Отчет

Лабораторная работа 6-7

ПКИЯП

Вариант 23

Студент: Цугель А.С.

Группа: ИБМ3-34Б

Преподаватель: Гапанюк Ю.

Задание:

1. Разработайте бота для Telegram. Бот должен представлять собой конечный автомат с тремя состояниями.

Пример кода:

```
import logging
from telegram import Update
from telegram.ext import (
    ApplicationBuilder,
    CommandHandler,
    ContextTypes,
)

logging.basicConfig(
    format="%(asctime)s - %(name)s - %(levelname)s - %(message)s",
    level=logging.INFO,
)

# Состояния
STATE_A = "A"
STATE_B = "B"
STATE_C = "C"

STATE_KEY = "fsm_state"  # ключ для хранения состояния в user_data


async def start(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """Старт: устанавливаем начальное состояние автомата."""
    context.user_data[STATE_KEY] = STATE_A
    await update.message.reply_text(
        "Привет! Я бот-автомат с тремя состояниями.\n"
        "Текущее состояние: A\n\n"
        "Команды:\n"
        "/next — переход вперёд\n"
        "/back — переход назад\n"
        "/reset — сброс в состояние A\n"
        "/state — показать текущее состояние"
    )


def get_state(context: ContextTypes.DEFAULT_TYPE) -> str:
    """Получить текущее состояние пользователя, по умолчанию A."""
    return context.user_data.get(STATE_KEY, STATE_A)
```

```python
def set_state(context: ContextTypes.DEFAULT_TYPE, state: str):
    """Установить состояние пользователя."""
    context.user_data[STATE_KEY] = state


async def state_cmd(update: Update, context:
ContextTypes.DEFAULT_TYPE):
    """Показать текущее состояние."""
    s = get_state(context)
    await update.message.reply_text(f"Текущее состояние: {s}")


async def next_cmd(update: Update, context:
ContextTypes.DEFAULT_TYPE):
    """Переход вперёд по автомату."""
    s = get_state(context)

    if s == STATE_A:
        set_state(context, STATE_B)
        await update.message.reply_text("Переход: A → B")
    elif s == STATE_B:
        set_state(context, STATE_C)
        await update.message.reply_text("Переход: B → C")
    elif s == STATE_C:
        await update.message.reply_text("Из состояния C перехода /next
нет.")
    else:
        await update.message.reply_text("Неизвестное состояние, делаю
сброс в A.")
        set_state(context, STATE_A)


async def back_cmd(update: Update, context:
ContextTypes.DEFAULT_TYPE):
    """Переход назад по автомату."""
    s = get_state(context)

    if s == STATE_A:
        await update.message.reply_text("Из состояния A перехода /back
нет.")
    elif s == STATE_B:
        set_state(context, STATE_A)
        await update.message.reply_text("Переход: B → A")
    elif s == STATE_C:
        set_state(context, STATE_B)
```

```
        await update.message.reply_text("Переход: C → B")
    else:
        await update.message.reply_text("Неизвестное состояние, делаю
сброс в A.")
        set_state(context, STATE_A)


async def reset_cmd(update: Update, context:
ContextTypes.DEFAULT_TYPE):
    """Сброс автомата в состояние A."""
    set_state(context, STATE_A)
    await update.message.reply_text("Состояние сброшено к A.")


async def help_cmd(update: Update, context:
ContextTypes.DEFAULT_TYPE):
    """Подсказка по командам."""
    await update.message.reply_text(
        "Я демонстрация конечного автомата из 3 состояний.\n"
        "Команды:\n"
        "/start — начать и установить состояние A\n"
        "/state — показать текущее состояние\n"
        "/next — переход вперёд (A→B, B→C)\n"
        "/back — переход назад (C→B, B→A)\n"
        "/reset — сброс в состояние A"
    )


def main():
    # ТВОЙ ТОКЕН БОТА:
    token = "8213423467:AAELwjjivOu0Hmh4ebnp5-iIt_zTb8buGY0"

    app = ApplicationBuilder().token(token).build()

    app.add_handler(CommandHandler("start", start))
    app.add_handler(CommandHandler("help", help_cmd))
    app.add_handler(CommandHandler("state", state_cmd))
    app.add_handler(CommandHandler("next", next_cmd))
    app.add_handler(CommandHandler("back", back_cmd))
    app.add_handler(CommandHandler("reset", reset_cmd))

    app.run_polling()


if __name__ == "__main__":
```

```
main()
```

Пример выполнения программы: