МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ)

**Институт №8**
**«Компьютерные науки и прикладная математика»**

# Лабораторные работы по курсу «Информационный поиск»

Выполнила: Белякова С. А.
Группа: М8О-407Б-22
Преподаватель: А. А. Кухтичев

**Москва, 2025**

# Лабораторная работа №1
# Предварительная подготовка текстового корпуса

## Условия

В данной лабораторной работе выполняется предварительная подготовка текстового корпуса, предназначенного для дальнейшего использования в задачах информационного поиска. Работа охватывает полный цикл первичной обработки документов и включает получение исходных данных, анализ их структуры и характеристик, а также формирование набора текстов, пригодных для последующей автоматической обработки.

Корпус документов загружается из открытых источников и сохраняется на локальном носителе с обязательным указанием происхождения данных. Проводится анализ структуры документов, состава текстового содержимого и доступной метаинформации. Дополнительно рассматриваются существующие поисковые системы для выбранных источников и анализируются их ограничения. По итогам рассчитываются основные статистические характеристики корпуса.

## Описание

Целью лабораторной работы является формирование текстового корпуса, пригодного для использования в последующих этапах курса. Для формирования корпуса были выбраны два независимых открытых научных ресурса, ориентированных на публикации в области компьютерных наук.

**ACL Anthology** представляет собой специализированный архив научных публикаций по компьютерной лингвистике. Документы доступны по постоянным URL и представлены в виде HTML-страниц с заголовком и основным текстом.

**arXiv** является открытым репозиторием научных статей и препринтов. Каждая публикация имеет уникальный идентификатор и HTML-представление, содержащее метаданные и текст статьи.

После загрузки и предварительной обработки корпус включил 58 866 документов. Совокупный объём данных составил 529 704 878 байт. Средний размер одного документа — 8 998,49 байт, что соответствует примерно 9 КБ очищенного текста.

Сформированный корпус является тематически однородным и имеет достаточный объём для дальнейших экспериментов по анализу текста и информационному поиску.

## Исходный код

Для формирования корпуса использовался скрипт `dump_corpus.py`, взаимодействующий с базой данных MongoDB. Для каждого источника извлекалась последняя версия HTML-документа, из которой выделялись заголовок и видимый текст.

Очищенное текстовое содержимое сохранялось в виде отдельных файлов в каталоге `docs/`. Дополнительно формировался файл `meta.tsv`, содержащий URL, источник данных, время загрузки, заголовок и длину текста. При необходимости исходный HTML сохранялся в сжатом виде.

## Выводы

В результате выполнения лабораторной работы был сформирован текстовый корпус на основе двух независимых открытых научных ресурсов — ACL Anthology и arXiv. Корпус приведён к унифицированному виду и может быть использован для последующих лабораторных работ по построению поисковых индексов и анализу качества поиска.

# Лабораторная работа №2
# Поисковый робот

## Условия

В рамках лабораторной работы требовалось реализовать поисковый робот для автоматизированной загрузки документов из сети Интернет. Программа принимает на вход путь к YAML-конфигурации, содержащей параметры подключения к базе данных и настройки логики обхода.

Результаты работы сохраняются в MongoDB и включают нормализованный URL, исходный HTML-документ, источник данных и время загрузки в формате Unix timestamp. Робот должен корректно останавливаться и возобновлять работу, а также выполнять повторную загрузку страниц только при изменении их содержимого.

## Описание

Целью работы являлась разработка устойчивого поискового робота для длительной обкачки документов из открытых источников. Управление логикой обхода осуществляется через YAML-конфигурацию, что позволяет изменять параметры без модификации кода.

Для обеспечения устойчивости всё состояние обхода хранится в базе данных, включая очередь URL и время последней загрузки. Реализована проверка изменений содержимого страниц, предотвращающая избыточную переобкачку.

## Исходный код

Робот реализован на языке Python с использованием стандартных библиотек для HTTP-запросов, работы с YAML и MongoDB. Архитектура ориентирована на хранение состояния в базе данных, что обеспечивает масштабируемость и корректное возобновление работы.

## Выводы

В ходе выполнения работы был реализован поисковый робот, удовлетворяющий требованиям задания. Решение поддерживает корректную остановку, возобновление и повторную загрузку документов только при изменении их содержимого.

# Лабораторная работа №3
# Токенизация текстов документов

## Условия

В рамках лабораторной работы требовалось реализовать процесс разбиения текстов документов на токены, которые в дальнейшем используются при индексации и поиске. Необходимо было определить правила токенизации, подробно описать их в отчёте, проанализировать достоинства и ограничения выбранного подхода, а также привести примеры неудачно выделенных токенов и возможные способы корректировки правил.

В результатах работы требовалось представить статистические характеристики процесса токенизации, включая общее количество токенов и их среднюю длину. Дополнительно необходимо было измерить время выполнения программы, описать зависимость времени работы от объёма входных данных и вычислить скорость токенизации в пересчёте на килобайт текста.

## Описание

Целью данной лабораторной работы являлась реализация токенизации текстов документов корпуса с последующим использованием полученных токенов при построении индексов и выполнении поисковых запросов. Токенизация выполняется в виде последовательного прохода по тексту документа, в ходе которого формируются токены на основе заранее заданных правил.

В процессе обработки применяется нормализация текста, включающая приведение символов к нижнему регистру и унификацию отдельных вариантов написания с учётом особенностей русского языка. Дополнительно предусмотрен режим со стеммингом, позволяющий сократить количество различных словоформ и упростить дальнейшую обработку текста.

## Правила токенизации

Токен определяется как непрерывная последовательность буквенных или цифровых символов, включающая символы кириллицы, латиницы и цифры. В качестве разделителей используются пробельные символы и знаки пунктуации.

Перед добавлением токена выполняется понижение регистра и нормализация отдельных символов, в частности замена буквы «ё» на «е». Для корректной обработки научных и технических текстов допускается сохранение дефиса внутри токена, что позволяет не разрушать составные обозначения, такие как версии программ или устойчивые термины. Токены, длина которых меньше заданного минимального порога, отбрасываются для уменьшения шума.

## Результаты и статистика

Токенизация была выполнена на полном корпусе, включающем 58 866 документов. Общий объём входного текста составил 529 704 878 байт. Суммарное количество выделенных токенов превысило 70 миллионов, а средняя длина одного токена составила около шести символов.

## Время выполнения и производительность

Время работы программы токенизации на всём корпусе составило около 6,4 секунды при включённом режиме стемминга. Скорость обработки составила приблизительно 81 тысячу килобайт в секунду, что соответствует обработке порядка 11 миллионов токенов в секунду. Такая производительность близка к практическому пределу для линейной токенизации и в основном ограничивается затратами на обработку Unicode и операции ввода-вывода.

## Выводы

В ходе лабораторной работы была реализована токенизация корпуса документов с едиными правилами нормализации текста. Были получены статистические показатели, характеризующие количество токенов, их среднюю длину и производительность алгоритма. Также были выявлены направления для дальнейшего улучшения качества токенизации.

# Лабораторная работа №4
# Лемматизация и стемминг

## Условия

В рамках лабораторной работы требовалось дополнить разработанную поисковую систему механизмом лемматизации, реализованным в виде стемминга. В простейшем варианте поиск должен выполняться без учёта словоформ. В более расширенном варианте допускается использование смешанного подхода с приоритетом точных совпадений.

Стемминг может применяться как на этапе построения индекса, так и при обработке поискового запроса. Необходимо было оценить влияние данного механизма на качество поиска и проанализировать случаи его ухудшения.

## Описание

Целью лабораторной работы являлось добавление в поисковую систему этапа стемминга, позволяющего снизить зависимость поиска от конкретных словоформ. Применение стемминга повышает полноту поиска и позволяет находить релевантные документы при различии форм слов в тексте и запросе.

Для корректной работы поисковой системы одинаковые правила обработки применяются как при индексации документов, так и при обработке пользовательских запросов.

## Журнал выполнения и проблемы

Стемминг был добавлен в цепочку обработки текста как отдельный этап, следующий за токенизацией и нормализацией. В процессе тестирования было выявлено, что в ряде случаев разные по смыслу слова могут сводиться к одной основе, что приводит к появлению нерелевантных документов в выдаче.

Особые сложности возникают при обработке технических токенов, содержащих цифры или дефисы. Для таких случаев рекомендуется либо отключать стемминг, либо применять специальные правила обработки.

## Выводы

В результате выполнения лабораторной работы в поисковую систему был внедрён механизм стемминга, применяемый согласованно на этапах индексации и обработки запросов. Это позволило повысить полноту поиска, однако выявило случаи ухудшения качества, требующие дополнительных механизмов ранжирования.

# Лабораторная работа №5
# Закон Ципфа

## Условия

Для подготовленного корпуса документов требовалось построить распределение терминов по частотам и визуализировать его в логарифмических координатах. Поверх эмпирического распределения необходимо было наложить теоретическую кривую, соответствующую закону Ципфа, и проанализировать степень совпадения.

## Описание

Для каждого терма корпуса вычислялась частота его встречаемости, после чего термы сортировались по убыванию частоты и каждому из них присваивался ранг. На основе полученных данных строился график зависимости частоты терма от его ранга в логарифмической шкале.

## Результаты и анализ

Полученное распределение демонстрирует характерную для текстовых корпусов структуру. Небольшое число наиболее частотных терминов формирует «голову» распределения, за которой следует участок с близким к линейному поведением в log-log координатах.

Отклонения от теоретической модели объясняются неоднородностью корпуса, наличием шумовых терминов и влиянием правил токенизации и стемминга.

## Выводы

Анализ показал, что распределение терминов в корпусе в целом соответствует закону Ципфа, при этом наблюдаемые отклонения объяснимы свойствами реальных текстовых данных.

# Лабораторная работа №7
# Построение булевого индекса

## Условия

В рамках лабораторной работы требовалось реализовать булев индекс по подготовленному корпусу документов. Индекс должен обеспечивать выполнение логического поиска и быть реализован в собственном бинарном формате хранения.

## Описание

Был построен обратный индекс, сопоставляющий каждому терму список идентификаторов документов, в которых он встречается. Дополнительно был создан прямой индекс, позволяющий по идентификатору документа восстанавливать его URL и заголовок.

## Журнал выполнения

Основной сложностью стало ограничение на использование стандартных ассоциативных контейнеров. Для решения задачи была применена схема внешней сортировки, при которой пары «терм–doc_id» сначала сохранялись во временные файлы, а затем сортировались и сливались.

## Выводы

В результате выполнения лабораторной работы был построен булев индекс по корпусу из 58 866 документов. Полученная структура данных допускает дальнейшее расширение и развитие.

# Лабораторная работа №8
# Реализация булевого поиска

## Условия

Требовалось реализовать механизм булевого поиска поверх ранее построенного обратного индекса. Поисковая система должна корректно обрабатывать логические операторы AND, OR, NOT и скобки.

## Описание

Поисковый компонент реализован в виде консольной утилиты, загружающей бинарные файлы индекса и выполняющей обработку пользовательского запроса. Запросы нормализуются по тем же правилам, что и документы при индексации.

## Реализация

Для корректной интерпретации логических выражений запрос преобразуется в постфиксную форму, после чего вычисляется с использованием стека. Операции конъюнкции, дизъюнкции и отрицания реализованы как операции над отсортированными списками идентификаторов документов.

## Выводы

В ходе выполнения лабораторной работы был реализован механизм булевого поиска, корректно работающий на полном корпусе документов. Полученное решение является основой для дальнейшего расширения поисковой системы.

## Выводы

В ходе работы я разобралась, как проектировать собственный токенизатор под русский и английский тексты, измерять его производительность и оценивать качество через Zipf.

Получила опыт построения частотных словарей и визуализации распределения терминов, увидела типичные отклонения реального корпуса от теоретической модели.

Освоила простые стемминг-правила и интегрировала их в индексатор и поиск, оценив влияние на размер словаря и результаты.

Практика построения булевого индекса и выполнения логических операций дала понимание низкоуровневых структур поиска.