Защищено:
Гапанюк Ю.Е.

"__"_____**2023** г.

Демонстрация:
Фень Н.Т.

"__"_____**2023** г.

Отчет по лабораторной работе № 6 по курсу Парадигмы и конструкции языков программирования

Тема работы: " Разработка бота на основе конечного автомата для Telegram с использованием языка Python. "

4
(количество листов)
Вариант № **1**

ИСПОЛНИТЕЛЬ:

студент группы ИУ5-52Б _____
(подпись)

Фень Н.Т.

"__"_____**2023** г.

Москва, МГТУ - **2023**

СОДЕРЖАНИЕ

1. Описание задания.....4
2. Текст программы.....4
3. Экранные формы с примерами выполнения программы**Error! Bookmark not defined.**

1. Описание задания

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

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

2. Текст программы

```
try:
    import logging
    from aiogram import Bot, Dispatcher, types
    from aiogram.types import InlineKeyboardButton, InlineKeyboardMarkup
    from aiogram.utils import executor
    from aiogram.dispatcher import FSMContext
    from aiogram.dispatcher.filters.state import State, StatesGroup
    from aiogram.contrib.fsm_storage.memory import MemoryStorage
    from aiogram.contrib.middlewares.logging import LoggingMiddleware
except ImportError:
    pass  # Ignore import error

from bot_token import token

#Базовое логгирование
logging.basicConfig(level=logging.INFO)


bot = Bot(token=token)
dp = Dispatcher(bot,storage=MemoryStorage())
dp.middleware.setup(LoggingMiddleware())
class MyStates(StatesGroup):
    state_start = State()
    state_middle = State()
    state_end = State()

@dp.message_handler(commands=['go'], state="*")
async def cmd_go(message: types.Message, state: FSMContext):
    await MyStates.state_start.set()
    await message.answer("Вы находитесь в состоянии начала. Введите /next для
перехода к следующему состоянию.")

@dp.message_handler(commands=['next'], state=MyStates.state_start)
async def cmd_next(message: types.Message):
    await MyStates.state_middle.set()
    await message.answer("Вы находитесь в среднем состоянии. Введите /end для
завершения.")

@dp.message_handler(commands=['end'], state=MyStates.state_middle)
async def cmd_end(message: types.Message):
    await MyStates.state_end.set()
    await message.answer("Вы находитесь в конечном состоянии. Введите /start для
начала заново.")

@dp.message_handler(commands=['start'], state="*")
async def cmd_start(message: types.Message):
    user = message.from_user
    keyboard = get_keyboard()
    await message.answer(f"Привет, {user.first_name}! Я простой бот с кнопками. Это
ответ на /start", reply_markup=keyboard)


@dp.message_handler(commands=['help'],state="*")
async def cmd_help(message: types.Message):
    await message.answer("Для проверки состояний введите /state\nДля проверки смены
```

```
состояний введите /go")

@dp.message_handler(commands=['state'],state="*")
async def cmd_help(message: types.Message):
    current_state = await dp.current_state(user=message.from_user.id).get_state()
    await message.answer(str(current_state))

# Handle text
@dp.message_handler(lambda message: message.text and not
message.text.startswith('/'))
async def handle_text(message: types.Message):
    await message.answer("Я понимаю только команды. Попробуйте использовать
/start.")


# Handle Нажатие на кнопки в сообщении /start
@dp.callback_query_handler(lambda c: True)
async def button_click(callback_query: types.CallbackQuery):
    if callback_query.data == 'button1':
        await callback_query.answer("Вы нажали на Кнопку 1!")
    elif callback_query.data == 'button2':
        await bot.send_message(callback_query.from_user.id, "Вы нажали кнопку 2")


# Генерация двух кнопок
def get_keyboard():
    keyboard = InlineKeyboardMarkup(row_width=2)
    buttons = [
        InlineKeyboardButton("Кнопка 1", callback_data='button1'),
        InlineKeyboardButton("Кнопка 2", callback_data='button2'),
    ]
    keyboard.add(*buttons)
    return keyboard


def main():
    # Запуск бота
    executor.start_polling(dp, skip_updates=True)


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