1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1.1 Введение

Составленное техническое задание по дисциплине «Алгоритмические основы обработки данных» является документом к курсовой работе, который отражает все этапы разработки программного продукта, а также процесс проектирования и выявление требований, предъявляемых конечному продукту.

1.1.1 Наименование программы

Название данного приложения “Работа со словарем” будет напрямую связываться с темой курсовой работы “Словарь”. Словарь – это индексируемая структура данных, доступ к элементам которой выполняется только по индексу (ключу). Элемент словаря состоит из ключаи связанного с ключом значения. Пример словаря: русско-английский словарь, элементы которого содержат слово на русском языке (ключ) и перевод слова на английский язык, например, {[список], list}. Для хранения словаря в оперативной памяти использовать динамический массив.

1.1.2 Краткая характеристика области применения программы

Программа предназначена для создания и изменения словаря, индексации по нему. Это приложение будет полезно для всех, кто работает с большим объемом информации.

1.2 Основание для разработки

Основанием для разработки является курсовая работа по дисциплине « Алгоритмические основы обработки данных», предусмотренная учебным планом направления подготовки 09.03.01 «Информатика и вычислительная техника» профиля «Цифровые комплексы, системы и сети».

1.3 Назначения разработки

Приложение может помочь пользователям, использующим любой сервис чтения в Интернете, для быстрого поиска информации удобным способом.

1.4 Требования, предъявляемые к программе

1.4.1 Требования к функциональным характеристикам программы

В приложении должны быть реализованы следующие операции:

- создание пустого словаря;
- добавление элемента в словарь;
- исключение элемента из словаря;
- поиск элемента словаря по ключу;
- изменение значения элемента;
- вывод словаря в порядке возрастания ключей.

1.4.2 Требования к техническим средствам, используемым при работе программы

Персональный компьютер пользователя должен быть оснащён графическим адаптером, также должна быть установлена ОС Windows (не ниже Windows 7).

1.4.3 Требования к языкам программы и среде разработки программы

Для разработки используется язык программирования C++, в качестве среды разработки выступает Visual Studio. Для разработки интерфейса пользователя задействован механизм Visual Studio.

1.4.4 Требования к информационным структурам на входе и выходе программы

В качестве входных данных программа принимает файл, содержащий словарь вида “Номер телефона - ФИО”

Выходные данные представляют собой файл, содержащий словарь вида “Номер телефона - ФИО”

1.5 Требования к программной документации

1. Пояснительная записка оформляется в соответствии с ЛНА РТУ МИРЭА.
2. Проектная документация, составленная в соответствии с ГОСТ.

В процессе создания приложения вся проделанная работа документируется, должны быть сохранены все детали разработки, а также трудности, с которыми пришлось столкнуться. Всё вышеперечисленное должно быть отражено в пояснительной записке, которая прилагается к работе.

1.6 Этапы разработки

1. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций 02.09.2024- 22.09.2024.
2. Разработка программы: 22.09.2024-30.11.2024.
3. Разработка программной документации: 01.12.2024-10.12.2024.
4. Оформление пояснительной записки: 11.12.2024-16.12.2024.
5. Защита курсовой работы: 23.12.2024-30.12.2024

2 ОБЗОР СПОСОБОВ ОРГАНИЗАЦИИ ДАННЫХ И ОБОСНОВАНИЕ ВЫБОРА СТРУКТУРЫ ДАННЫХ ДЛЯ ЭФФЕКТИВНОГО ВЫПОЛНЕНИЯ ОПЕРАЦИЙ

Данные могут быть организованы различными способами. Тип структуры данных в программе оказывает большое влияние на ее производительность. Для того чтобы выбрать наиболее простой и эффективный способ организации данных в программе рассмотрим несколько типов структур данных.

2.1 Бинарное дерево поиска

Бинарное дерево поиска - это бинарное дерево (бинарное дерево — это иерархическая структура данных, в которой каждый узел имеет значение, оно же является в данном случае и ключом, и ссылки на левого и правого потомка.), обладающее дополнительными свойствами: значение левого потомка меньше значения родителя, а значение правого потомка больше значения родителя для каждого узла дерева. То есть, данные в бинарном дереве поиска хранятся в отсортированном виде. При каждой операции вставки нового или удаления существующего узла отсортированный порядок дерева сохраняется. При поиске элемента сравнивается искомое значение с корнем. Если искомое больше корня, то поиск продолжается в правом потомке корня, если меньше, то в левом, если равно, то значение найдено и поиск прекращается.

2.2 Словарь

Словарь (ассоциативный массив) – это индексируемая структура данных, доступ к элементам которой выполняется только по индексу (ключу). Элемент словаря состоит из ключа и связанного с ключом значения. Словарь поддерживает добавление пар ключ-значение, получение значения по ключу, удаление пары ключ-значение, получение размера словаря, проверка словаря на пустоту. Имеет быстрый

доступ к данным по ключу, а также гибкость хранения данных разного типа. Но занимает больше памяти, чем простые списки, и не гарантирует упорядоченность элементов.

2.3 Хэш-таблица

Хэш-таблица - структура данных, реализующая интерфейс ассоциативного массива, а именно, она позволяет хранить пары (ключ, значение) и выполнять три операции: операцию добавления новой пары, операцию удаления и операцию поиска пары по ключу. Она реализуется с использованием хэш-функции, которая преобразует ключ в индекс массива (или другой структуры), где хранится значение. Позволяет быстро искать, добавлять и удалять данные, так как операции выполняются за O(1). Но при большом количестве данных могут возникать коллизии (ситуация, когда разные ключи дают одинаковый хэш). Также требует больше памяти

2.4 Очередь

Очередь – динамическая структура данных, работающая по принципу FIFO – first in, first out (первый вошел, первый вышел). Элементы добавляются в конец очереди, и оттуда же извлекаются. Позволяет добавить элемент в конец очереди, удалить элемент в конце очереди, получить доступ к концу и началу очереди, получить размер массива. Очередь часто используется в алгоритмах для организации порядка выполнения задач. Полезна в алгоритмах, где требуется упорядоченная обработка данных. Неэффективна для поиска и случайного доступа, поскольку доступ к элементам идет строго в порядке добавления.

2.5 Массив

Массив — одна из самых простых и часто применяемых структур данных. Другие структуры данных, к примеру, стеки и очереди, являются производными от

массивов. Сами массивы используются для обработки большого количества однотипных данных.

Элементы в массиве имеют свой индекс — номер элемента, по которому можно производить поиск. Существует два типа массивов: одномерные и многомерные. Первые представляют собой простейшие линейные структуры, которые называются вложенными и включают другие массивы. Чаще всего в программировании используются одномерные и двумерные массивы.

Также существуют динамические массивы, для выделения работы с ними используются отдельные формы операторов new и delete: new[] и delete[]. Динамическое выделение массива позволяет устанавливать его длину во время выполнения программы.

2.6 Выбор структуры данных

В ходе использования программы пользователь вводит в консоль текст, либо вводит название текстовых файлов, поэтому нужно воспользоваться типом данных string (строковый).

Чтобы реализовать хранение данных в файле, необходимо использовать файловый поток fstream для чтения, записи и хранения файла.

Для хранения паспортных данных и ФИО, напишем свою структуру данных, используя типы данных int и string.

Для хранения структур из ФИО и паспортных данных используем map, т.к именно словарь нам надо реализовать.