Московский Авиационный Институт
(Национальный исследовательский университет)

Институт №8 «Информационные технологии и прикладная математика»
Кафедра вычислительной математики и программирования

Лабораторная работа №3

по курсу «Информационный поиск»

Студент: Зайцев Н.В.
группа М8О-208М-20

Преподаватель: Кухтичев А.А.

Москва, 2021

Лабораторная работа № 3

Требуется построить поисковый индекс, пригодный для булева поиска, по подготовленному в ЛР1 корпусу документов.

Требования к индексу:

- самостоятельно разработанный, бинарный формат представления данных. Формат необходимо описать в отчёте, в побайтовом представлении;
- формат должен предполагать расширение, т.к. в следующих работах он будет меняться под требования новых лабораторных работ;
- использование текстового представления или готовых баз данных не допускается;
- кроме обратного индекса, должен быть создан «прямой» индекс, содержащий в себе как минимум заголовки документов и ссылки на них (понадобятся для выполнения ЛР4, при генерации страницы поисковой выдачи);
- для термов должна быть как минимум понижена капитализация.

Ход работы

В работе были использованы WinAPI для поиска файлов и их перекодирования и хеш-таблица. Папка с документами располагается на два уровня выше файлов программы (по умолчанию в папке docs).

В файлах Tokenizing.h и Tokenizing.cpp происходит разбиение текста на токены. Благодаря структуре Location вся информация по файлу хранится в одном месте – номер файла в общем списке, номер токена в файле, номер строки и номер символа в строке. Поскольку основная часть текста статей на русском, используются только русские слова, а прочие символы – цифры, латиница, греческий и символы кодировки – считаются разделителями. Для удобства анализа исходная кодировка UTF-8 переводится в UCS-2 и вводится функция понижения регистра и определения символа как кириллицы в соответствии с таблицей Юникод-символов. Перекодировка описана в файлах Encoding.h и Encoding.cpp с использованием обвертки над функциями WinAPI.

В файлах Storage.h и Storage.cpp создается класс хранилища, где будут храниться индексы, токены и их положение в файлах статей. Используются процессы сериализации и десериализации для обработки файлов, а также сниппеты для создания описания результатов поиска.

В файле Utils.h модифицируются алгоритмы, активно используемые в STL, под данную задачу – например, функции swap, fill.

В файлах UserVector.h, UserString.h и UserList.h создаются рукописные контейнеры вектора, строки и списка, используемые затем при создании хеш-таблицы в файле UserHashTable.h. В качестве хеш-функции была выбрана функция murmurhash2 – простая и быстрая хеш-функция с хорошим распределением, возвращающая 32-разрядное беззнаковое число. Она описана в файлах murmur_hash2.h и murmur_hash2.cpp.

В файлах FS.h и FS.cpp прописаны рукописные обвертки над функциями WinAPI, а именно флаги поиска, поиск файлов в папке, объединение путей, абсолютный путь, существование файла или папки, чтение файла, вывод информации, а также класс файла для чтения или записи. В файлах Exception.h и Exception.cpp прописаны все возможные исключения и ошибки с их кодами.

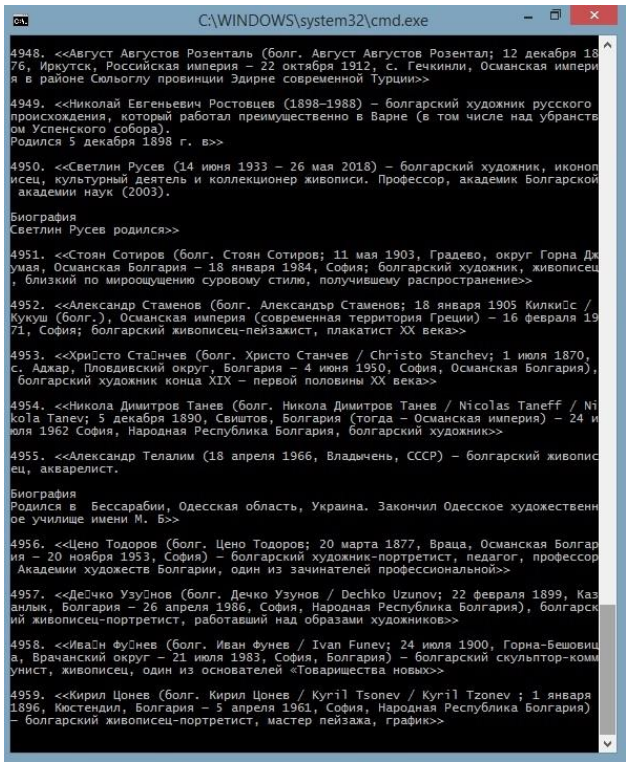
В файле Global.h прописанные шаблоны вектора, строки и списка сравниваются с STL-контейнерами. Это сделано для проверки работы рукописных контейнеров.

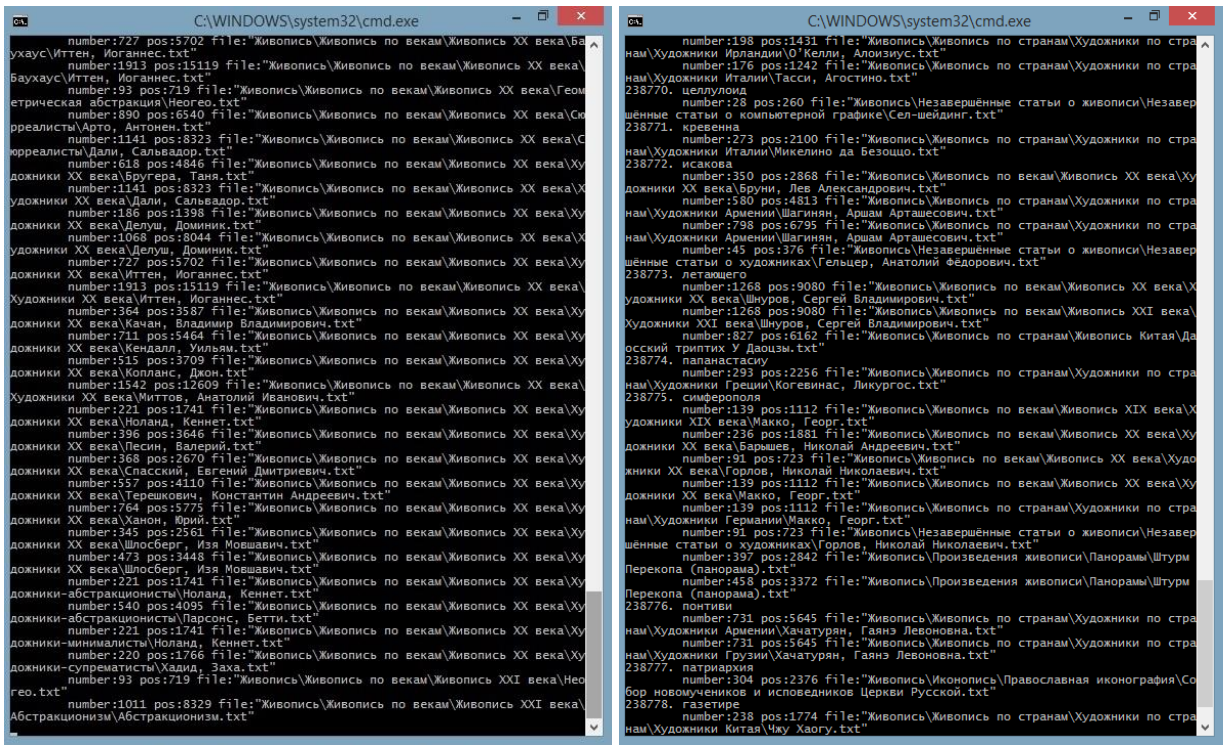
В результате вся работа программы сводится к следующему:

- меняется кодировка;
- начинается поиск файлов, происходит получение полного пути до документов;
- данные файлов разбиваются на токены и сохраняются в бинарный файл индекса;
- далее происходит загрузка полученных данных из файла и вывод токенов в консоль.

Программа запускается в двух режимах через файлы формата bat:

- «Обработка и сохранение» – происходит анализ файлов, строится база данных по поиску и сохраняется в файл index.binary.
- «Подгрузка из файла и вывод» – из полученной базы данных происходит выгрузка найденных слов в консоль.





Статистическая информация

Размер индекса – 69724 Кб.

Время создания индекса базы данных – 1252 секунды.

Время загрузки индекса базы данных – 934 секунды.

Для ускорения работы можно попробовать добавить в хеш-таблицу вектор с указателями на все пары типа «ключ-значение» (в данной лабораторной номер токена и сам токен), чтобы при сохранении полученной базы данных не проверять все ячейки таблицы на пустоту.

Описание файла бинарного формата

Файл состоит из трех последовательных частей – имен документов, их описания и токенов. На один элемент каждой части выделяется по 4 байта. Списки документов, их описания и токены хранятся в контейнерах, сами названия файлов, сниппетов и токенов сохранены в виде строк. Кроме того, для информации по каждому токену используется структура, в которую входят индекс файла, номер токена и его позиция в файле; на каждую часть структуры выделяется по 4 байта.

Список файлов		
Имя файла		
Длина строки, 4 байта	Один символ Юникод в строке, 2 байта	...

Список описаний файлов		
Текст описания		
Длина строки, 4 байта	Один символ Юникод в строке, 2 байта	...

Список токенов		
Токен в виде строки		
Длина строки, 4 байта	Один символ Юникод в строке, 2 байта	...
Структура с информацией о токенах		
Индекс файла, 4 байта	Номер токена в файле, 4 байта	Позиция в файле, 4 байта

Вывод

В ходе выполнения лабораторной работы был рассмотрен вариант индексирования корпуса документов. В качестве хеш-функции была выбрана функция murmurhash2 как наиболее подходящая к задаче при заданных условиях. Также были самостоятельно написаны все необходимые STL-контейнеры.