

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ
КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**
**Ордена трудового Красного Знамени федеральное государственное
бюджетное**
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математическая кибернетика и информационные технологии

Отчет по лабораторной работе №3.

Выполнил: студент группы БВТ2402

Безматерных Иван Алексеевич

Москва, 2025

Цель работы: изучить принципы работы хэш-таблиц как фундаментальной структуры данных и приобрести практические навыки их реализации. Освоить методы разрешения коллизий и научиться правильно переопределять методы equals() и hashCode() для корректной работы с пользовательскими объектами в хэш-таблицах.

Задание 1:

1. Создайте класс HashTable, который будет реализовывать хэш-таблицу с помощью метода цепочек.
2. Реализуйте методы put(key, value), get(key) и remove(key), которые добавляют, получают и удаляют пары «ключ-значение» соответственно.
3. Добавьте методы size() и isEmpty(), которые возвращают количество элементов в таблице и проверяют, пуста ли она.

```
import java.util.LinkedList;

public class HashTable<K, V> {

    private static class Entry<K, V> {
        private K key;
        private V value;

        public Entry(K key, V value) {
            this.key = key;
            this.value = value;
        }

        public K getKey() { return key; }
        public V getValue() { return value; }
        public void setValue(V value) { this.value = value; }
    }

    private LinkedList<Entry<K, V>>[] table;
    private int size;
    private static final int DEFAULT_CAPACITY = 16;

    @SuppressWarnings("unchecked")
    public HashTable() {
```

```

        table = new LinkedList<Entry<K, V>>[DEFAULT_CAPACITY];
        size = 0;
    }

    private int hash(K key) {
        return Math.abs(key.hashCode()) % table.length;
    }

    public void put(K key, V value) {
        int index = hash(key);

        if (table[index] == null) {
            table[index] = new LinkedList<Entry<K, V>>();
        }

        for (Entry<K, V> entry : table[index]) {
            if (entry.getKey().equals(key)) {
                entry.setValue(value);
            }
        }
    }

```

```

    }

    table[index].add(new Entry<K, V>(key, value));
    size++;
}

public V get(K key) {
    int index = hash(key);

    if (table[index] == null) {
        return null;
    }

    for (Entry<K, V> entry : table[index]) {
        if (entry.getKey().equals(key)) {
            return entry.getValue();
        }
    }

    return null;
}

public boolean remove(K key) {
    int index = hash(key);

    if (table[index] == null) {

```

```

        return false;
    }

    for (Entry<K, V> entry : table[index]) {
        if (entry.getKey().equals(key)) {
            table[index].remove(entry);
            size--;

            if (table[index].isEmpty()) {
                table[index] = null;
            }

            return true;
        }
    }

    return false;
}

public int size() {
    return size;
}

```

```

public boolean isEmpty() {
    return size == 0;
}

public void display() {
    System.out.println(x:"Содержимое хэш-таблицы:");
    System.out.println("Размер: " + size);
    System.out.println("Пуста: " + isEmpty());

    for (int i = 0; i < table.length; i++) {
        if (table[i] != null) {
            System.out.print("Индекс " + i + ": ");
            for (Entry<K, V> entry : table[i]) {
                System.out.print "[" + entry.getKey() + "=" + entry.getValue() + " ] ");
            }
            System.out.println();
        }
    }
    System.out.println();
}

Run main | Debug main | Run | Debug
public static void main(String[] args) {
    System.out.println(x:"ДЕМОНСТРАЦИЯ РАБОТЫ ХЭШ-ТАБЛИЦЫ");
    System.out.println(x:"=====\\n");
}

```

```

HashMap<String, Integer> studentGrades = new HashMap<>();

System.out.println(x:"1. Начальное состояние:");
studentGrades.display();

System.out.println(x:"2. Добавляем оценки студентов:");
studentGrades.put(key:"Алексей", value:85);
studentGrades.put(key:"Мария", value:92);
studentGrades.put(key:"Иван", value:78);
studentGrades.put(key:"Ольга", value:95);
studentGrades.display();

System.out.println(x:"3. Получаем оценки:");
System.out.println("Оценка Алексея: " + studentGrades.get(key:"Алексей"));
System.out.println("Оценка Марии: " + studentGrades.get(key:"Мария"));
System.out.println("Оценка несуществующего студента: " + studentGrades.get(key:"Петр"));
System.out.println();

System.out.println(x:"4. Обновляем оценку Ивана:");
studentGrades.put(key:"Иван", value:88);
System.out.println("Новая оценка Ивана: " + studentGrades.get(key:"Иван"));
studentGrades.display();

System.out.println(x:"5. Удаляем Ольгу:");

```

```

System.out.println("Удаление выполнено: " + removed);
System.out.println("Оценка Ольги после удаления: " + studentGrades.get(key:"Ольга"));
studentGrades.display();

System.out.println(x:"6. Финальная проверка:");
System.out.println("Количество студентов: " + studentGrades.size());
System.out.println("Таблица пуста: " + studentGrades.isEmpty());

```

Вывод:

```

ДЕМОНСТРАЦИЯ РАБОТЫ ХЭШ-ТАБЛИЦЫ
=====

1. Начальное состояние:
Содержимое хэш-таблицы:
Размер: 0
Пуста: true

2. Добавляем оценки студентов:
Содержимое хэш-таблицы:
Размер: 4
Пуста: false
Индекс 3: [Мария=92]
Индекс 5: [Алексей=85]
Индекс 7: [Иван=78]
Индекс 12: [Ольга=95]

3. Получаем оценки:
Оценка Алексея: 85
Оценка Марии: 92
Оценка несуществующего студента: null

4. Обновляем оценку Ивана:
Новая оценка Ивана: 88
Содержимое хэш-таблицы:
Размер: 4
Пуста: false
Индекс 3: [Мария=92]
Индекс 5: [Алексей=85]
Индекс 7: [Иван=88]
Индекс 12: [Ольга=95]

5. Удаляем Ольгу:
Удаление выполнено: true
Оценка Ольги после удаления: null
Содержимое хэш-таблицы:
Размер: 3
Пуста: false
Индекс 3: [Мария=92]
Индекс 5: [Алексей=85]

Индекс 3: [Мария=92]
Индекс 5: [Алексей=85]
Индекс 7: [Иван=88]

6. Финальная проверка:
Количество студентов: 3
Таблица пуста: false
PS C:\Users\ivanb\OneDrive\Рабочий стол\Лабораторная работа №3>

```

Задание 2. Работа с встроенным классом HashMap.

4. Реализация хэш-таблицы для хранения информации о книгах в библиотеке. Ключом будет ISBN книги, а значением — объект класса Book, содержащий информацию о названии, авторе и количестве копий. Необходимо реализовать операции вставки, поиска и удаления книги по ISBN.

```

import java.util.Enumeration;
import java.util.Hashtable;
class Book {
    private String title;
    private String author;
    private int copies;

    public Book(String title, String author, int copies) {
        this.title = title;
        this.author = author;
        this.copies = copies;
    }

    public String getTitle() { return title; }
    public String getAuthor() { return author; }
    public int getCopies() { return copies; }

    public void setCopies(int copies) { this.copies = copies; }

    @Override
    public String toString() {
        return "Название: '" + title + "', Автор: '" + author + "', Копий: " + copies;
    }
}

public class Library {

```

```

    private Hashtable<String, Book> books;

    public Library() {
        books = new Hashtable<>();
    }

    public void addBook(String isbn, String title, String author, int copies) {
        Book book = new Book(title, author, copies);
        books.put(isbn, book);
        System.out.println("Книга добавлена: " + title + " (ISBN: " + isbn + ")");
    }

    public Book findBook(String isbn) {
        Book book = books.get(isbn);
        if (book != null) {
            System.out.println("Найдена книга: " + book + " (ISBN: " + isbn + ")");
        } else {
            System.out.println("Книга с ISBN " + isbn + " не найдена");
        }
        return book;
    }

    public boolean removeBook(String isbn) {
        Book removedBook = books.remove(isbn);
        if (removedBook != null) {

```

```

        System.out.println("книга удалена: " + removedBook.getTitle() + " (ISBN: " + isbn
        return true;
    } else {
        System.out.println("книга с ISBN " + isbn + " не найдена для удаления");
        return false;
    }
}

public void displayAllBooks() {
    System.out.println(x:"\n=== ВСЕ КНИГИ В БИБЛИОТЕКЕ ===");
    if (books.isEmpty()) {
        System.out.println(x:"Библиотека пуста");
        return;
    }

    Enumeration<String> isbnns = books.keys();
    while (isbnns.hasMoreElements()) {
        String isbn = isbnns.nextElement();
        Book book = books.get(isbn);
        System.out.println("ISBN: " + isbn + " -> " + book);
    }
}

```

```

Run main | Debug main | Run | Debug
public static void main(String[] args) {
    Library library = new Library();

    System.out.println(x:"=== ДОБАВЛЕНИЕ КНИГ ===");
    library.addBook(isbn:"978-5-389-07435-4", title:"Мастер и Маргарита", author:"Михаил
    library.addBook(isbn:"978-5-17-087885-8", title:"Преступление и наказание", author:"Ф
    library.addBook(isbn:"978-5-699-79890-5", title:"Война и мир", author:"Лев Толстой",

    System.out.println(x:"\n=== ПОИСК КНИГ ===");
    library.findBook(isbn:"978-5-389-07435-4");
    library.findBook(isbn:"999-0-000-00000-0");

    library.displayAllBooks();

    System.out.println(x:"\n=== УДАЛЕНИЕ КНИГИ ===");
    library.removeBook(isbn:"978-5-699-79890-5");

    library.removeBook(isbn:"999-0-000-00000-0");

    library.displayAllBooks();
}

```

Вывод:

```
==== ДОБАВЛЕНИЕ КНИГ ====
Книга добавлена: Мастер и Маргарита (ISBN: 978-5-389-07435-4)
Книга добавлена: Преступление и наказание (ISBN: 978-5-17-087885-8)
Книга добавлена: Война и мир (ISBN: 978-5-699-79890-5)

==== ПОИСК КНИГ ====
Найдена книга: Название: 'Мастер и Маргарита', Автор: 'Михаил Булгаков', Копий: 3 (ISBN: 978-5-389-07435-4)
Книга с ISBN 999-0-000-00000-0 не найдена

==== ВСЕ КНИГИ В БИБЛИОТЕКЕ ====
ISBN: 978-5-389-07435-4 -> Название: 'Мастер и Маргарита', Автор: 'Михаил Булгаков', Копий: 3
ISBN: 978-5-17-087885-8 -> Название: 'Преступление и наказание', Автор: 'Федор Достоевский', Копий: 5
ISBN: 978-5-699-79890-5 -> Название: 'Война и мир', Автор: 'Лев Толстой', Копий: 2

==== УДАЛЕНИЕ КНИГИ ====
Книга удалена: Война и мир (ISBN: 978-5-699-79890-5)
Книга с ISBN 999-0-000-00000-0 не найдена для удаления

==== ВСЕ КНИГИ В БИБЛИОТЕКЕ ====
ISBN: 978-5-389-07435-4 -> Название: 'Мастер и Маргарита', Автор: 'Михаил Булгаков', Копий: 3
ISBN: 978-5-17-087885-8 -> Название: 'Преступление и наказание', Автор: 'Федор Достоевский', Копий: 5
ISBN: 978-5-699-79890-5 -> Название: 'Война и мир', Автор: 'Лев Толстой', Копий: 2

==== УДАЛЕНИЕ КНИГИ ====
Книга удалена: Война и мир (ISBN: 978-5-699-79890-5)
Книга с ISBN 999-0-000-00000-0 не найдена для удаления

==== ВСЕ КНИГИ В БИБЛИОТЕКЕ ====
ISBN: 978-5-389-07435-4 -> Название: 'Мастер и Маргарита', Автор: 'Михаил Булгаков', Копий: 3
ISBN: 978-5-17-087885-8 -> Название: 'Преступление и наказание', Автор: 'Федор Достоевский', Копий: 5
ISBN: 978-5-699-79890-5 -> Название: 'Война и мир', Автор: 'Лев Толстой', Копий: 2

==== УДАЛЕНИЕ КНИГИ ====
Книга удалена: Война и мир (ISBN: 978-5-699-79890-5)
Книга с ISBN 999-0-000-00000-0 не найдена для удаления
ISBN: 978-5-389-07435-4 -> Название: 'Мастер и Маргарита', Автор: 'Михаил Булгаков', Копий: 3
ISBN: 978-5-17-087885-8 -> Название: 'Преступление и наказание', Автор: 'Федор Достоевский', Копий: 5
ISBN: 978-5-699-79890-5 -> Название: 'Война и мир', Автор: 'Лев Толстой', Копий: 2
```

Вывод: в процессе выполнения лабораторной работы я изучил принципы работы хэш-таблиц как фундаментальной структуры данных и приобрел практические навыки их реализации. Освоил методы разрешения коллизий и научился правильно переопределять методы equals() и hashCode() для корректной работы с пользовательскими объектами в хэш-таблицах.