**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

# Курс «Базовые компоненты интернет-технологий**»**

Отчет по лабораторным работам №5-6

«Основные конструкции языка Python»

| Выполнил: |  | Проверил: |
| --- | --- | --- |
| студент группы ИУ5-32Б |  | преподаватель каф. ИУ5 |
| Ткаченко В. Л. |  | Гапонюк Ю. Е. |
| Подпись и дата |  | Подпись и дата |

Москва, 2021 г.

### **Задание:**

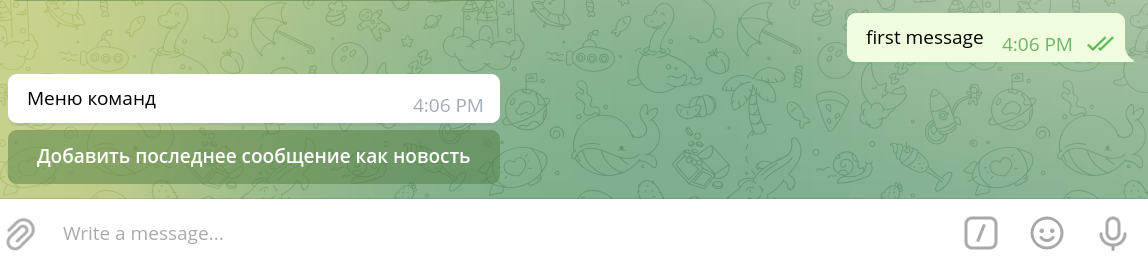
1. Разработайте бота для Telegram. Бот должен реализовывать конечный автомат из трех состояний.

**Листинг программы:**

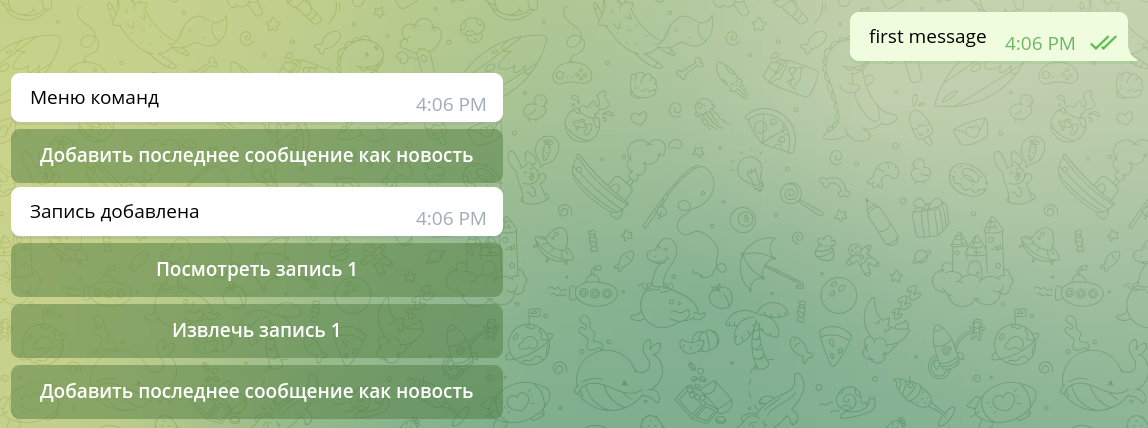
| *# buttons\_manager.py*  from telebot import types import constants  class ButtonsManager():  def \_\_init\_\_(self, news\_stack):  self.news\_stack = news\_stack   def getSeeNewsButton(self):  if self.news\_stack.pt >= len(self.news\_stack.news): return None   return types.InlineKeyboardButton(  self.news\_stack.see\_text(),  callback\_data=constants.SEE\_CMD  )   def getEjectNewsButton(self):  if self.news\_stack.pt >= len(self.news\_stack.news): return None   return types.InlineKeyboardButton(  self.news\_stack.eject\_text(),  callback\_data=constants.EJECT\_CMD  )   def getAddNewsButton(self):  return types.InlineKeyboardButton(  self.news\_stack.add\_text(),  callback\_data=constants.ADD\_CMD  )   def getToBeginButton(self):  if self.news\_stack.pt == 0: return None   return types.InlineKeyboardButton(  self.news\_stack.to\_begin\_text(),  callback\_data=constants.TO\_BEGIN\_CMD  )   *# news\_stack.py* import constants  class NewsStack():  def \_\_init\_\_(self):  self.news = []  self.pt = 0  self.last\_message = ''   def see(self):  news = self.news[self.pt]  self.pt += 1  return news   def see\_text(self):  return constants.SEE\_CMD\_TEXT.format(len(self.news) - self.pt)   def eject(self):  news = self.news[self.pt]  del self.news[self.pt]  return news   def eject\_text(self):  return constants.EJECT\_CMD\_TEXT.format(len(self.news) - self.pt)   def add\_text(self):  return constants.ADD\_CMD\_TEXT.format()   def add\_last\_message(self):  self.news.append(self.last\_message)   def to\_begin\_text(self):  return constants.TO\_BEGIN\_CMD\_TEXT.format()   *# constants.py*  SEE\_CMD = 'see command' EJECT\_CMD = 'eject command' ADD\_CMD = 'add command' TO\_BEGIN\_CMD = 'to begin command'  SEE\_CMD\_TEXT = 'Посмотреть запись {}' EJECT\_CMD\_TEXT = 'Извлечь запись {}' ADD\_CMD\_TEXT = 'Добавить последнее сообщение как новость' TO\_BEGIN\_CMD\_TEXT = 'Вернуться к началу'  MENU\_TEXT = 'Меню команд'  SEE\_RES\_TEXT = 'Запись просмотрена' EJECT\_RES\_TEXT = 'Запись извлечена' ADD\_RES\_TEXT = 'Запись добавлена' TO\_BEGIN\_RES\_TEXT = 'В начале новостей' CMD\_NOT\_FOUND = 'Такой команды нет'%   *# main.py* import telebot import constants from telebot import types from news\_stack import NewsStack from buttons\_manager import ButtonsManager   token = open('token').read() bot = telebot.TeleBot(token)   def actions\_markup():  markup = types.InlineKeyboardMarkup()  seeNews = button\_manager.getSeeNewsButton()  ejectNews = button\_manager.getEjectNewsButton()  addNews = button\_manager.getAddNewsButton()  toBegin = button\_manager.getToBeginButton()   seeNews and markup.add(seeNews)  ejectNews and markup.add(ejectNews)  addNews and markup.add(addNews)  toBegin and markup.add(toBegin)   return markup   @bot.message\_handler(regexp=".\*") def any\_message(message):  news\_stack.last\_message = message.text  markup = actions\_markup()   bot.send\_message(message.chat.id, constants.MENU\_TEXT, reply\_markup=markup)   @bot.callback\_query\_handler(func=lambda call: True) def handle(call):  chat = call.message.chat  answer\_text = constants.CMD\_NOT\_FOUND   if call.data == constants.SEE\_CMD:  bot.send\_message(chat.id, news\_stack.see())  answer\_text = constants.SEE\_RES\_TEXT  elif call.data == constants.EJECT\_CMD:  bot.send\_message(chat.id, news\_stack.eject())  answer\_text = constants.EJECT\_RES\_TEXT  elif call.data == constants.ADD\_CMD:  news\_stack.add\_last\_message()  answer\_text = constants.ADD\_RES\_TEXT  elif call.data == constants.TO\_BEGIN\_CMD:  news\_stack.pt = 0  answer\_text = constants.TO\_BEGIN\_RES\_TEXT   markup = actions\_markup()   bot.send\_message(call.message.chat.id, answer\_text, reply\_markup=markup)  bot.answer\_callback\_query(call.id)   if \_\_name\_\_ == '\_\_main\_\_':  news\_stack = NewsStack()  button\_manager = ButtonsManager(news\_stack)   bot.polling() |
| --- |

**Пример работы:**

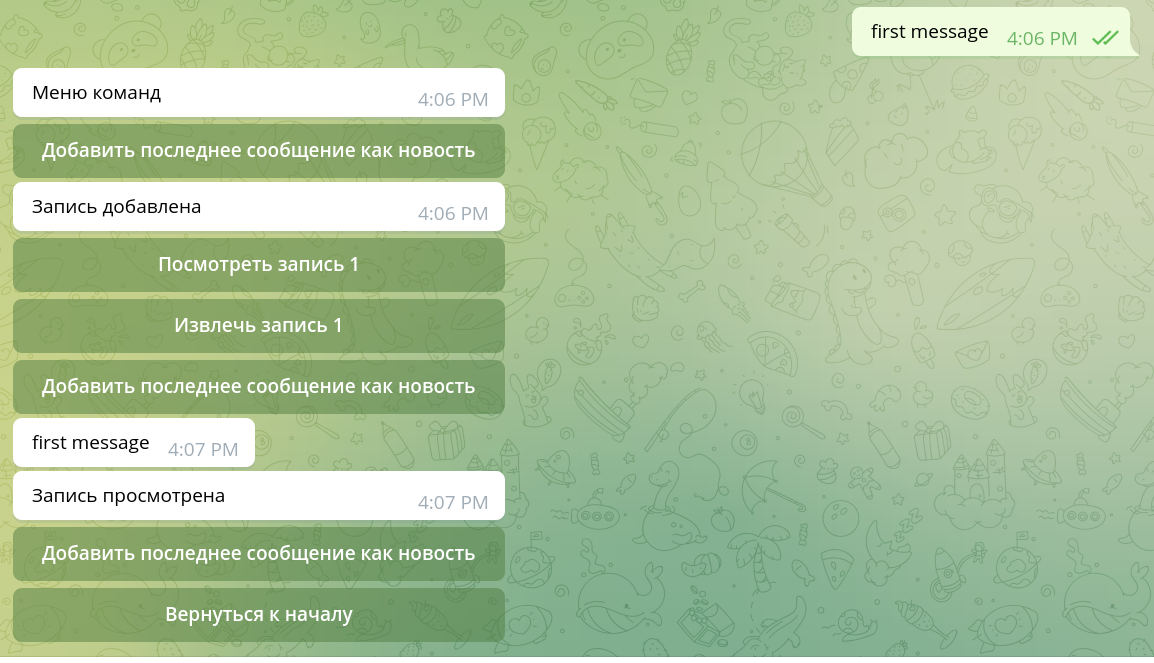
Новость

****

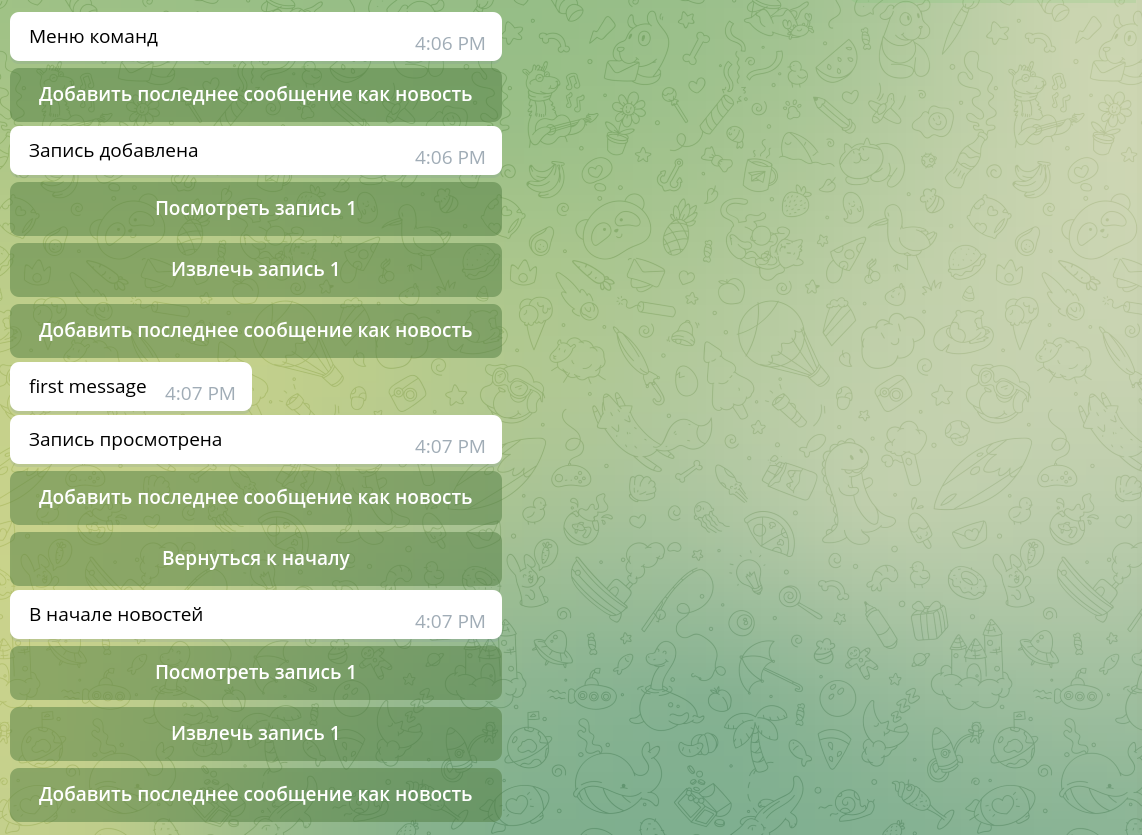
Добавить новость

****

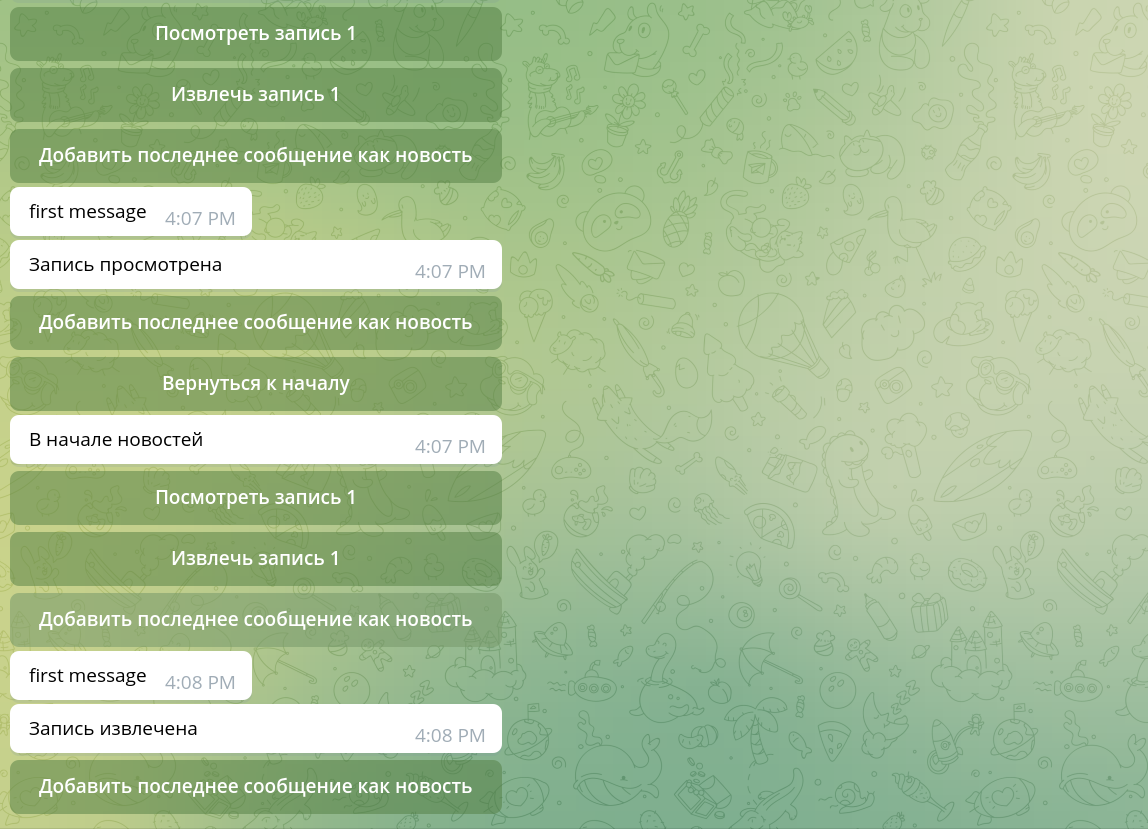
Посмотреть новость

****

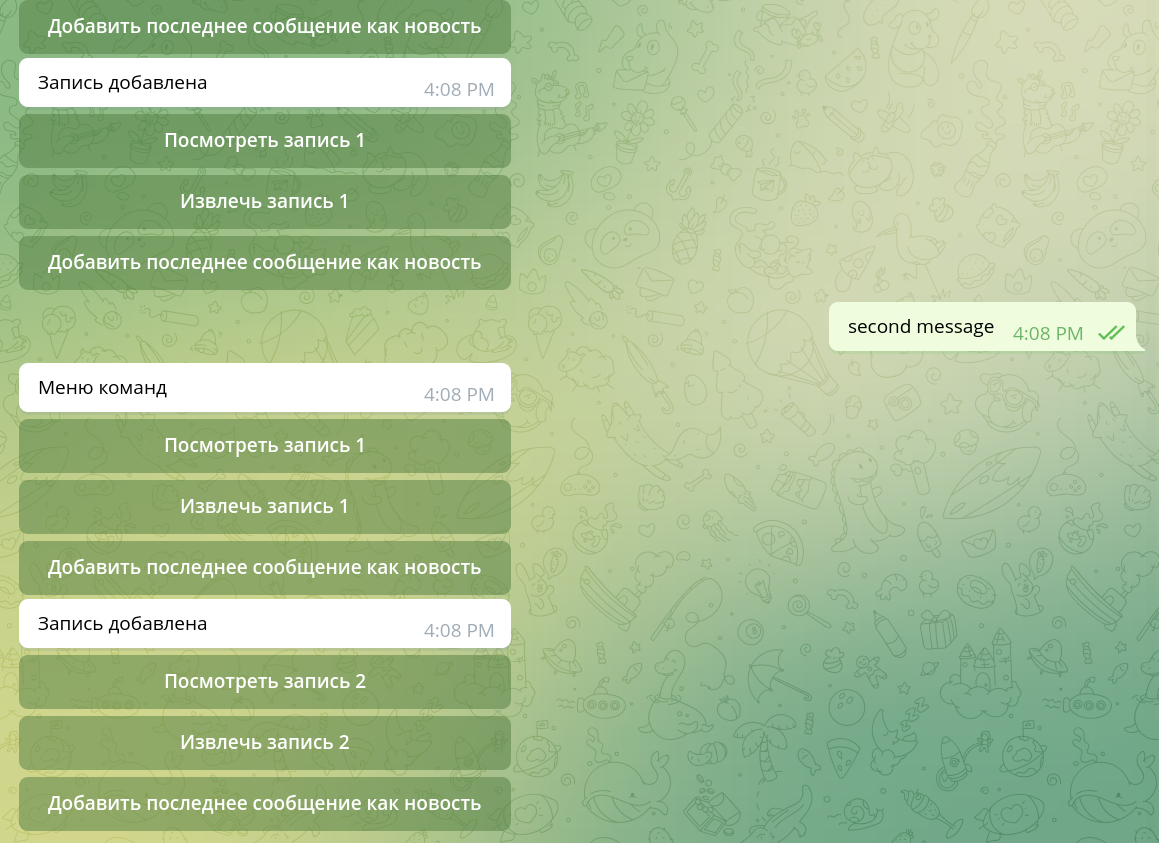
В начало новостей

****

Извлечь новость

****

Добавить две новости

****