МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»

# Отчет по лабораторным работам
# по курсу «Технологии параллельного программирования»

Выполнил: Полиха Александр Владимирович
Группа: М8О-403Б-22
Преподаватель: Кухтичев Антон Алексеевич

Москва, 2025

## Цель работы

Разработать поисковой движок на C++ для выбранного корпуса документов. Этот корпус документов должен собираться поисковым роботом, хранится корпус в MongoDB.
Привести несколько запросов к существующим поисковикам, указать недостатки.

## Описание данных

В качестве корпуса данных был выбран
В качестве первого источника был выбран набор статей из Wikipedia, а именно – Биографии музыкальных деятелей. Этот ресурс известен своим большим количество статей. В качестве второго источника был выбран belcanto, в нем насчитывается порядка 5 000 биографий, а в Wikipedia около больше 30 000. Для парсинга был написан поисковой робот на Python, который собирает html Документы, выполняя правила robots.txt.
Текст у этих документов состоит из сырого html, есть мета информация по кодировке, описанию, ключевые слова, автор.
Документ содержит:

1. Основной контент статьи
2. Заголовок
3. Дата/время публикации
4. Рубрика/тема
5. Текст статьи
6. Изображения/видео (иногда) с подписями/источниками
7. Ссылки внутри текста и блоки по теме
8. Навигация и сервисные блоки
9. Элементы авторизации/регистрации
10. URL

Для выбранных источников существует поисковик на сайте, также можно воспользоваться поиском Google, Yandex. Поэтому их можно использовать для выполнения лабораторной работы.

## Примеры существующих поисковиков

Поиск на сайте:

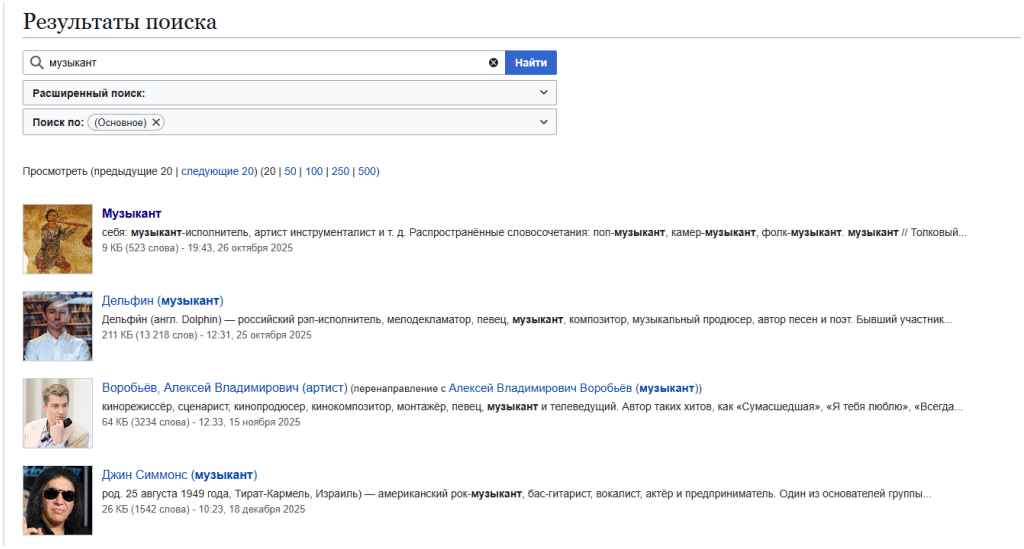


Рисунок 1 – поиск на сайте

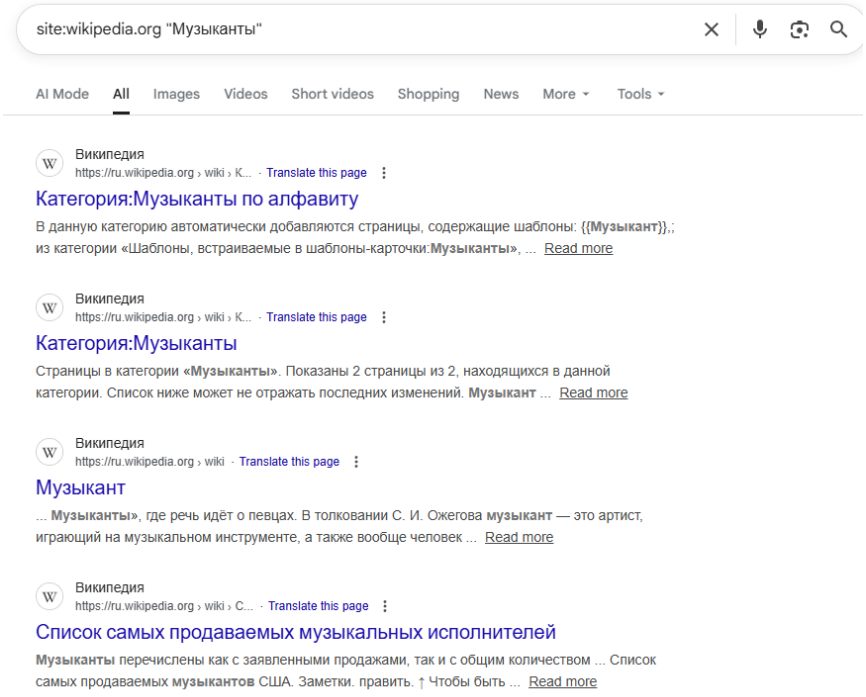


Рисунок 2 – поиск через Google

Недостатки существующих поисковиков:

- Для обычного пользователя любого браузера может быть трудно найти нужное содержимое без дополнительных знаний по составлению грамотного запроса.
- Необъективность, при одинаковом запросе, без использования параметров поисковика, результаты могут быть разными, например, Яндекс будет продвигать их маркетплейс на самую первую позицию при запросе «купить наушники», в то время как google подвинет на первую позицию в выпадающем списке ozon.

- Часто после выполнения запроса, в результатах отображаются только популярные источники.

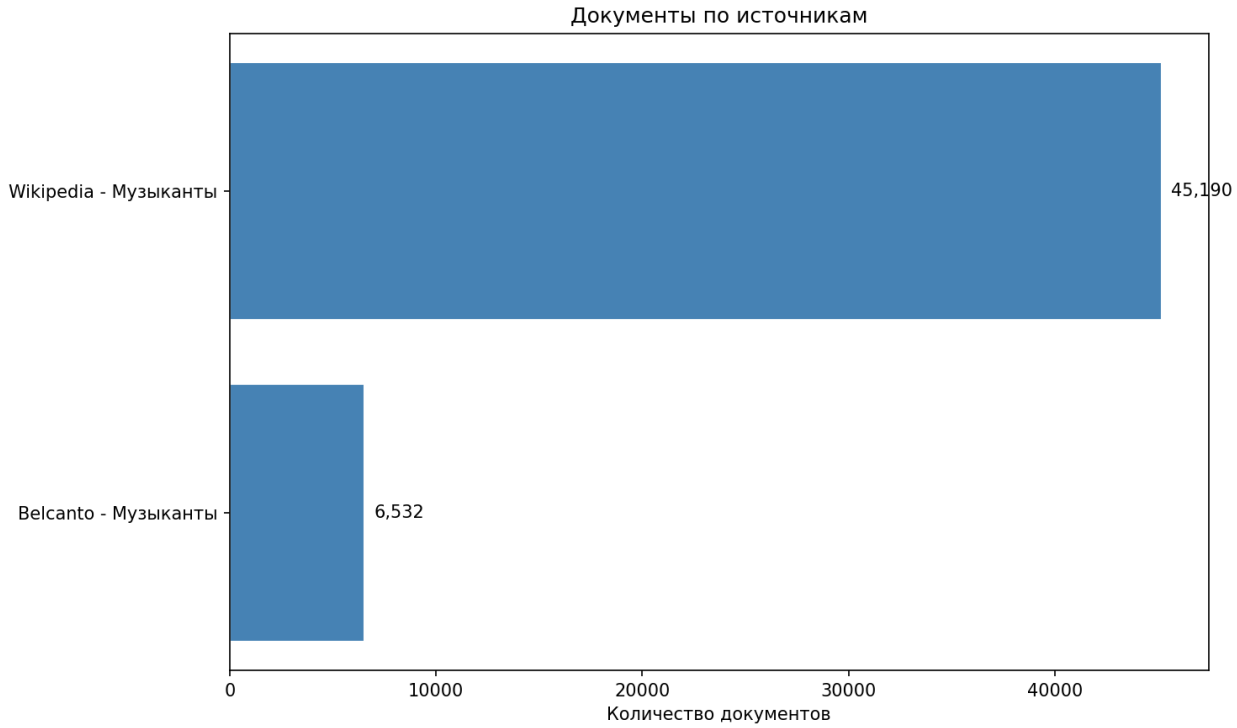


Рисунок 3 – Распределение документов по источникам

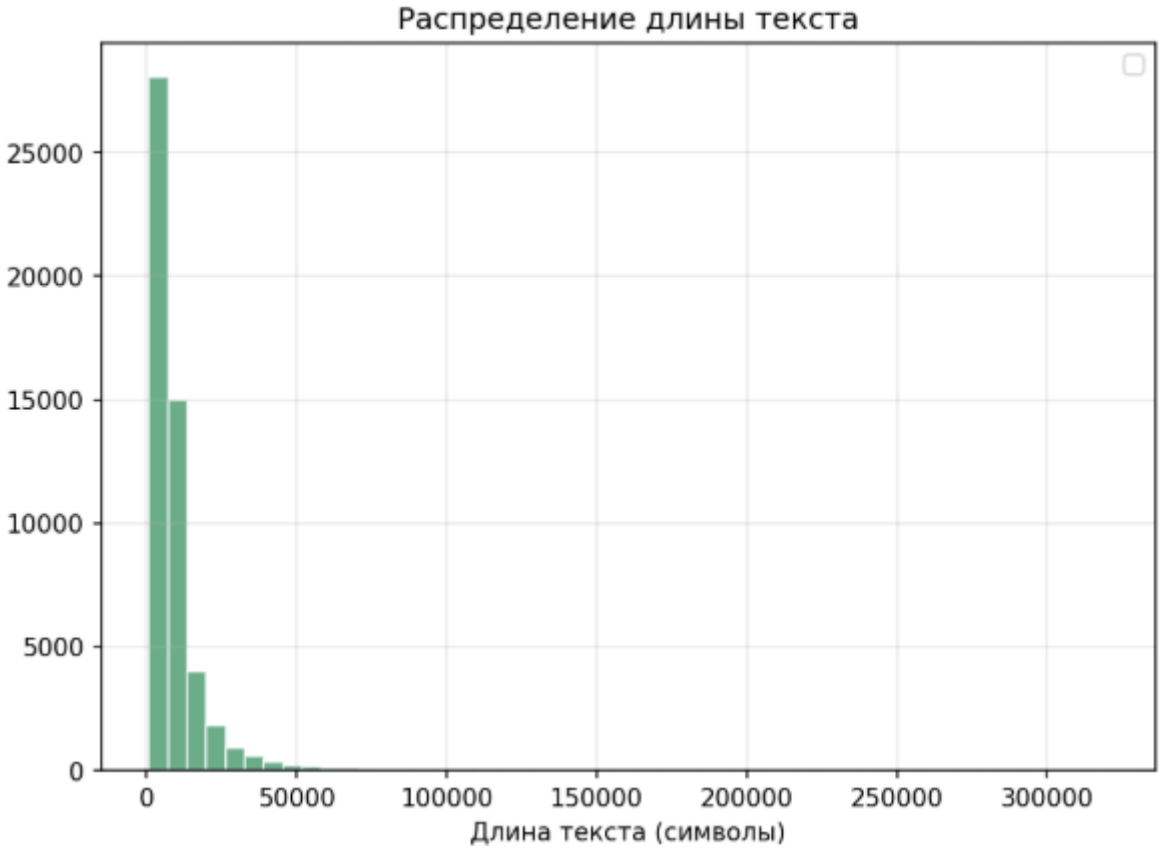


Рисунок 4 – Распределение количества символов в документах

После работы поискового робота мы получаем:

- 51 722 – количество документов.
- 10 287 символов – средний размер текста в документе.
- Объем корпуса – 1.47 GB

## Токенизация

Токенизация — это процесс преобразования исходного текста в последовательность элементарных единиц, называемых токенами.
Перед токенизацией нужно убрать мусорные теги html вроде <script>, <meta>, <header> и другие.
Затем происходит приведение к нижнему регистру.
Если токен слишком маленький, то его не используем. Также не используем токен, если он полностью из цифр.
С токенизацией есть проблемы, например, токен «абдаль-мумин» - содержит дефис, для чего нужны отдельные правила для токенизации.
После работы индексатора были получены следующие результаты:
Документов: 51,722
Всего токенов: 53,295,593
Уникальных токенов: 1,321,300
Уникальных стемов: 835,856
Средняя длина токена: 7.15 символов
Размер индекса: 155.86 MB
Скорость токенизации: 2812.43 КБ/сек
Постройка индекса заняла 1682.31 секунды

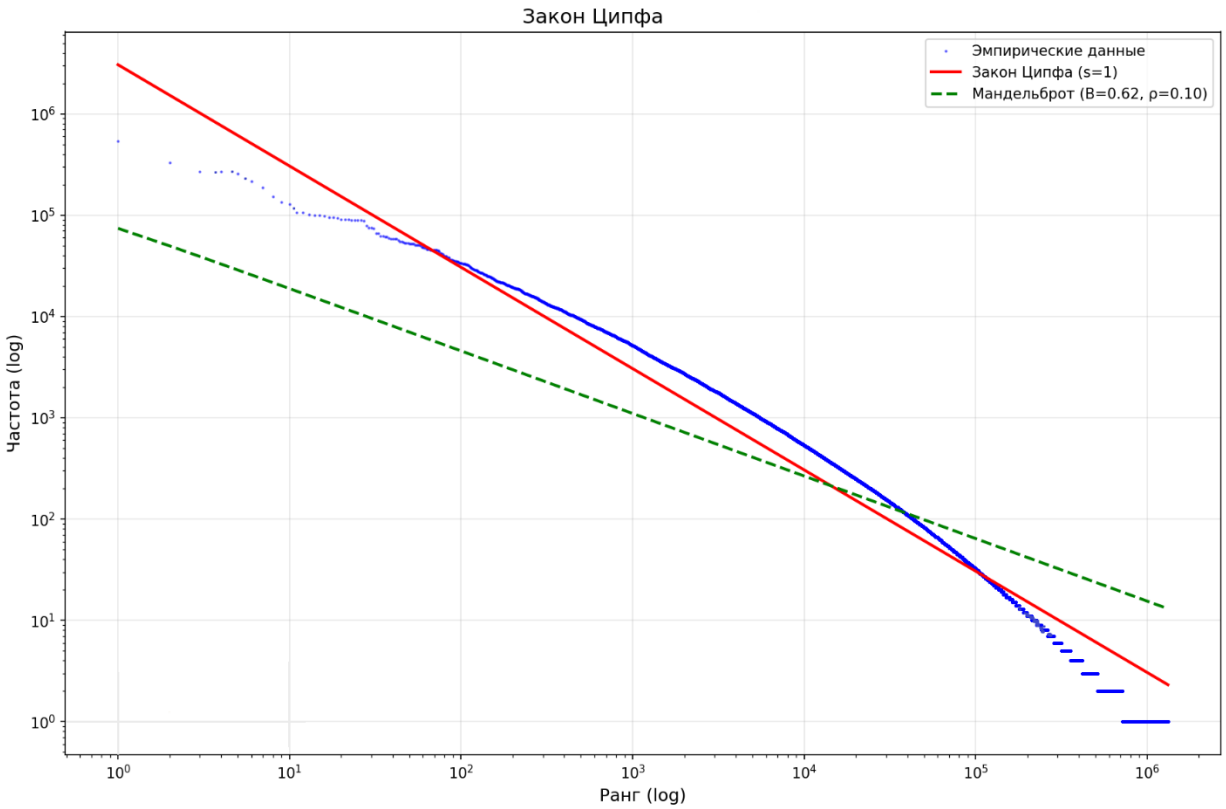


Рисунок 5 – закон Ципфа

## Стемминг

Плюсы:

- Быстрый поиск
- Меньший размер индекса (меньше уникальных термов)

Минусы:

- Нельзя искать по точной форме слова

Запрос: пианист
Без стемминга: будет найдено только «пианист»
Со стеммингом:  «пиан»: Будет найдено пианист, пианино, фортепиано.

## Булев индекс

ПРЯМОЙ ИНДЕКС (Forward Index):
Для каждого документа:
4 байта - doc_id
2 байта - длина title
N байт - title (UTF-8)
2 байта - длина url
M байт - url (UTF-8)

ОБРАТНЫЙ ИНДЕКС (Inverted Index):
4 байта - количество термов
Для каждого терма:
1 байт - длина терма
N байт - терм (UTF-8)
4 байта - количество документов (DF)
4*DF байт - массив doc_id (отсортирован)

## Булев поиск

В наличии есть следующие операции: AND, OR, NOT, группировка(круглые скобки).

Результатами поиска являются ссылками на источники

## Примеры поиска

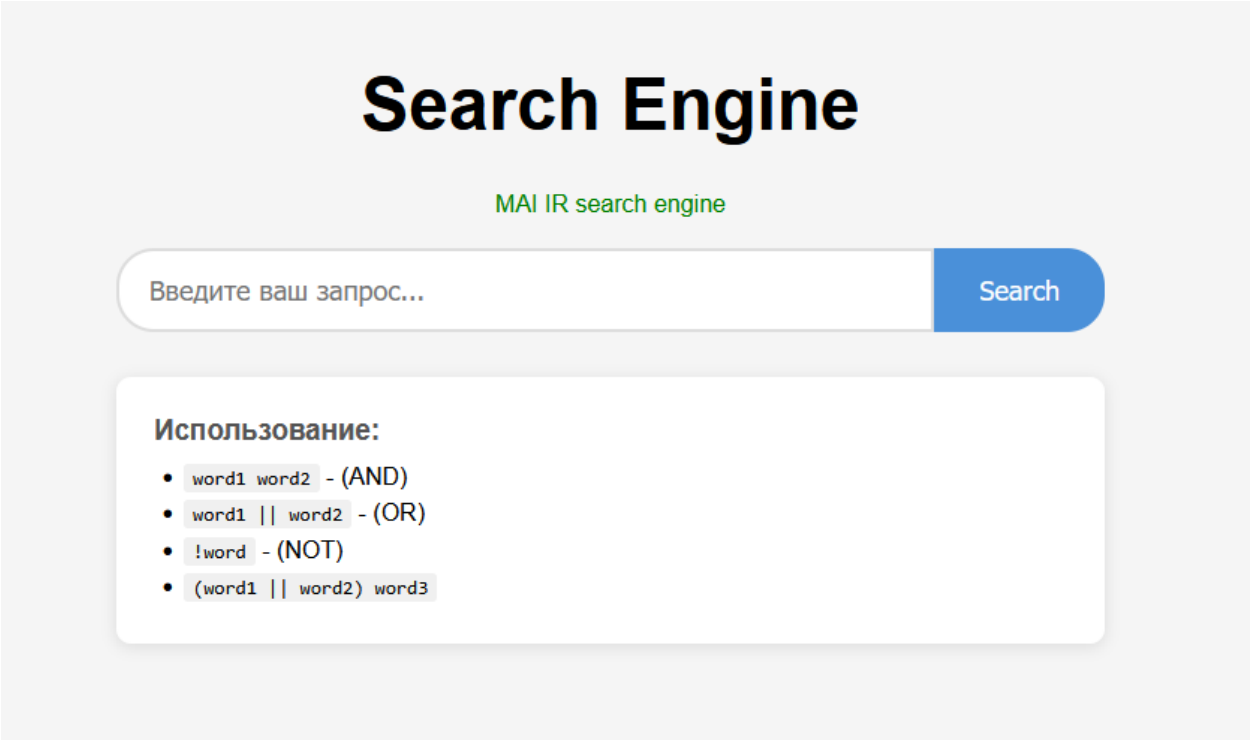


Рисунок 6 – web-интерфейс

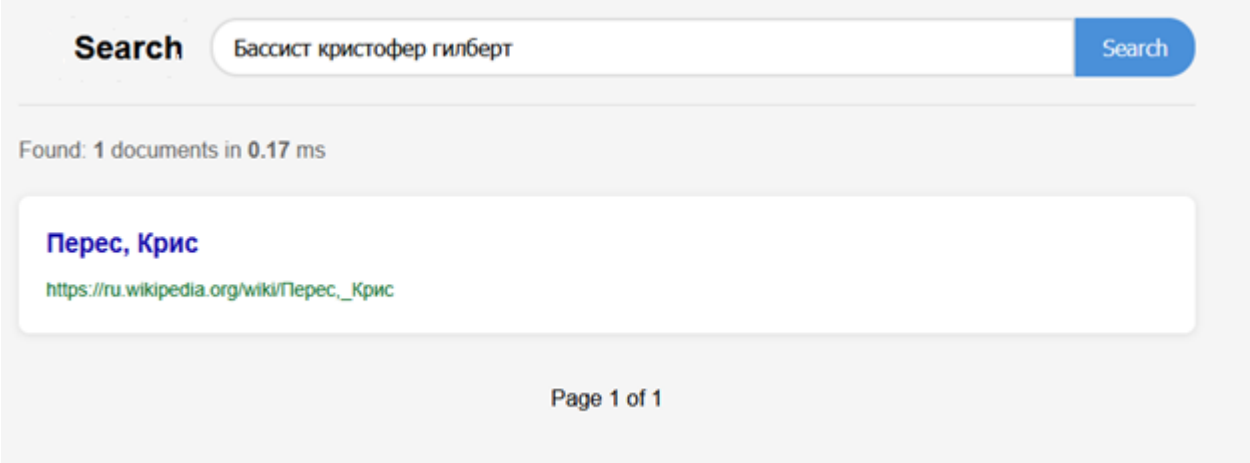


Рисунок 7 – результаты поиска

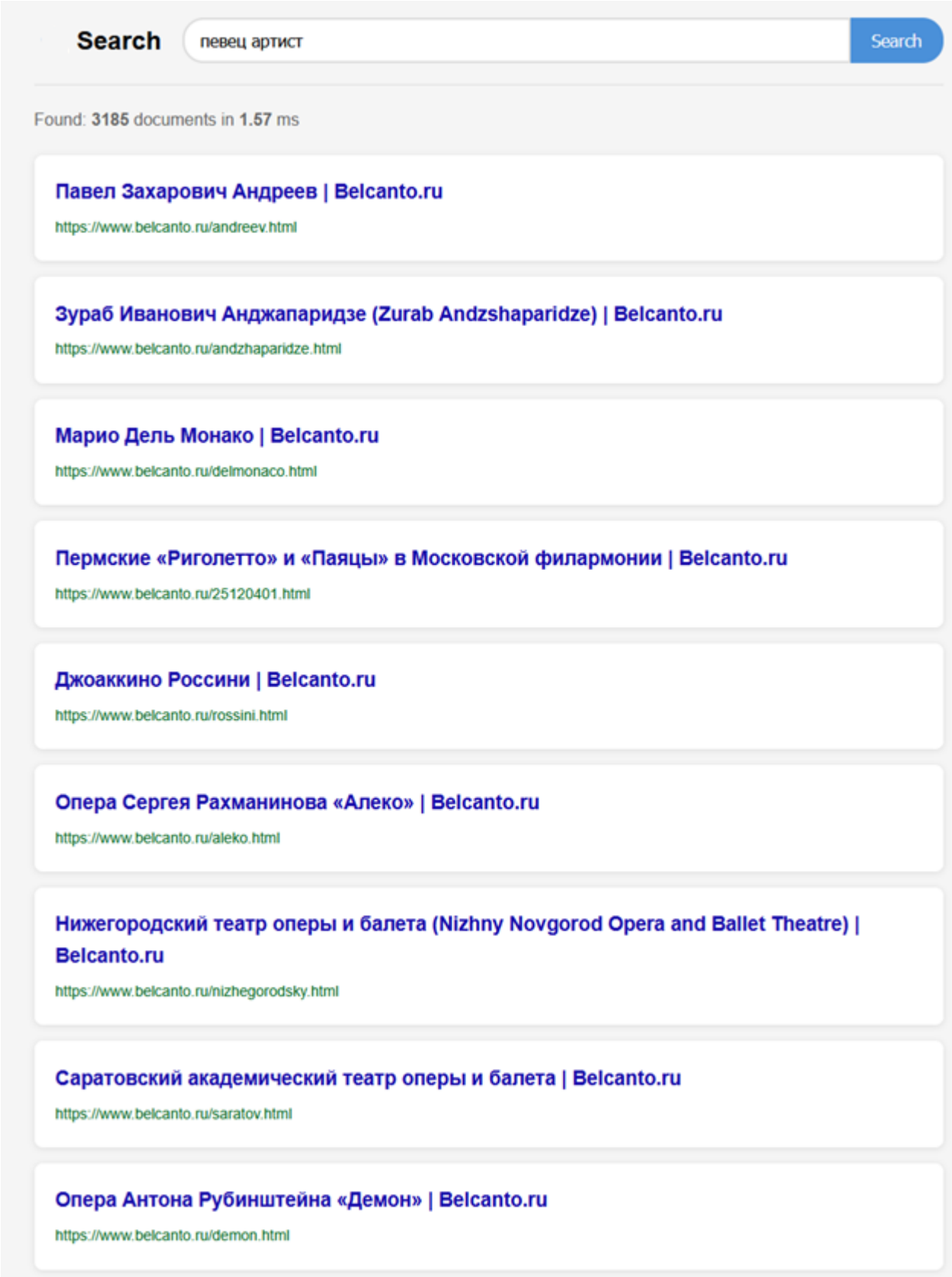


Рисунок 8 - результаты поиска «певец артист»

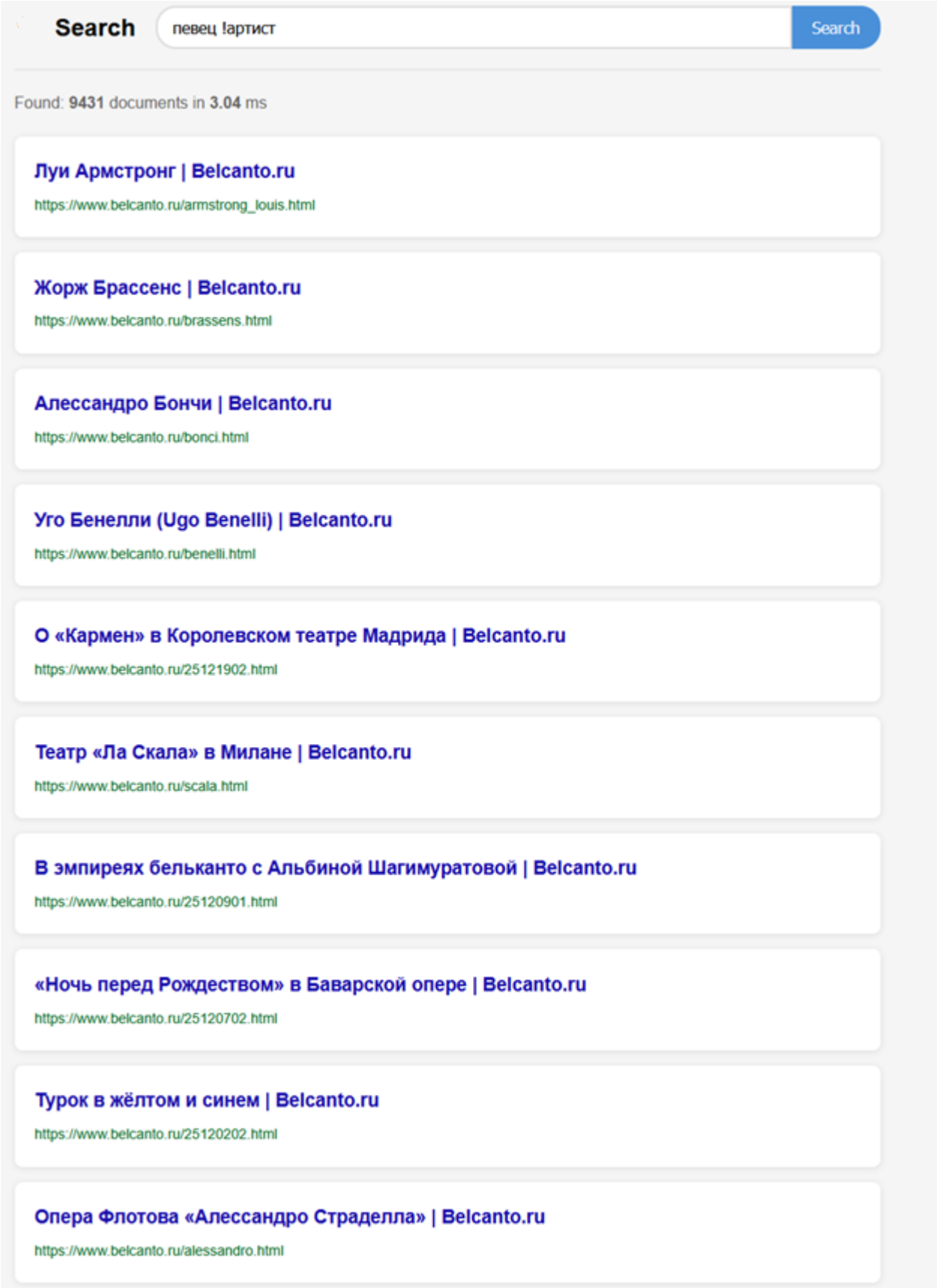


Рисунок 9 – результаты поиска «певец !артист»

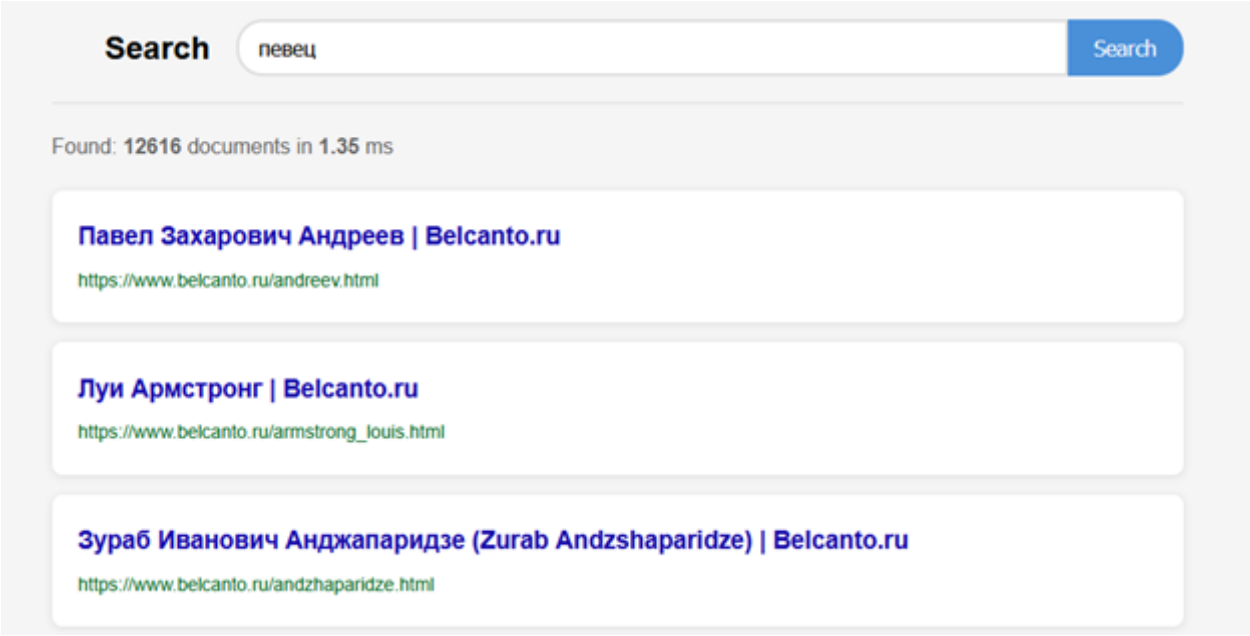


Рисунок 10 – результаты поиска «певец»

## Выводы

В процессе выполнения лабораторных работ были изучены процесс работы современных поисковых систем, процесс индексации. Была реализована своя поисковая система для конкретного корпуса данных. Для ее создания выполнялось множество последовательных действий: определение корпуса данных, дальнейшую его очистку и нормализацию, токенизацию и стемминг, построение прямого и обратного индекса. Реализация булева поиска, а также простого web-интерфейса.