**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции

языков программирования»

Отчет по лабораторной работе №3

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-33Б |  | преподаватель каф. ИУ5 |
| Нестерова Екатерина |  | Гапанюк Ю.Е. |
|  |  |  |

Москва, 2024 г.

Постановка задачи:

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

Код:

import telebot  
import random  
  
bot = telebot.TeleBot("7821045113:AAGJNlwg8wFj1wtb9TPTfBPPRR7nQv7GsHY")  
user\_data = {}  
states={  
 'start':0,  
 'min':1,  
 'max':2  
  
}  
  
def generate\_markup():  
 markup = telebot.types.ReplyKeyboardMarkup(resize\_keyboard=True)  
 btn1 = telebot.types.KeyboardButton('Начало диапазона')  
 btn2 = telebot.types.KeyboardButton('Конец диапазона')  
 btn3 = telebot.types.KeyboardButton('Генерировать число')  
 markup.add(btn1, btn2, btn3)  
 return markup  
  
  
@bot.message\_handler(commands=['start'])  
def start\_message(message):  
 user\_data[message.chat.id] = {'min': None, 'max': None,'state':states['start']} # Инициализируем данные пользователя  
 bot.send\_message(message.chat.id, "Привет! Давай сгенерируем число в диапазоне", reply\_markup=generate\_markup())  
  
  
@bot.message\_handler(func=lambda message: message.text == 'Начало диапазона')  
def get\_min\_value(message):  
 user\_data[message.chat.id]['state'] = states['min']  
 bot.send\_message(message.chat.id, "Введите начальное значение диапазона:")  
 # Сохраняем состояние: ввод начального значения  
  
  
@bot.message\_handler(func=lambda message: message.text == 'Конец диапазона')  
def get\_max\_value(message):  
 user\_data[message.chat.id]['state'] = states['max'] # Сохраняем состояние: ввод конечного значения  
  
 bot.send\_message(message.chat.id, "Введите конечное значение диапазона:")  
  
  
@bot.message\_handler(func=lambda message: message.text == 'Генерировать число')  
def generate\_number(message):  
 if user\_data[message.chat.id]['min'] is None or user\_data[message.chat.id]['max'] is None:  
 bot.send\_message(message.chat.id, "Сначала укажите диапазон!")  
 return  
  
 min\_value = int(user\_data[message.chat.id]['min'])  
 max\_value = int(user\_data[message.chat.id]['max'])  
  
 if min\_value > max\_value:  
 bot.send\_message(message.chat.id, "Минимальное значение должно быть меньше максимального")  
 return  
  
 random\_number = random.randint(min\_value, max\_value)  
 bot.send\_message(message.chat.id, f"Сгенерированное число: {random\_number}")  
 user\_data[message.chat.id]['state'] = states['start']  
  
@bot.message\_handler(func=lambda message: True)  
def process\_input(message):  
 chat\_id = message.chat.id  
 state = user\_data[chat\_id]['state']  
  
 if state == states['min']:  
 try:  
 user\_data[chat\_id]['min'] = int(message.text)  
  
 user\_data[chat\_id]['state'] = states['start'] # Сбрасываем состояние после ввода  
 bot.send\_message(chat\_id, "Минимальное значение установлено. Используйте кнопки для дальнейших действий", reply\_markup=generate\_markup())  
 except ValueError:  
 bot.send\_message(chat\_id, "Введите корректное число!")  
 elif state == states['max']:  
 try:  
 user\_data[chat\_id]['max'] = int(message.text)  
 user\_data[chat\_id]['state'] = states['start'] # Сбрасываем состояние после ввода  
 bot.send\_message(chat\_id, "Максимальное значение установлено. Используйте кнопки для дальнейших действий",reply\_markup=generate\_markup())  
 except ValueError:  
 bot.send\_message(chat\_id, "Введите корректное число!")  
  
  
bot.polling()



