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()
```

```python
# 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()
```

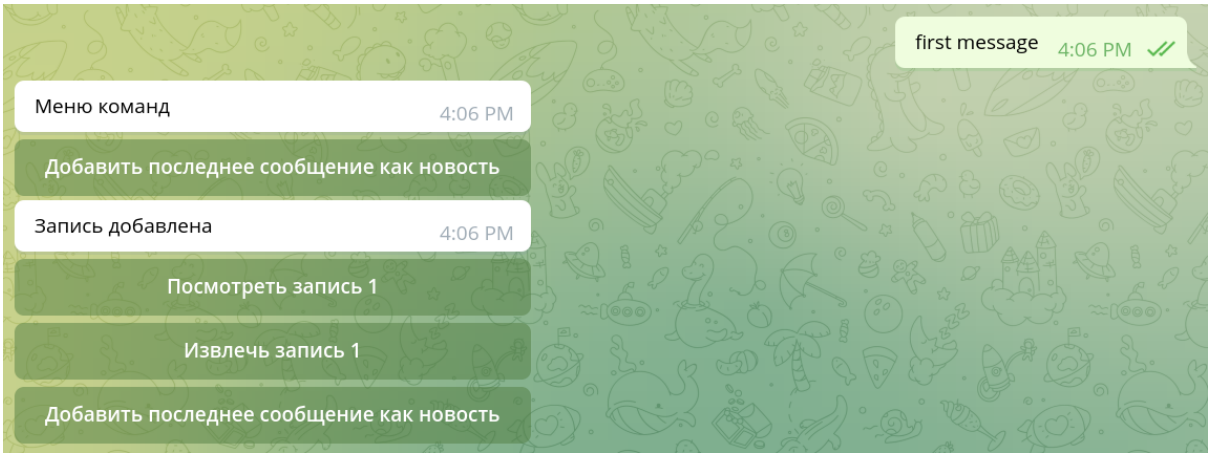
```
button_manager = ButtonsManager(news_stack)

bot.polling()
```

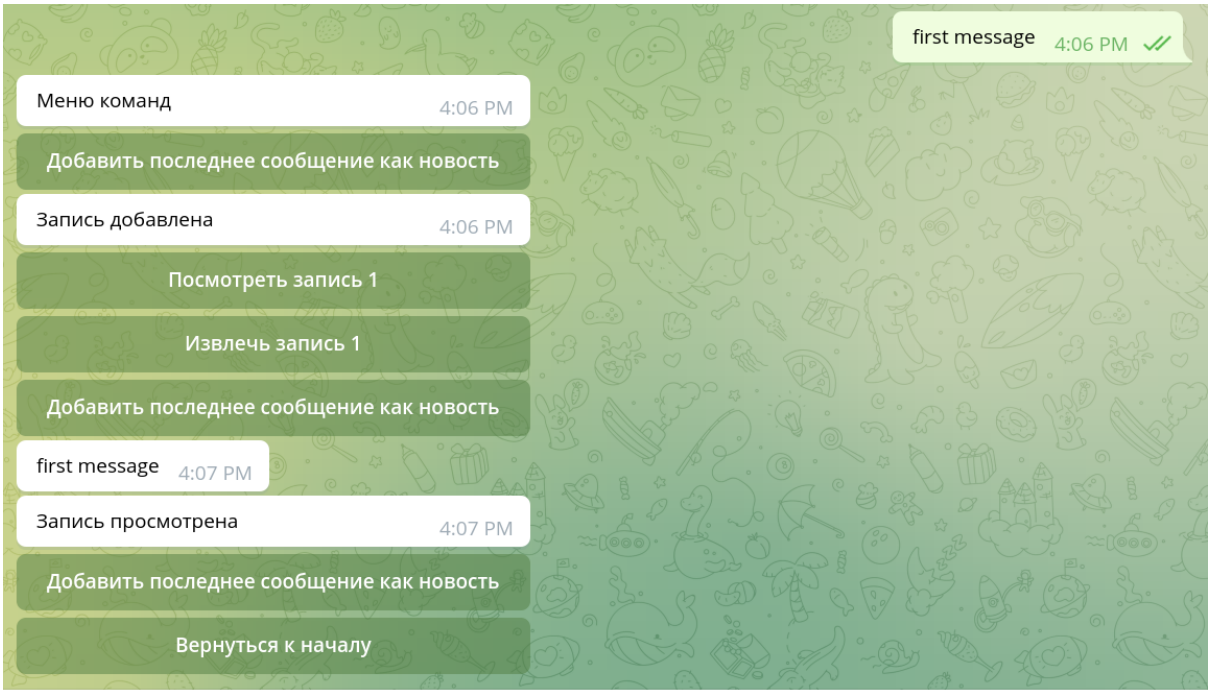
Пример работы:

Новость

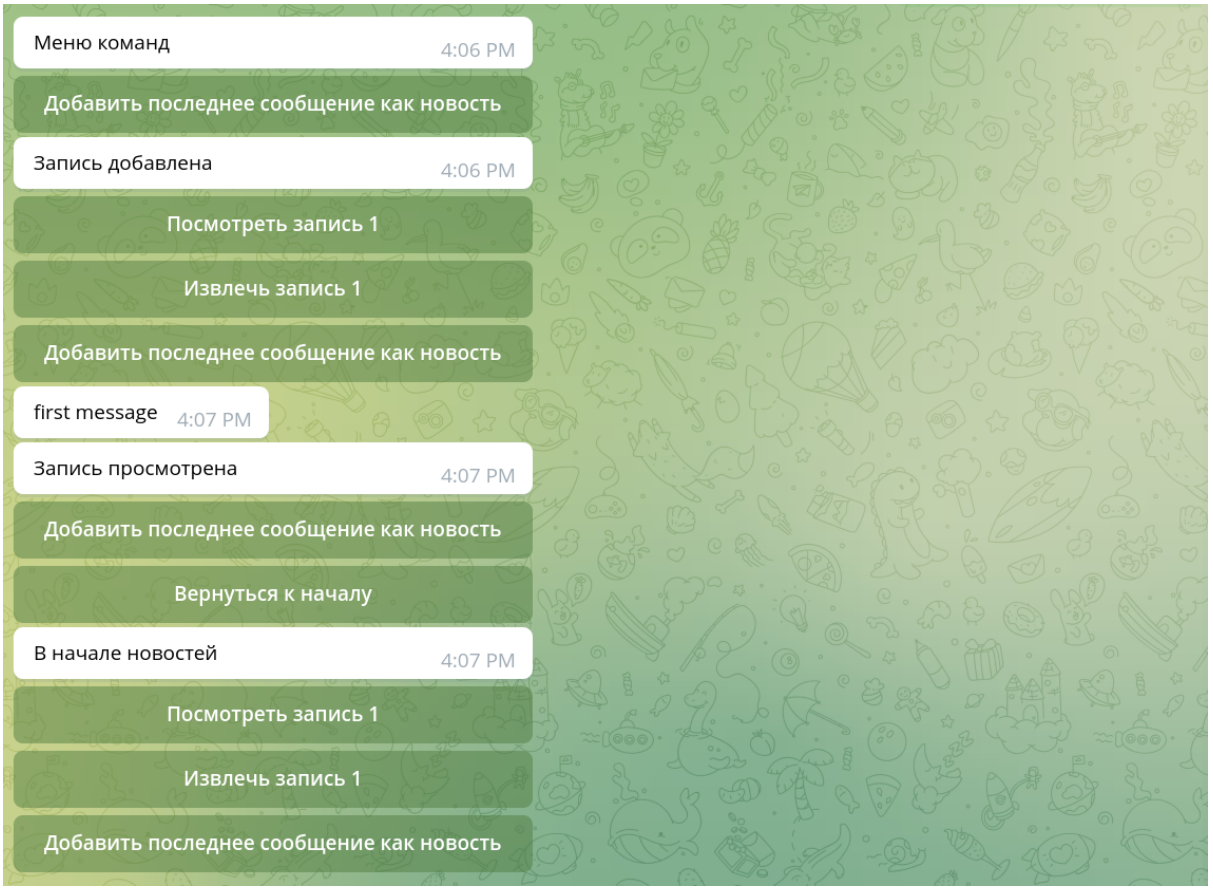
Добавить новость



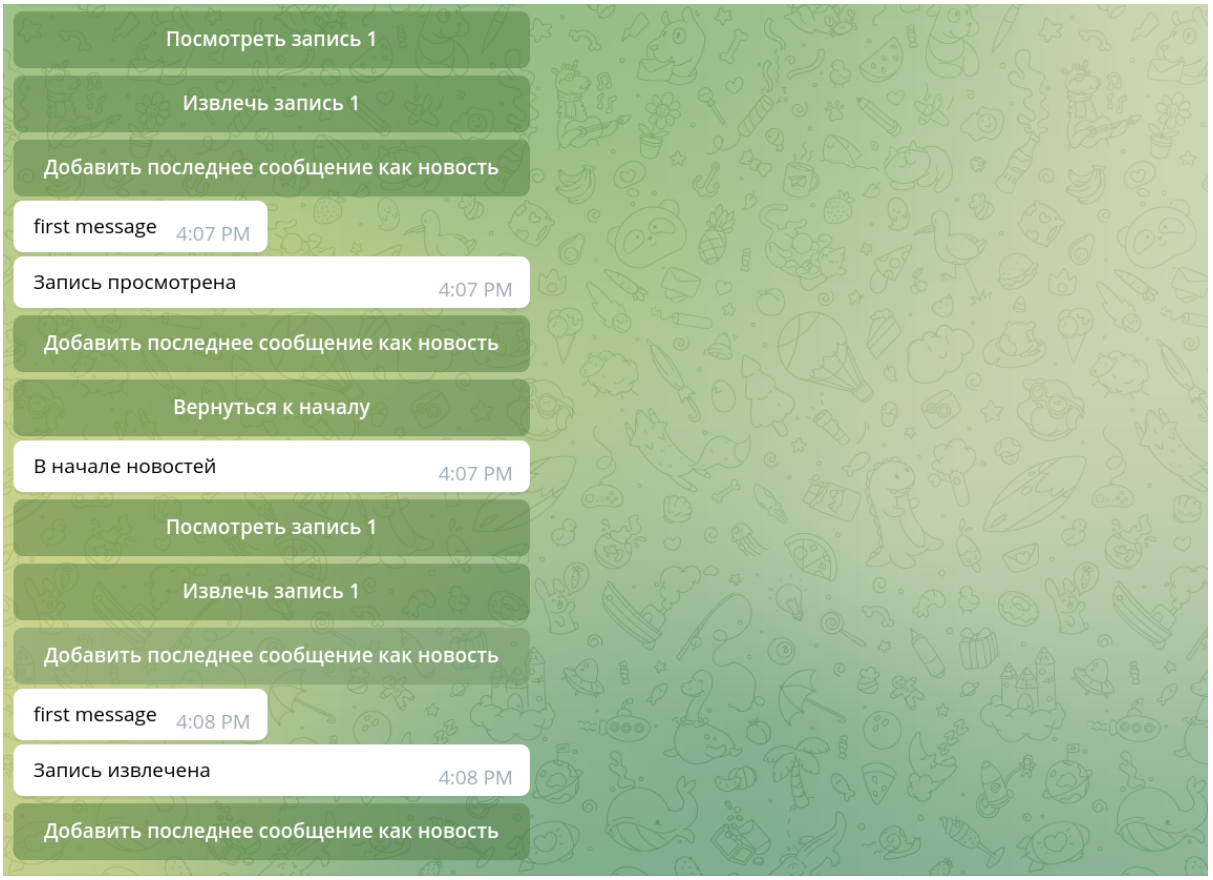
Посмотреть новость



В начало новостей



Извлечь новость



Добавить две новости