Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

КУРСОВОЙ ПРОЕКТ

Дисциплина: Алгоритмы и структуры данных

Тема: Разработка хеш-таблицы с двойным хешированием.

Выполнил
студент гр. 3530901/90003 _____ Бехтольд Е.В.
(подпись)

Руководитель _____ Ахин М.Х.
(подпись)

«___» _____ 2020 г.

Санкт-Петербург
2020

Оглавление

Техническое задание ……..3

Метод решения ……...4

Скриншоты программы ……6

Литература ……10

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Реализовать в виде класса на Kotlin хеш-таблицу с двойным хешированием. Класс таблицы должен реализовывать интерфейс mutableMap, Kotlin в обобщённом виде. Класс должен обеспечивать возможность неограниченного роста размера таблицы (в пределах памяти компьютера). Дополнить реализацию графическим представлением.

GitHub репозиторий проекта: https://github.com/KathyBekh/DoubleHashMap.git

МЕТОД РЕШЕНИЯ

В проекте использована концепция MVC (Model-View-Controller) для отделения бизнес-логики от визуализации. Код разделен на три пакета классов: controller, model, view.

Содержимое пакета model:

- DoubleHashingMap — класс реализует интерфейс mutableMap, Kotlin.
- Entry — определяет свойства объектов класса DoubleHashinfMap.

Содержимое пакета controller:

- DoubleHashingMapController - создаёт объект DoubleHashingMap для работы в графическом редакторе и реализует безопасное использование методов DoubleHashingMap.

Содержимое пакета view:

- DoubleHashingMapApp — класс TornadoFX, запускает класс DoubleHashinfMapView.
- DoubleHashinfMapView — отвечает за отображение хеш-таблицы на экране.
- TableProperty — задаёт свойства для корректного отображения таблицы.
- HelpWindow – генерирует окно с описанием графического функционала хеш-таблицы.

Более подробная работа программы:

При запуске программы DoubleHashingMapController() создает объект класса DoubleHashingMap и обеспечивает взаимосвязь логики и отображения.

На рабочем экране справа отображаются поля для ввода «key» и «value», куда вводятся значения ключей и значений для дальнейших манипуляций с ними. Под полями ввода располагаются функциональные кнопки: ADD, FIND, DELETE, LOAD, CLEAR, HELP.

Пользователь может добавить новую пару в таблицу, заполнив поля для ввода и нажав на кнопку ADD, или загрузить файл формата .txt, нажав на кнопку LOAD. Доступные форматы файлов ограничены для выбора пользователем. Данные файла представляют собой строки пар «ключ — значение» разделенные знаком « - ». Строки не удовлетворяющие этому условию не отображаются в рабочем окне.

Удаление пары осуществляется при заполненном текстовом поле «key» и нажатии на кнопку DELETE. Очистить таблицу можно через кнопку CLEAR. Данные удаляются из таблицы безвозратно. FIND — осуществляет поиск по ключу. При нажатии на кнопку HELP появляется окно с описанием функционала кнопок.

При удалении пары и очищении таблицы появляются окна с подтверждением действий.

Пары размещаются в таблице в соответствии с концепцией двойного хеширования. Двойное хеширование (англ. double hashing) — метод борьбы с коллизиями, возникающими при открытой адресации, основанный на использовании двух хеш-функций для построения различных последовательностей исследования хеш-таблицы.

Для реализации данной концепции использовался следующий метод:

1. Вычисляется хеш-код ключа по функции firstHash().
2. Проверяется занята ячейка или нет, если нет то пара размещается в таблице, если да переходим к следующему шагу.
3. Вычисляется сдвиг — это хеш-код ключа по функции secondHash(). Суммируем хеш-код и сдвиг получаем новую ячейку для размещения пары.
4. Выполняем шаг 2. Если возникает коллизия то увеличиваем сдвиг на 1, пока не найдём свободную ячейку.

Все значения хеш-кодов берутся по модулю текущего размера таблицы. Функции firstHash() и secondHash() - взаимно простые, что обеспечивает равномерное распределение данных по таблице.

В логике программы разработаны методы в соответствии с интерфейсом mutableMap, языка программирования Kotlin. А именно:

- put() - сопоставляет указанное значение с указанным ключом в таблице. Возвращает: предыдущее значение, связанное с ключом, или ноль, если ключ не присутствовал в таблице.
- Remove() - удаляет указанный ключ и соответствующее ему значение из таблицы. Возвращает: предыдущее значение, связанное с ключом, или нулевое, если ключ не присутствовал в таблице.
- PutAll() - обновляет таблицу парами «ключ-значение» из указанной таблицы.
- Clear() - удаляет все элементы из таблицы.

А также методы и переменные интерфейса Map языка программирования Kotlin от которого наследуется интерфейс mutableMap.

- ContainsKey() и containsValue() - методы проверяют наличие ключа или значения в таблице.
- IsEmpty() - метод возвращает true, если таблица не содержит данных и false в ином случае.
- Size — возвращает количество пар «ключ-значение» в таблице.
- Keys — возвращает множество (set) ключей.
- Value — возвращает коллекцию (Collection) значений.
- Entries — возвращает множество (set) пар «ключ — значение».
- Get() - возвращает значение, соответствующее данному ключу, или null, если такого ключа нет в таблице.

Дополнительно к программе реализованы тесты. Тесты разделены на два класса: ControllerTest() и DoubleHashingMapTest(). Логика программы тестируется в классе DoubleHashingMapTest. ControllerTest тестирует работу контроллера.

СКРИНШОТЫ ПРОГРАММЫ

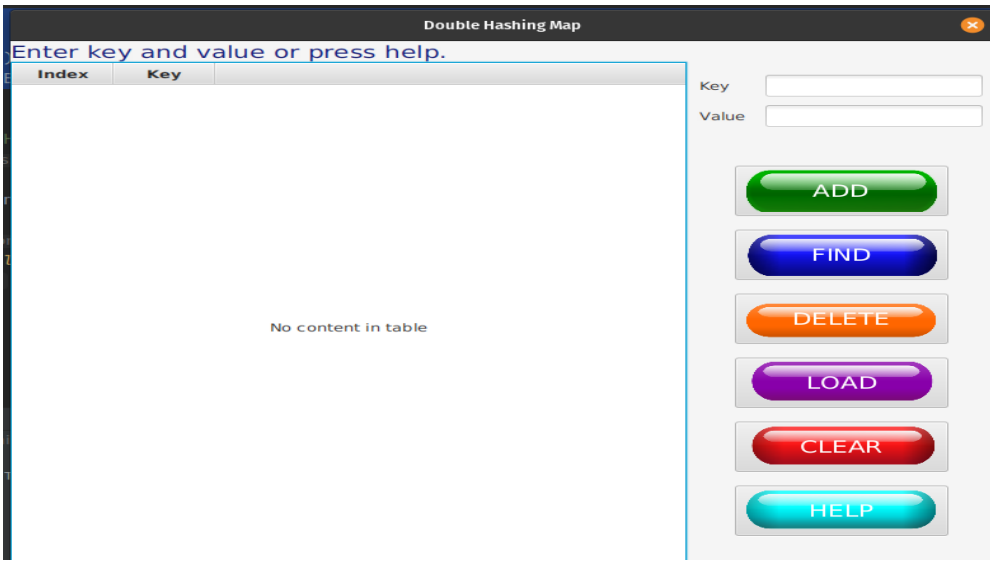

Рис. 1. Начальное рабочее окно.

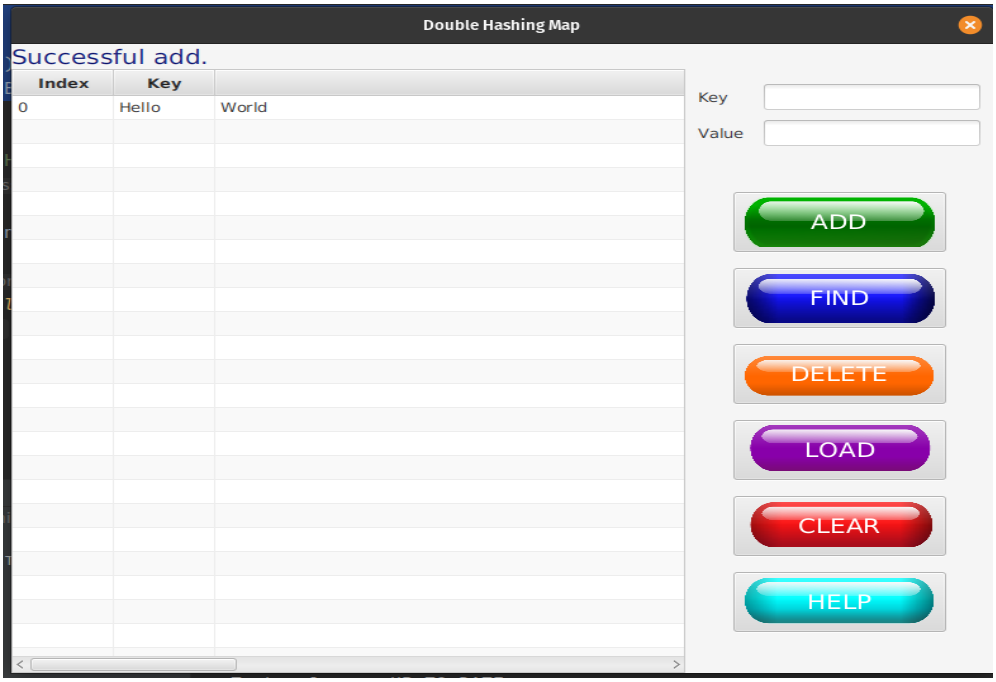

Рис. 2. Добавление пары.

Рис. 3. Добавление пар из файла.

Рис. 4. Окно-сообщение найденного элемента.

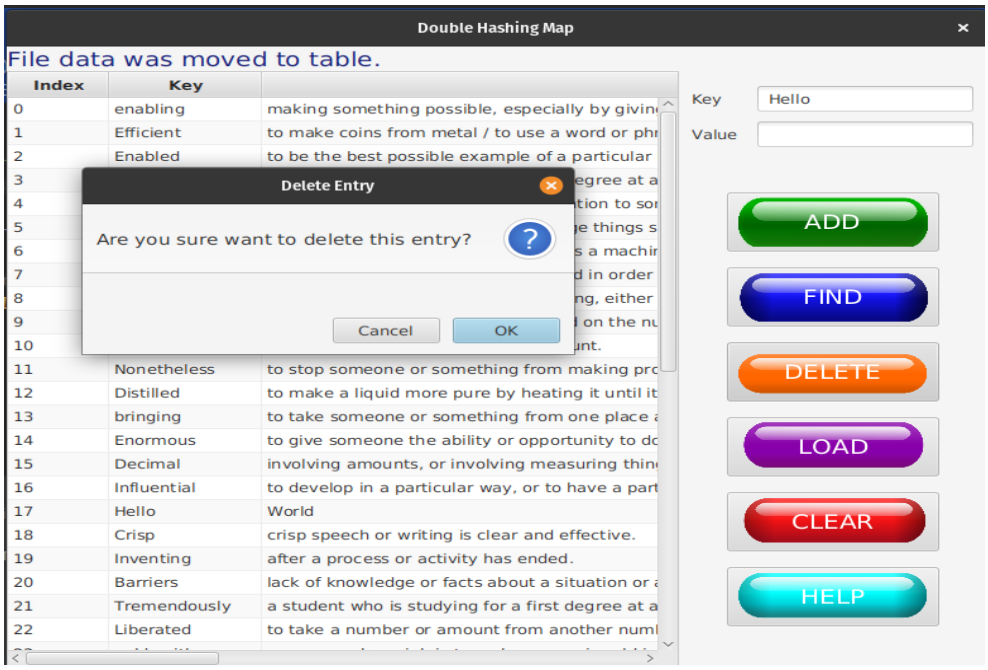

Рис. 5. Окно-сообщение удаления элемента.

Рис.6. Окно-сообщение об очищении таблицы.

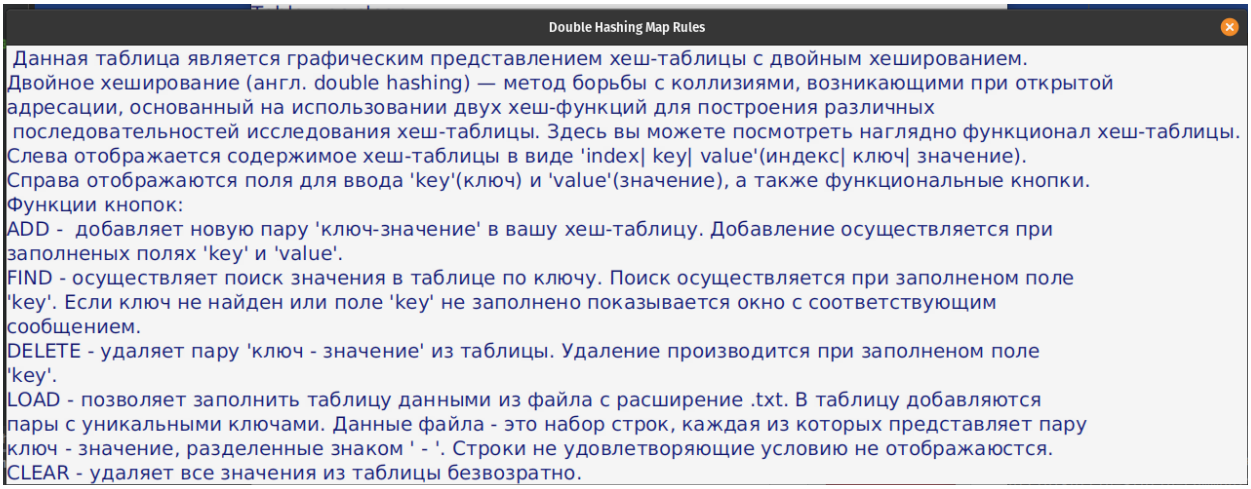

Рис.7. Окно с описанием работы программы.

ЛИТЕРАТУРА.

1. Описание языка программирования Kotlin: https://kotlinlang.org/docs/reference/
2. Описание framework TornadoFX: https://edvin.gitbooks.io/tornadofx-guide/content/
3. Описание реализации хеш-таблицы на языке Java от компании Oracle: https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html
4. Википедия: https://en.wikipedia.org/wiki/Double_hashing