

Защищено:  
Гапанюк Ю.Е.

Демонстрация:  
Падалко К.Р.

"\_\_" \_\_\_\_\_ 2023 г.

"\_\_" \_\_\_\_\_ 2023 г.

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

**Тема работы: "Разработка простого бота для Telegram с  
использованием языка Python. "**

4

(количество листов)

Вариант № 23

ИСПОЛНИТЕЛЬ:

студент группы ИУ5Ц-  
54Б

Падалко К.Р.

\_\_\_\_\_  
(подпись)

"\_\_" \_\_\_\_\_ 2023 г.

## Содержание

1. Описание задания.....	3
2. Текст программы.....	3
3. Экранные формы с примерами выполнения программы.....	4

## 1. Описание задания

1. Разработайте простого бота для Telegram. Бот должен использовать функциональность создания кнопок.
2. Разработайте бота для Telegram. Бот должен реализовывать конечный автомат из трех состояний.

## 2. Текст программы

Файл «main.py»

```
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

from bot_t import token
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("Начало. Введите /next.")

@dp.message_handler(commands=['next'], state=MyStates.state_start)
async def cmd_next(message: types.Message):
    await MyStates.state_middle.set()
    await message.answer("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("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}! 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("Попробуйте использовать /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()

```