавляем нажатую кнопку
    в историю

    keyboard = [
        [
            InlineKeyboardButton("стандарт", callback_data="state3_button1"),
            InlineKeyboardButton("комфорт", callback_data="state3_button2"),
            InlineKeyboardButton("люкс", callback_data="state3_button3")
        ]
    ]
    reply_markup = InlineKeyboardMarkup(keyboard)
    await query.edit_message_text(
```

```

        "выберете комплектацию:", reply_markup=reply_markup
    )
    return STATE2

async def state2_handler(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    query = update.callback_query
    await query.answer()
    context.user_data['button_history'].append(query.data) # Добавляем нажатую кнопку
    в историю
    keyboard = [
        [
            InlineKeyboardButton("резина", callback_data="end_button1"),
            InlineKeyboardButton("коврики", callback_data="end_button2"),
            InlineKeyboardButton("скидка", callback_data="end_button3")
        ]
    ]
    reply_markup = InlineKeyboardMarkup(keyboard)
    await query.edit_message_text(
        "выберете дополнительные услуги", reply_markup=reply_markup
    )
    return STATE3

async def state3_handler(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    query = update.callback_query
    await query.answer()
    context.user_data['button_history'].append(query.data) # Добавляем нажатую кнопку
    в историю
    if query.data in ["end_button1", "end_button2", "end_button3"]:
        history_text = "История нажатых кнопок:\n"
        history_text += "\n".join(context.user_data['button_history'])
        await query.edit_message_text(history_text)
        return ConversationHandler.END
    return ConversationHandler.END

async def cancel(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    await update.message.reply_text("Диалог отменен. Введите /start, чтобы начать заново.")
    return ConversationHandler.END

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

    conv_handler = ConversationHandler(
        entry_points=[CommandHandler("start", start)],
        states={
            STATE1: [CallbackQueryHandler(state1_handler)],
            STATE2: [CallbackQueryHandler(state2_handler)],
            STATE3: [CallbackQueryHandler(state3_handler)],
        },
        fallbacks=[CommandHandler("cancel", cancel)],
    )

```

```

)


application.add_handler(conv_handler)
application.run_polling()

if __name__ == "__main__":
    main()

```

Пример работы:


Сегодня



Илья

/start

✓ 02:44



lab 6 python


Выберете автомобиль:

02:44

лада веста


лада гранта

лада нива



Илья

/start




lab 6 python

выберете комплектацию:

стандарт


комфорт

люкс



Илья

/start




lab 6 python

выберете дополнительные услуги

резина


коврики

скидка



Илья

/start



lab 6 python

История нажатых кнопок:

state2_button1

state3_button3

end_button1