|  |  |  |
| --- | --- | --- |
| 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 3](#_Toc148091332)

[2. Текст программы 3](#_Toc148091333)

[3. Экранные формы с примерами выполнения программы 4](#_Toc148091334)

# 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.

# 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()