**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая Кибернетика и Информационные технологии»

Лабораторная работа №3

По дисциплине «Информационные технологии и программирование»

Выполнил: Студент группы

БПИ 2301

Антонова Ирина

Москва

2024

**Цель:** изучение Hash-таблиц.

**Задание:**

Задание 1:

1. Создайте класс HashTable, который будет реализовывать хэштаблицу 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.

Задание 2: Работа с встроенным классом HashMap

Вариант 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.

**Ход работы:**

**Задание 1:**

import java.util.LinkedList;

public class HashTable<K, V> {

    private LinkedList<Entry<K, V>>[] table;

    private int size;

    private static final int CAPACITY = 15;

    // Внутренний класс для хранения пар ключ-значение

    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;

        }

    }

    // Конструктор, инициализирующий таблицу заданного размера

    @SuppressWarnings("unchecked")

    public HashTable() {

        table = new LinkedList[CAPACITY];

        size = 0;

    }

    // Хеш-функция

    private int hash(K key) {

        return Math.abs(key.hashCode()) % CAPACITY;

    }

    // Добавление пары ключ-значение

    public void put(K key, V value) {

        int index = hash(key);

        // Проверка, есть ли уже ключ в таблице

        if (table[index] == null) {

            table[index] = new LinkedList<>();

        }

        for (Entry<K, V> entry : table[index]) { // Разрешение коллизии

            if (entry.getKey().equals(key)) {

                entry.setValue(value); // Обновление значения, если ключ уже существует

                return;

            }

        }

        // Добавление новой записи, если ключа нет

        table[index].add(new Entry<>(key, value));

        size++;

    }

    // Получение значения по ключу

    public V get(K key) {

        int index = hash(key);

        if (table[index] != null) {

            for (Entry<K, V> entry : table[index]) {

                if (entry.getKey().equals(key)) {

                    return entry.getValue();

                }

            }

        }

        return null;

    }

    // Удаление пары ключ-значение

    public void remove(K key) {

        int index = hash(key);

        if (table[index] != null) {

            for (int i = 0; i < table[index].size(); i++) {

                if (table[index].get(i).getKey().equals(key)) {

                    table[index].remove(i);

                    size--;

                    return;

                }

            }

        }

        // Ключ не найден

    }

    // Получение размера таблицы

    public int size() {

        return size;

    }

    // Проверка, пуста ли таблица

    public boolean isEmpty() {

        return size == 0;

    }

    public static void main(String[] args) {

        HashTable<String, String> hashTable = new HashTable<>();

        hashTable.put("apple", "red");

        hashTable.put("banana", "yellow");

        hashTable.put("orange", "orange");

        System.out.println(hashTable.get("apple")); // Вывод: red

        System.out.println(hashTable.get("banana")); // Вывод: yellow

        hashTable.remove("banana");

        System.out.println(hashTable.get("banana")); // Вывод: null

        System.out.println(hashTable.size()); // Вывод: 2

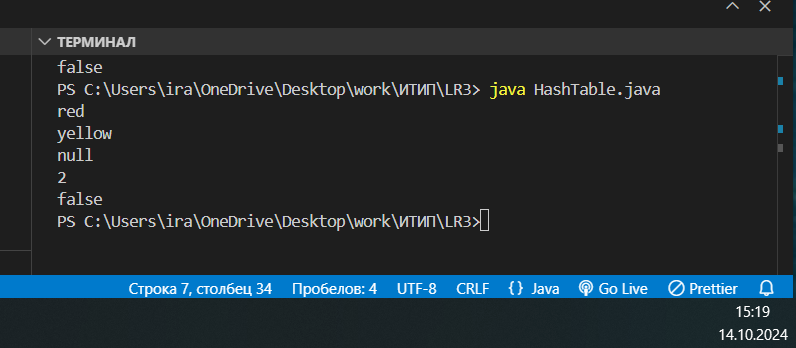
        System.out.println(hashTable.isEmpty()); // Вывод: false

    }

}

Двусвязный список — это связная структура данных, состоящая из набора последовательно связанных записей, называемых узлами. Каждый узел содержит два поля, называемых ссылками, которые указывают на предыдущий и последующий элементы в последовательности узлов. В таком списке каждый элемент (кроме первого и последнего) связан с предыдущим и следующим за ним элементами.

При запуске кода получим:



**Задание 2:**

Реализация класса Book:

public class Book {

    private String name;

    private String author;

    private int numberOfCopies;

    public Book(){

        this("Unknow", "Unknow", 0);

    }

    public Book(String name, String author, int numberOfCopies){

        this.name = name;

        this.author = author;

        this. numberOfCopies = numberOfCopies;

    }

    public String getName(){

        return name;

    }

    public void setName(String name){

        this.name = name;

    }

    public String getAuthor(){

        return author;

    }

    public void setAuthor(String author){

        this.author = author;

    }

    public int getNum(){

        return numberOfCopies;

    }

    public void setNum(int numberOfCopies){

        this.numberOfCopies = numberOfCopies;

    }

}

Работа с встроенным классом:

import java.util.HashMap;

public class BookTable{

    private HashMap<String, Book> books;

    public BookTable() {

        books = new HashMap<>();

    }

    public void addBook(String isbn, Book book) {

        books.put(isbn, book);

    }

    public Book getBook(String isbn) {

        return books.get(isbn);

    }

    public Book removeBook(String isbn) {

        return books.remove(isbn);

    }

    public int size() {

        return books.size();

    }

    public boolean isEmpty() {

        return books.isEmpty();

    }

    public static void main(String[] args) {

        // Создание экземпляра библиотеки

        BookTable library = new BookTable();

        // Создание книг

        Book book1 = new Book("1984", "George Orwell", 3);

        Book book2 = new Book("To Kill a Mockingbird", "Harper Lee", 2);

        Book book3 = new Book("The Great Gatsby", "F. Scott Fitzgerald", 5);

        // Добавление книг в библиотеку

        library.addBook("978-0451524935", book1);

        library.addBook("978-0061120084", book2);

        library.addBook("978-0743273565", book3);

        // Вывод количества книг в библиотеке

        System.out.println("Количество книг в библиотеке: " + library.size()); // 3

        // Получение книги по ISBN

        Book retrievedBook = library.getBook("978-0451524935");

        if (retrievedBook != null) {

            System.out.println("Полученная книга: " + retrievedBook.getName() + " автор: " + retrievedBook.getAuthor() + ", копий: " + retrievedBook.getNum());

        } else {

            System.out.println("Книга не найдена.");

        }

        // Удаление книги

        Book removedBook = library.removeBook("978-0061120084");

        if (removedBook != null) {

            System.out.println("Удалена книга: " + removedBook.getName());

        } else {

            System.out.println("Книга не найдена для удаления.");

        }

        // Проверка, пуста ли библиотека

        System.out.println("Библиотека пуста? " + library.isEmpty()); // false

        // Вывод оставшихся книг в библиотеке

        System.out.println("Осталось книг в библиотеке: " + library.size()); // 2

        // Вывод всех оставшихся книг

        System.out.println("Список оставшихся книг:");

        for (String isbn : library.books.keySet()) {

            Book book = library.getBook(isbn);

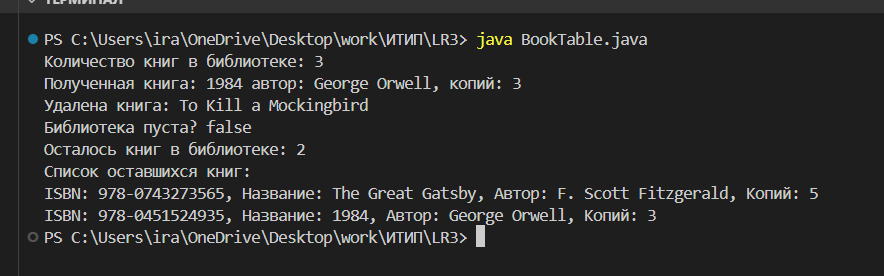
            System.out.println("ISBN: " + isbn + ", Название: " + book.getName() + ", Автор: " + book.getAuthor() + ", Копий: " + book.getNum());

        }

    }

}

Ответ программы после запуска:



**Вывод:** получили опыт работы с хэш-таблицами..