Лабораторная работа №5-6

Парадигмы и конструкции языков программирования

Барабаш Диана

ИБМ3-34Б

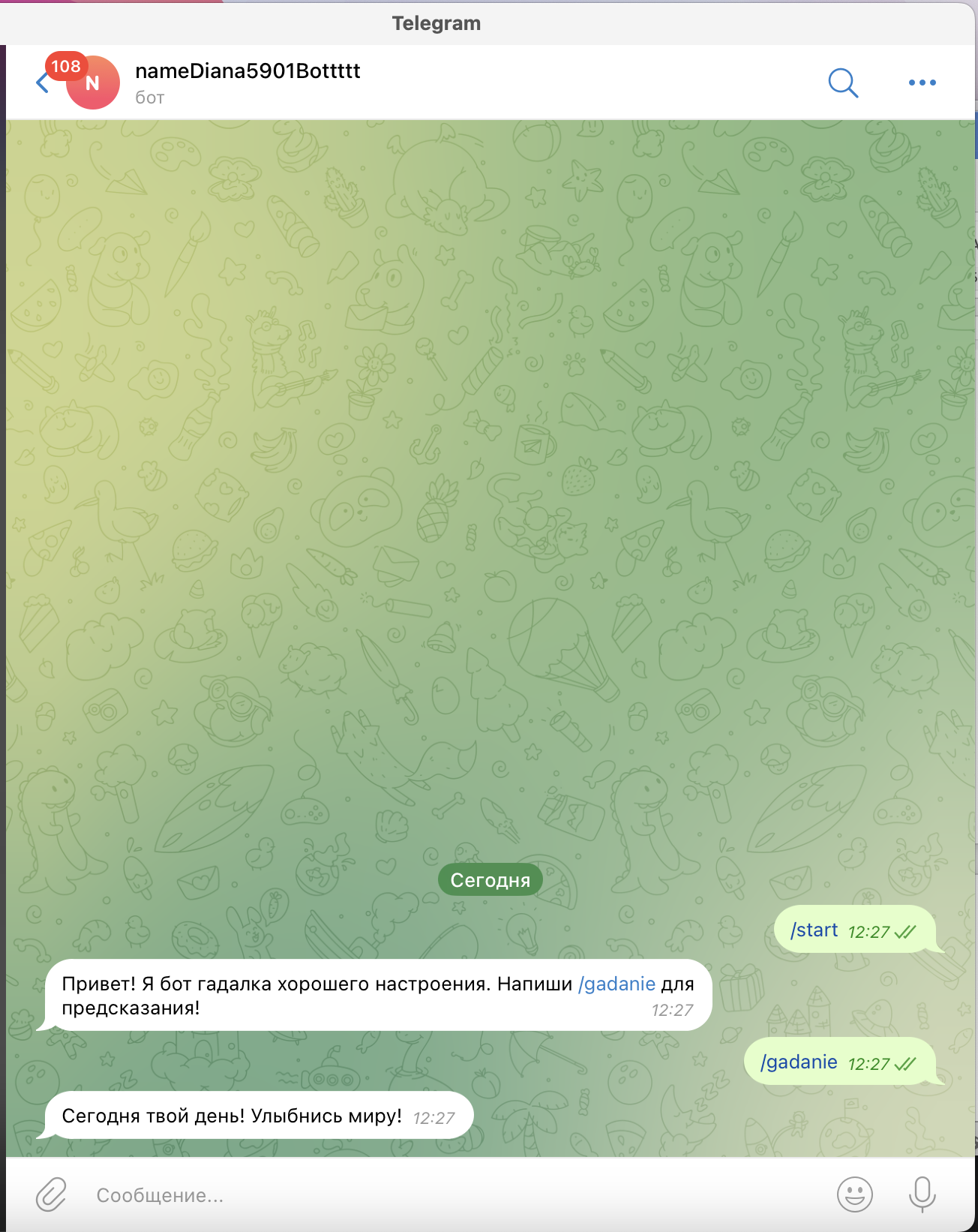
**Задание 1**

1. Разработайте простого бота для Telegram.

**Код программы:**

from telegram.ext import Application, CommandHandler  
  
# Вставьте ваш токен сюда  
TOKEN = "7566643105:AAFz2Z-nSFSYD7A5-rk94UN4WYqVURtVCS4"  
  
# Команда /start  
async def start(update, context):  
 await update.message.reply\_text("Привет! Я бот гадалка хорошего настроения. Напиши /gadanie для предсказания!")  
  
# Команда /gadanie  
async def gadanie(update, context):  
 predictions = [  
 "Сегодня твой день! Улыбнись миру!",  
 "Удача на твоей стороне!",  
 "Ожидай приятный сюрприз!",  
 "Ты сильнее, чем думаешь!",  
 "Скоро сбудется твоя мечта!"  
 ]  
 from random import choice  
 prediction = choice(predictions)  
 await update.message.reply\_text(prediction)  
  
# Основная функция  
def main():  
 # Создаем приложение  
 application = Application.builder().token(TOKEN).build()  
  
 # Регистрируем команды  
 application.add\_handler(CommandHandler("start", start))  
 application.add\_handler(CommandHandler("gadanie", gadanie))  
  
 # Запускаем бота  
 application.run\_polling()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Работа бота:**

****

**Задание 2:**

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

**Код программы:**

from telegram import ReplyKeyboardMarkup, ReplyKeyboardRemove, InlineKeyboardMarkup, InlineKeyboardButton  
from telegram.ext import Application, CommandHandler, MessageHandler, filters, ConversationHandler  
  
# Вставьте ваш токен сюда  
TOKEN = "7834194992:AAEE9ciU0ItiY-rKXu\_9PakXElFwtjn38IM"  
  
# Этапы диалога  
CHOOSING = 1  
  
# Словарь предсказаний  
predictions = {  
 "любовь": [  
 "Скоро ты встретишь свою вторую половинку!",  
 "Любовь витает в воздухе – будь готов!",  
 "Старый друг может стать чем-то большим...",  
 "Ваша любовь станет сильнее, чем когда-либо."  
 ],  
 "работа": [  
 "Успех в карьере уже на подходе!",  
 "Новая возможность постучится в твою дверь.",  
 "Тебя ждет повышение – прояви инициативу!",  
 "Будь уверен в себе, и работа принесет плоды."  
 ],  
 "здоровье": [  
 "Здоровье крепкое – можно сворачивать горы!",  
 "Не забывай отдыхать, чтобы не выгореть.",  
 "Спорт и правильное питание – твои ключи к успеху.",  
 "Тело отблагодарит тебя за заботу о нем."  
 ]  
}  
  
# Команда /start  
async def start(update, context):  
 await update.message.reply\_text(  
 "Привет! Я бот гадалка. Я могу рассказать тебе что-то о твоем будущем.\n"  
 "Напиши /gadanie, чтобы получить предсказание!"  
 )  
  
# Команда /gadanie  
async def gadanie(update, context):  
 # Клавиатура с вариантами  
 reply\_keyboard = [["Любовь", "Работа", "Здоровье"]]  
  
 await update.message.reply\_text(  
 "О чем ты хочешь узнать? Выбери категорию:",  
 reply\_markup=ReplyKeyboardMarkup(  
 reply\_keyboard, one\_time\_keyboard=True, resize\_keyboard=True  
 ),  
 )  
 return CHOOSING  
  
# Обработчик выбора категории  
async def choose\_category(update, context):  
 category = update.message.text.lower()  
  
 if category in predictions:  
 from random import choice  
 prediction = choice(predictions[category])  
 await update.message.reply\_text(  
 f"Твое предсказание о {category}:\n\n{prediction}",  
 reply\_markup=ReplyKeyboardRemove(), # Убираем клавиатуру после выбора  
 )  
 else:  
 await update.message.reply\_text(  
 "Прости, я не понимаю эту категорию. Попробуй снова: Любовь, Работа или Здоровье."  
 )  
 return CHOOSING # Оставляем в состоянии CHOOSING, чтобы пользователь мог выбрать снова  
  
 # Завершаем диалог  
 await update.message.reply\_text(  
 "Хочешь еще предсказание? Напиши /gadanie!", reply\_markup=ReplyKeyboardRemove()  
 )  
 return ConversationHandler.END  
  
# Команда /cancel для выхода из диалога  
async def cancel(update, context):  
 await update.message.reply\_text(  
 "Гадание отменено. Напиши /gadanie, если захочешь попробовать снова!",  
 reply\_markup=ReplyKeyboardRemove(),  
 )  
 return ConversationHandler.END  
  
# Основная функция  
def main():  
 # Создаем приложение  
 application = Application.builder().token(TOKEN).build()  
  
 # Настройка ConversationHandler для управления диалогами  
 conv\_handler = ConversationHandler(  
 entry\_points=[CommandHandler("gadanie", gadanie)],  
 states={  
 CHOOSING: [MessageHandler(filters.TEXT & ~filters.COMMAND, choose\_category)],  
 },  
 fallbacks=[CommandHandler("cancel", cancel)],  
 )  
  
 # Регистрируем команды  
 application.add\_handler(CommandHandler("start", start))  
 application.add\_handler(conv\_handler)  
  
 # Запускаем бота  
 application.run\_polling()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Работа Бота:**

****