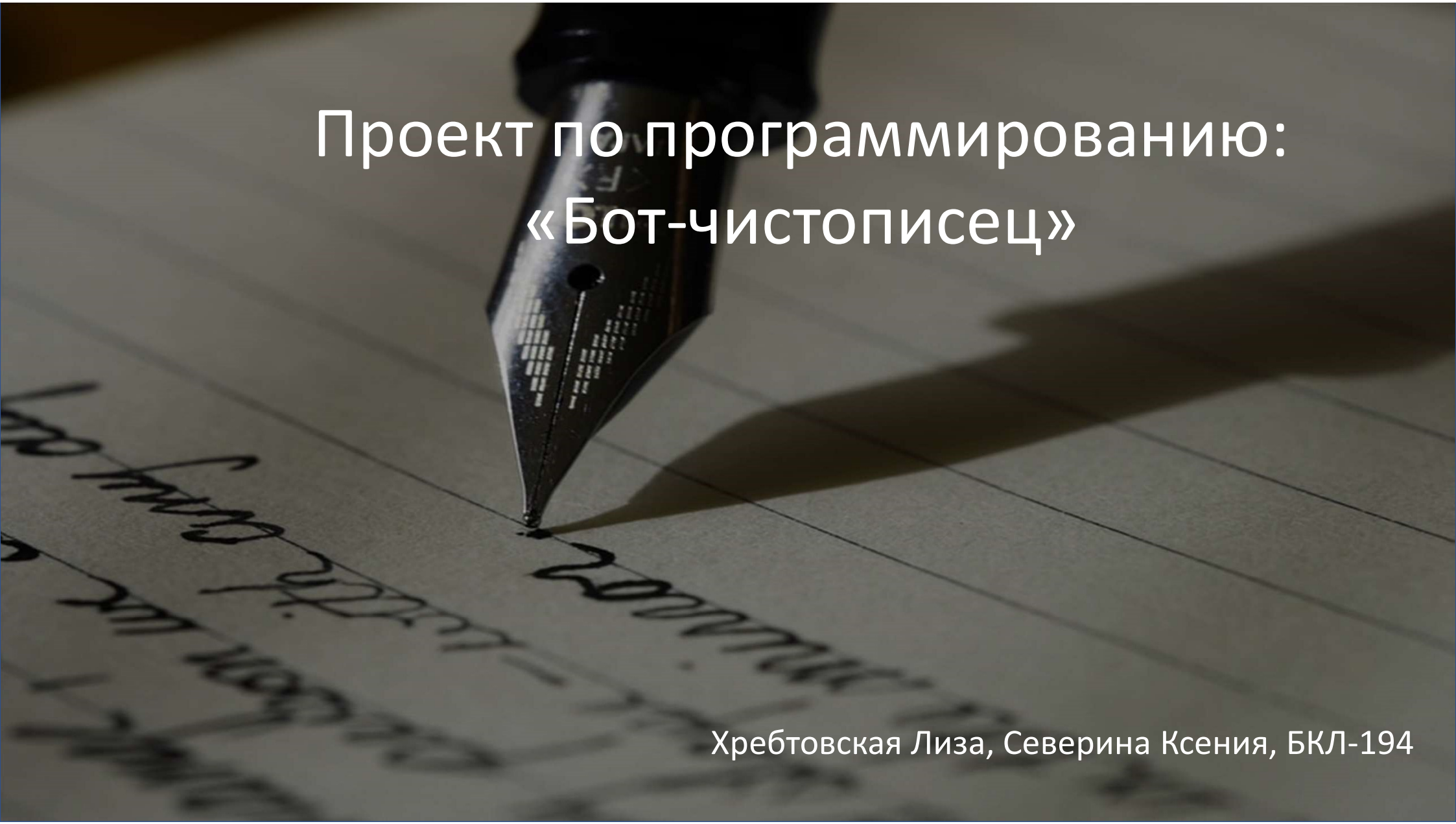
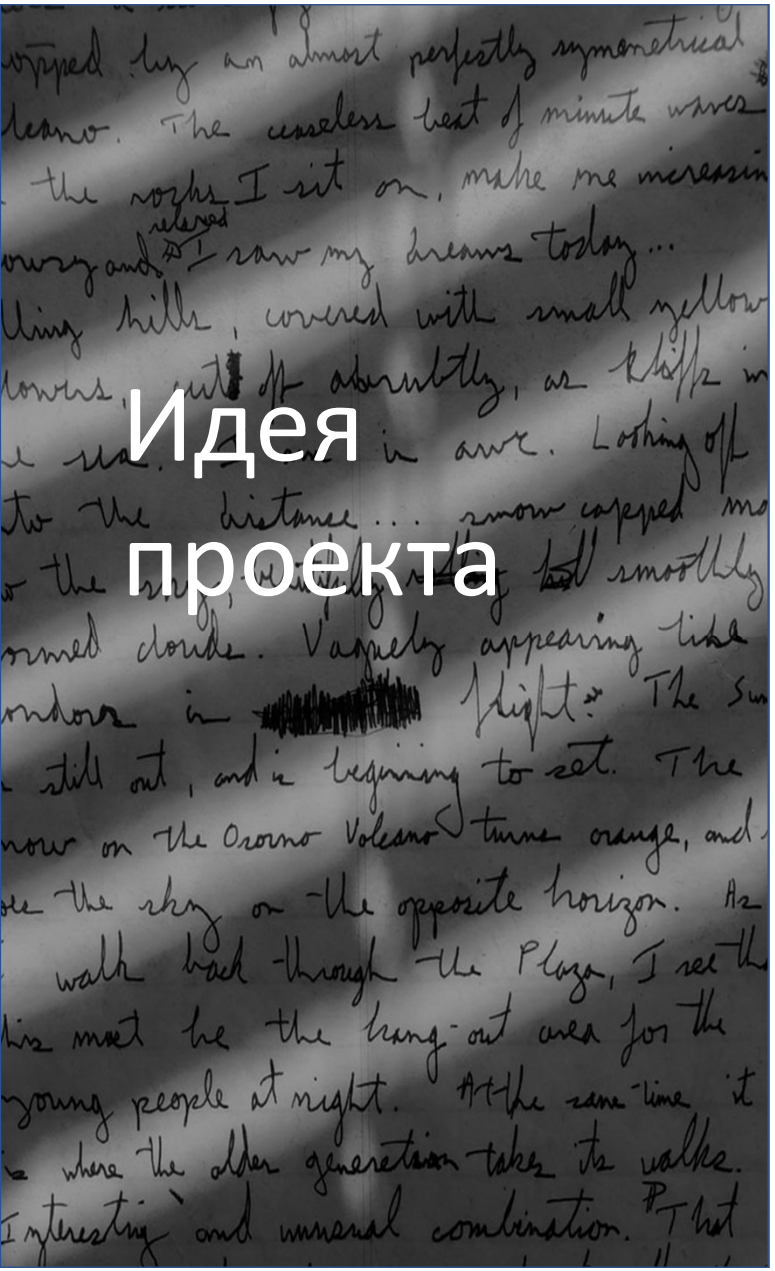


Проект по программированию: «Бот-чистописец»

Хребтовская Лиза, Северина Ксения, БКЛ-194



Идея проекта

Создать программу, которая могла бы анализировать текст и улучшать его на стилистическом уровне:

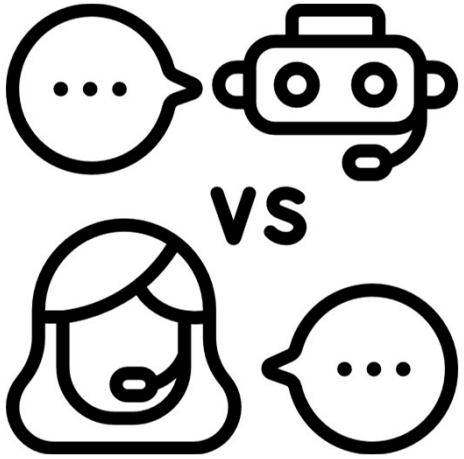
1) Находить слова-паразиты и:

a. удалять их, если они не несут смысловой нагрузки

b. предлагать пользователю список синонимов к словам-паразитам

2) Предлагать синонимы к любому слову, введённому пользователем

Как работает бот?



Команды

/start 20:27

Привет, я бот-чистописец. У меня есть две способности:

(1) Отправь мне сообщение вида «/fixtext текст» и я его почищу: уберу слова-паразиты и обозначу неявные формы.

(2) Отправь мне сообщение вида «/syndict слово» и я выдам тебе 10 синонимов этого слова с сайта synonymonline.ru.

Чтобы просмотреть список возможных команд, введи /help.

20:27

Write a message...

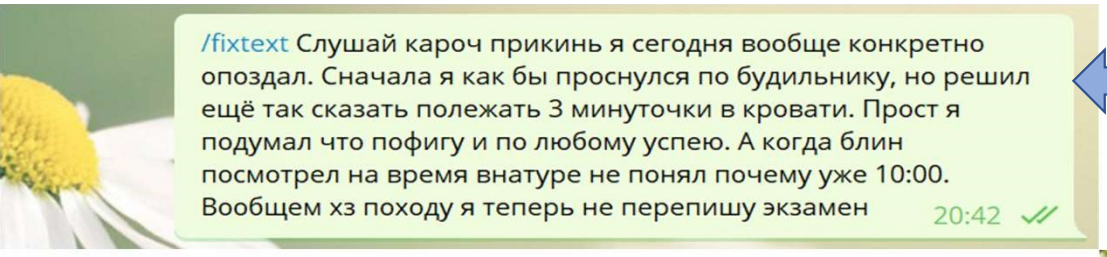
Б /start старт

Б /help помощь

Б /fixtext удаление/замена слов-паразитов

Б /syndict словарь синонимов

/

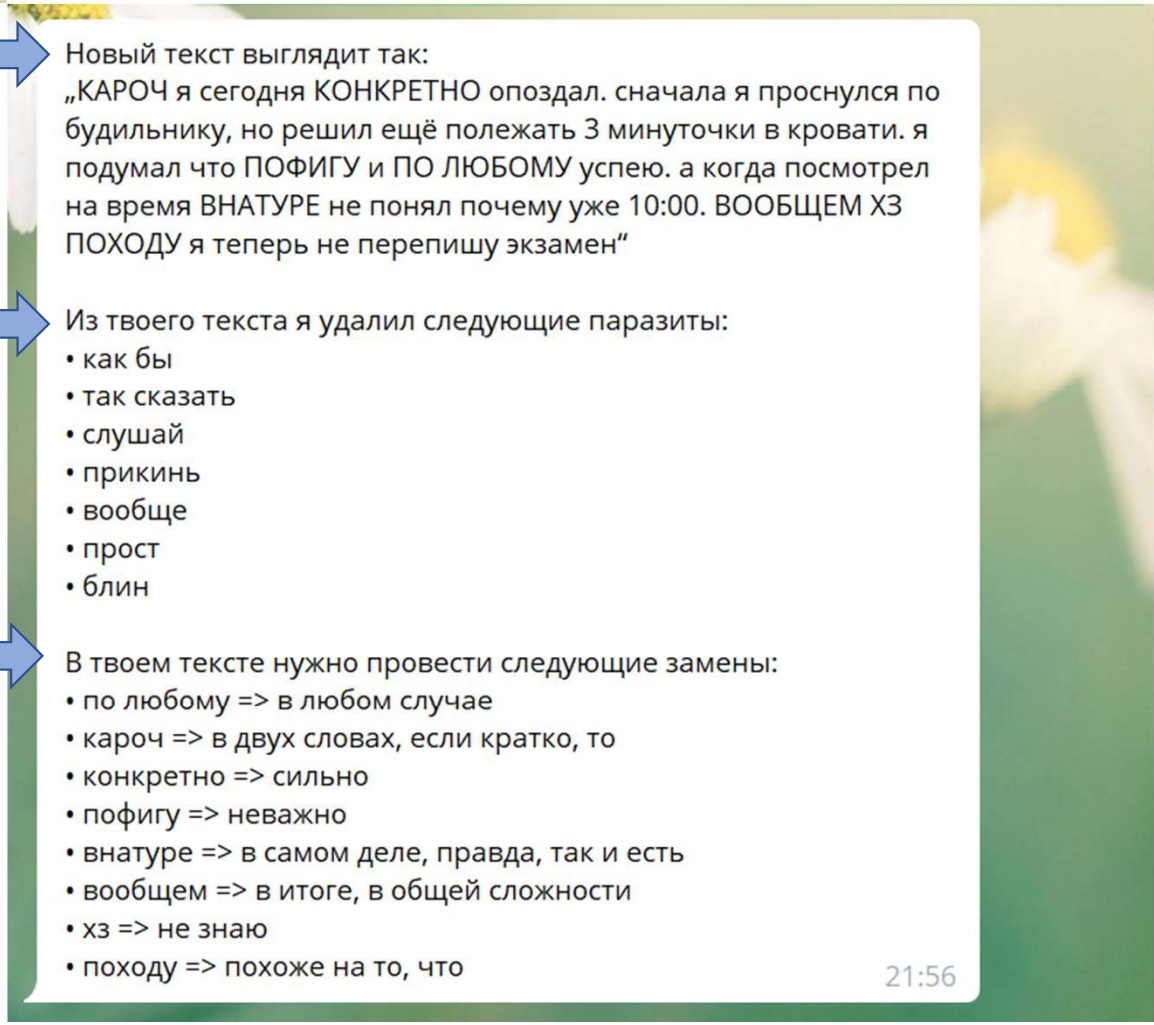


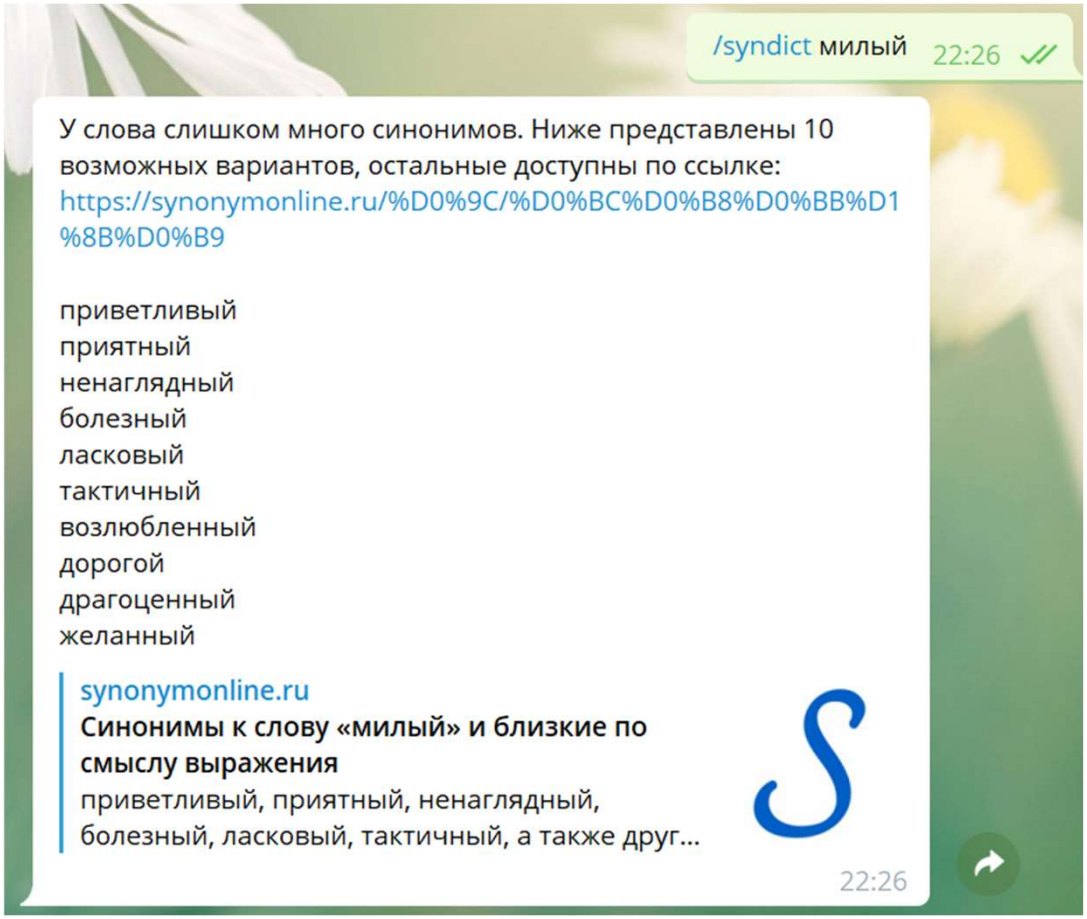
Пример текста

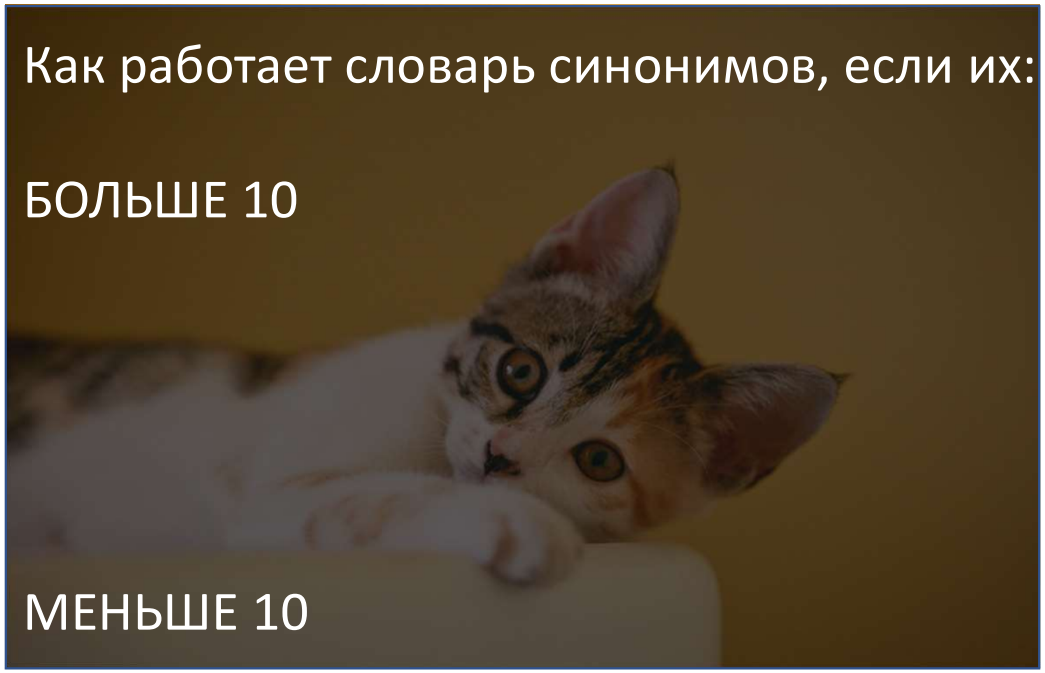
1. Бот возвращает текст, удаляя слова-паразиты, к которым нет синонимов в словаре + выделяет капсом слова, к которым есть синонимы

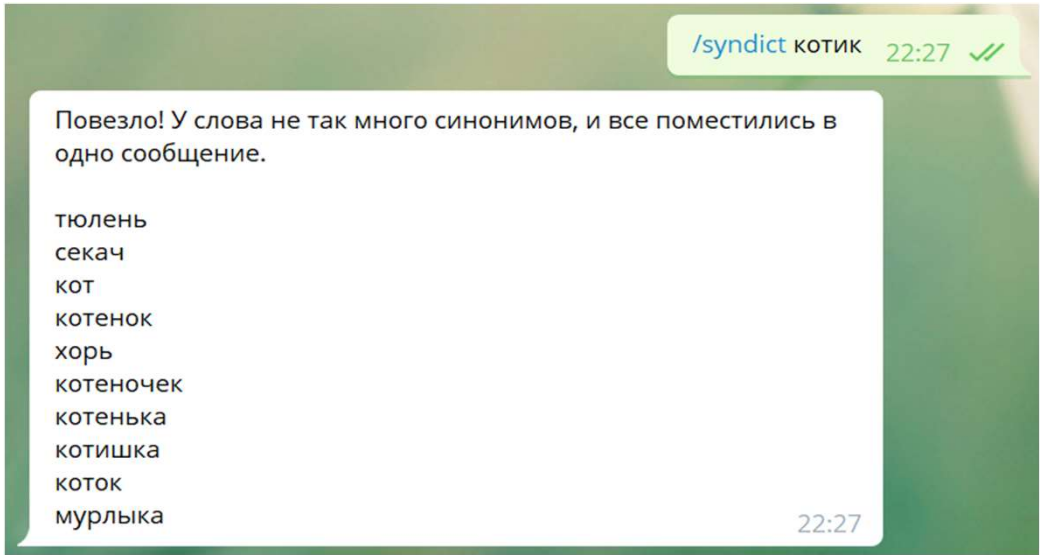
2. Программа указывает какие слова были удалены в полученном тексте

3. Программа предлагает синонимы к выделенным капсом словам-паразитам









База с синонимами слов

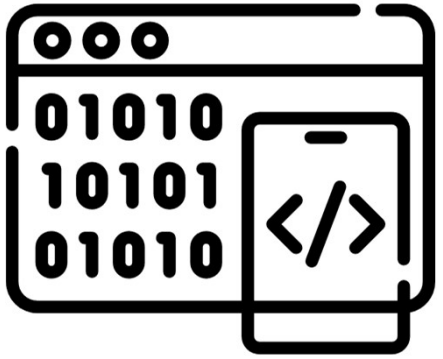
Интересные места в коде

1. Удаление/замена слов-паразитов

```
for parasyte in DICT_PAR:
# если паразит есть в тексте...
if parasyte in text:
  parasList.append(parasyte)
  # и если паразит ни на что не заменяется...
  if DICT_PAR[parasyte] == "":
    # то мы его удаляем из текста
    text = text.replace(parasyte, "")
    deleteList.append(parasyte)
  # но если заменяется...
  else:
    # то выделяем в тексте капсом...
    text = text.replace(parasyte, parasyte.upper())
    # и записываем паразит в отдельный список
    fixList.append(parasyte)
```

I. изначальный код:

Удалял “ну” в “уснуть”((



```
new_wordList = wordList

# проверяем каждое слово с новом списке слов текста
for word in range(len(new_wordList)):
  # все буквы маленькие
  new_wordList[word] = new_wordList[word].lower()
  for parasyte in DICT_PAR:
    # если слово является паразитом...
    if new_wordList[word] == parasyte:
      # записываем его в список всех паразитов
      parasList.append(parasyte)
      # если не заменяется, то удаляем и записываем в список удаленных
      if DICT_PAR[parasyte] == "":
        wordList[word] = ""
        deleteList.append(parasyte)
      # если заменяется, то выделяем капсом и записываем в список для фикса
      else:
        wordList[word] = wordList[word].upper()
        fixList.append(parasyte)
```

II. поправленный код:

не реагировал на паразиты из двух слов(

III. Правильно работающий код

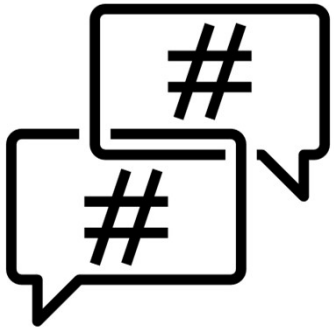
```
text = text.lower()

# если паразит = 2 слова
for parasyte in DICT_PAR:
  if " " in parasyte:
    # если паразит есть в тексте...
    if parasyte in text:
      parasList.append(parasyte)
      # и если паразит ни на что не заменяется...
      if DICT_PAR[parasyte] == "":
        # то мы его удаляем из текста
        text = text.replace(parasyte, "")
        deleteList.append(parasyte)
      # но если заменяется...
      else:
        # то выделяем в тексте капсом...
        caps = parasyte.upper()
        text = text.replace(parasyte, caps)
        # и записываем паразит в отдельный список
        fixList.append(parasyte)

# делаем список слов с измененными двойными паразитами
wordList = list_making(text)
new_wordList = wordList
```

```
# если паразит = 1 слово
for word in range(len(new_wordList)):
  for parasyte in DICT_PAR:
    # если слово является паразитом...
    if new_wordList[word] == parasyte:
      # записываем его в список всех паразитов
      parasList.append(parasyte)
      # если не заменяется, то удаляем и записываем в список удаленных
      if DICT_PAR[parasyte] == "":
        wordList[word] = ""
        deleteList.append(parasyte)
      # если заменяется, то выделяем капсом и записываем в список для фикса
      else:
        wordList[word] = wordList[word].upper()
        fixList.append(parasyte)
```

Интересные места в коде

```
"""
СОЗДАЕМ СПИСОК СЛОВ ТЕКСТА
ЧТОБЫ ЗНАКИ ПРЕПИНАНИЯ СЧИТАЛИСЬ ЗА СЛОВА
"""
# тире нет тк оно и так отделено пробелами с обеих сторон
def list_making(text):
  text = text.replace(".", " .")
  text = text.replace(",", " ,")
  text = text.replace("!", " !")
  text = text.replace("?", " ?")
  text = text.replace(":", " :")
  text = text.replace(";", " ;")
  text = text.replace("(", "( ")
  text = text.replace(")", " )")
  text = text.replace("«", "« ")
  text = text.replace("»", " »")
  text = re.sub(r'\"(.+?)\"', r'« \1 »', text)
  wordList = text.split()

  return wordList
```

2. Функция, отделяющая пробелами знаки препинания



Интересные места в коде

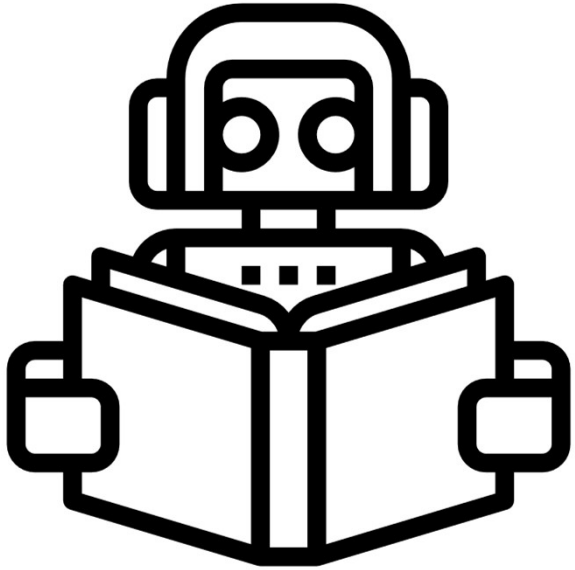
```
"""
УБРАТЬ ПРОБЕЛЫ КОГДА ЗАМЕНЯЕМ ПАРАЗИТЫ
"""
def space_del(text):
  # убираем кучу пробелов, оставляем один красивенький
  text = re.sub(r"\s+", " ", text)

  # убираем пробелы в начале и конце строки
  if text.startswith(" "):
    text = "" + text[1:]
  if text.endswith(" "):
    text = text[:-1] + ""

  # убираем одинарные пробелы перед знаками препинания + после
  text = text.replace(" .", ".")
  text = text.replace(" ,", ",")
  text = text.replace(" !", "!")
  text = text.replace(" ?", "?")
  text = text.replace(" :", ":")
  text = text.replace(" ;", ";")
  text = text.replace(" )", ")")
  text = text.replace("( ", "(")
  text = text.replace(" »", "»")
  text = text.replace("« ", "«")

  return text
```

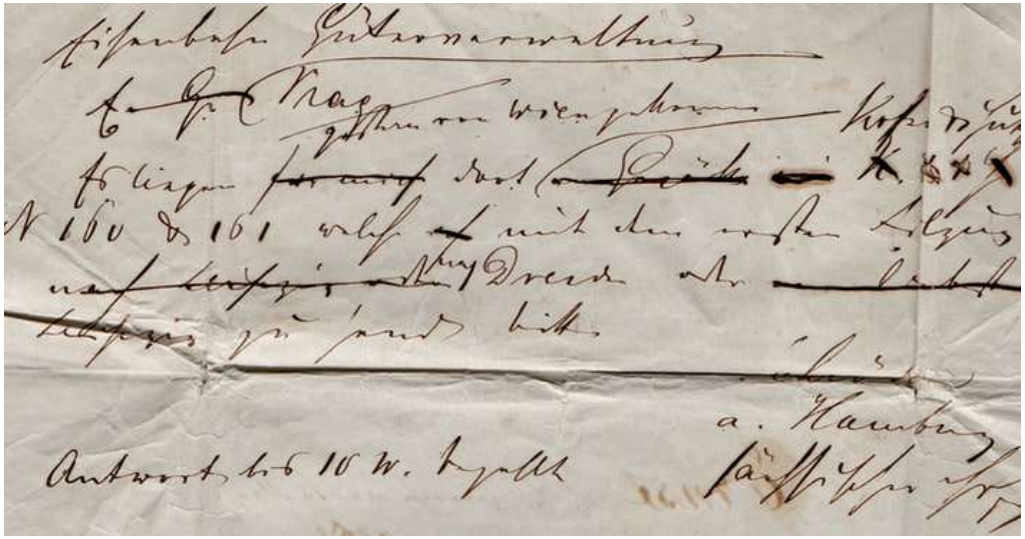
3. Функция, удаляющая лишние пробелы, которые появлялись в процессе работы бота

Что получилось?
Бот, который находит в отправленном тексте слова-паразиты, удаляет/заменяет их в зависимости от конкретного случая

+

Бот обладает базой синонимов для любых слов, предлагает 10 самых частотных синонимов





Модули, использованные в коде:

- re
- telebot
- requests
- time