Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления

Кафедра Интеллектуальных информационных технологий

**ОТЧЁТ**

по дисциплине «Аппаратное обеспечение интеллектуальных систем»

Лабораторная работа №6

Вариант 22

Выполнил: Самута Д. В.

гр. 221703

Проверил: Е. А. Казаченко

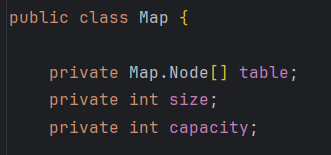
Минск 2024

**Тема:** Моделирование хеш-таблиц;

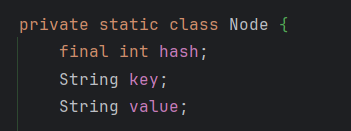
**Цель работы:** освоение навыков построения и проверки таблиц хеширования.

**Задание**: Линейное разрешение коллизий: квадратичный поиск

**Таблица**:



Хэш-таблица представляет из себя массив из Node, где Node – структура, содержащая ключ, значение и хэш-код.



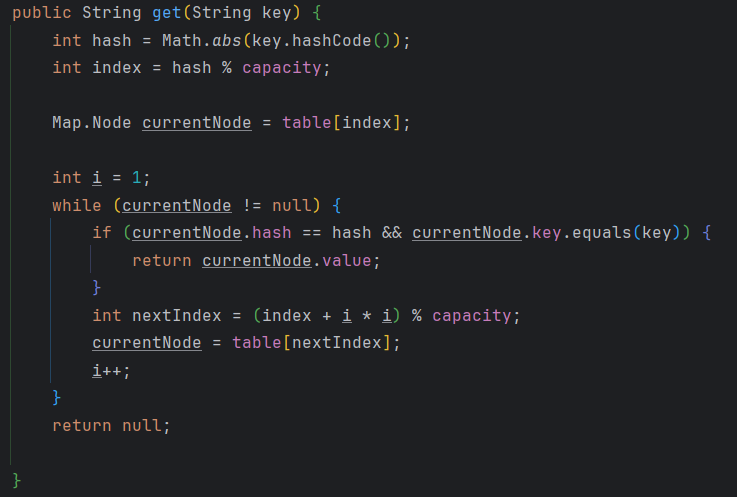
size – кол-во записей, добавленных на данный момент в массив.

capacity – размер table.

**Методы**:

Реализованы CRUD методы, а также метод вывода таблицы.

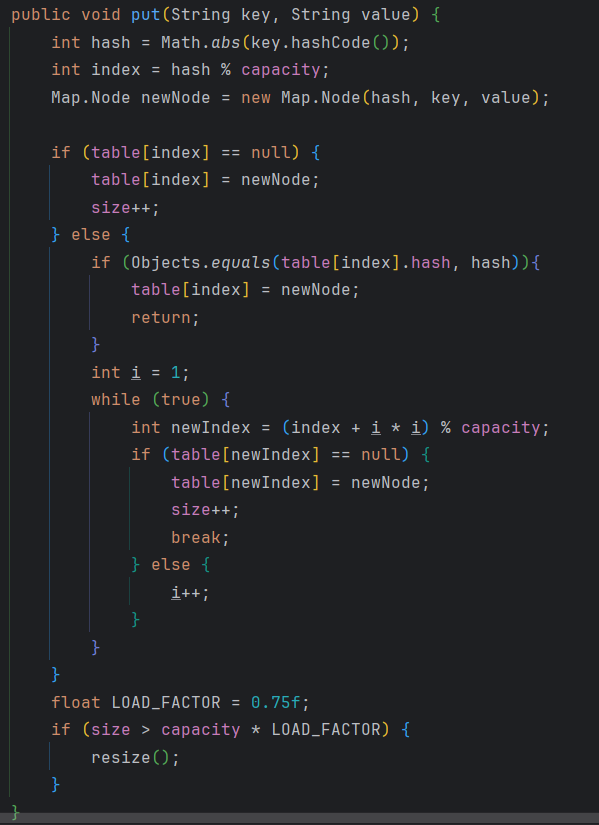
1. Получение значения по ключу



По полученному хэшу в самом начале вычисляется индекс в массиве, в котором может находится значение, в случае, если при добавлении не было коллизии. В случае возникновения коллизии при добавлении, она решается с помощью квадратичного поиска.

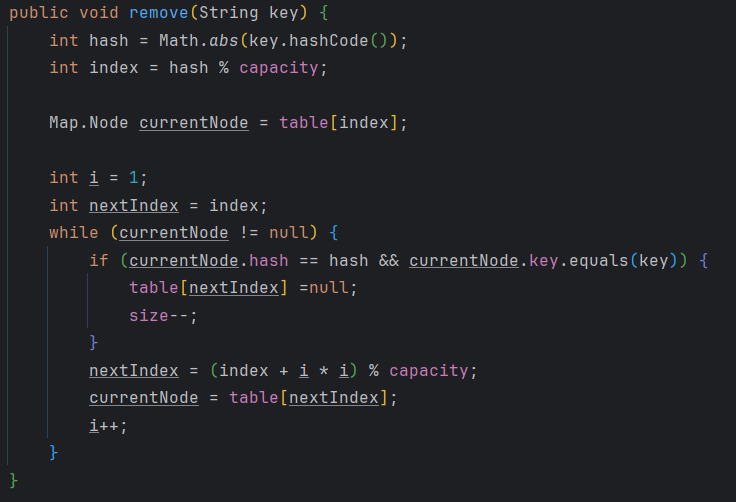
Суть метода: при добавлении элемента, если возникает коллизия (когда два ключа имеют одинаковый хэш), новый элемент помещается в ячейку, которая находится на определенном расстоянии от исходной позиции, и это расстояние увеличивается квадратично с каждой итерацией. При поиске элемента также используется квадратичный шаг, чтобы обеспечить обход всех ячеек в таблице.

1. Добавление по ключу/изменение значение по существующему ключу



При добавлении, если коллизии не возникает (т.е. по данному хэшу в таблице ничего нет) значение просто добавляется по этому индексу в массив. Если по данному индексу в массиве лежит значение, то проверяется, равны ли хэшы. Если равны, то происходит перезапись значения(т.е. происходит изменение значение по существующему ключу) . Если не равны, то происходит решение коллизии(т.е. добавление по ключу), пока не будет найдено свободное место в массиве. Также, если кол-во значений в массиве станет 0.75 размера массива, то произойдет увеличение размера массива в 2 раза.

1. Удаление по ключу

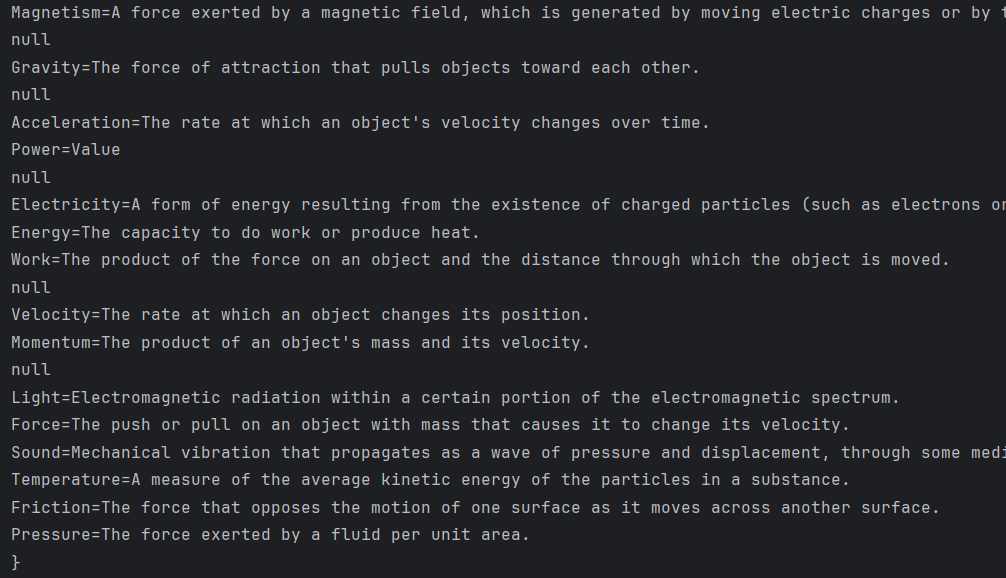


Аналогично, как и поиск, только при нахождении элемента по ключу по этому индексу в массиве ставится null.

1. Вывод таблицы истинности



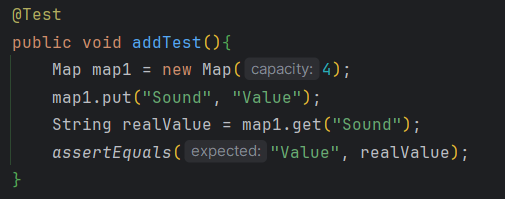
Пример вывода:



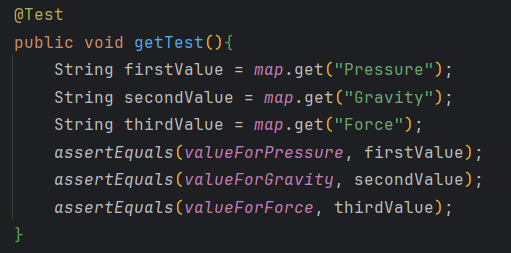
**Тестирование**:

Для каждого CRUD метода был написан тест:

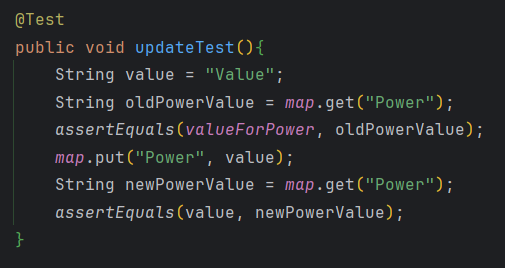
1. Добавление:



1. Получение по ключу:



1. Изменение по ключу:



1. Удаление по ключу:



**Вывод**: Таким образом, в результате выполнения лабораторной работы, была реализована такая структура данных, как хэш-таблица, имеющая свои преимущества и недостатки. Преимущества: поиск (в лучшем случае) осуществляется за O(n), что быстрее чем в таких структурах, как массив, бинарное дерево, граф и т.д; существует большое кол-во реализаций хэш-таблиц (способов решения коллизий). Недостатки: может случится такая ситуация, что хэши будут равны (это редко, однако это возможно); при заполнении какого-то процента массива, нужно создавать массив большего размера, и заново заполнять его значениями.