**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

**Лабораторная работа №3**

**Класс Object. Работа с хэш-таблицами**

Выполнил: студент группы \_\_\_БВТ2402\_\_\_

Голиков Михаил Вячеславович

Руководитель: \_Мосева Марина Сергеевна\_

Москва, 2077

## Цель работы

Целью лабораторной работы является изучение и закрепление навыков работы с хэш-таблицами в Java, включая добавление, поиск и удаление элементов, а также применение этих знаний для реализации телефонной книги с объектами класса Contact. Работа направлена на практическое освоение методов работы с коллекциями и структурой данных «ключ-значение».

## Индивидуальное задание

Реализовать класс HashTable, который хранит пары «ключ-значение» с использованием метода цепочек для разрешения коллизий. Ключом и значением будет строка. Необходимо реализовать методы вставки (put), поиска (get) и удаления (remove) элементов, а также методы size() и isEmpty().

Дополнительно реализовать работу с телефонной книгой через хэш-таблицу (или HashMap), где ключом является номер телефона, а значением — объект Contact, содержащий имя, электронную почту и дополнительные данные. Реализовать операции добавления, поиска и удаления контактов (Вариант 6).

## Основная часть

**Вариант 6**

В ходе работы были реализованы следующие компоненты:

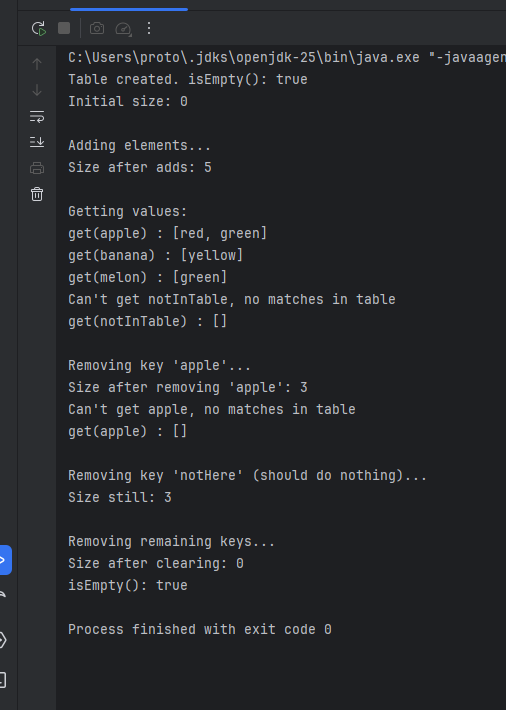
1. **Класс HashTable** — реализует хэш-таблицу с методом цепочек. Включает методы:
   * put(key, value) — добавление пары ключ-значение,
   * get(key) — получение всех значений по ключу,
   * remove(key) — удаление элементов по ключу,
   * size() — получение количества элементов,
   * isEmpty() — проверка на пустоту таблицы.
2. **Программа TaskOne** — тестирует работу класса HashTable через добавление, поиск и удаление элементов, включая случаи дублирующихся ключей и несуществующих элементов.
3. **Программа TaskTwo** — реализует телефонную книгу с классом Contact. Основные функции:
   * добавление нового контакта,
   * поиск контакта по номеру телефона,
   * удаление контакта.  
     Все операции сопровождаются выводом информации в консоль для наглядности.

package Lab\_3;  
  
import java.util.ArrayList;  
import java.util.LinkedList;  
import java.util.List;  
  
  
  
  
public class HashTable {  
  
 public static class Entry{  
 String key;  
 String value;  
  
 Entry(String key, String value){  
 this.key = key;  
 this.value = value;  
 }  
 }  
  
 private int numIndexes;  
 private int elemCounts;  
 protected LinkedList<Entry>[] table;  
  
  
 public HashTable(int numIndexes){  
 this.numIndexes = numIndexes;  
 this.elemCounts = 0;  
  
 table = new LinkedList[numIndexes];  
 for (int i = 0; i < this.numIndexes; i++){ // правильно создаём таблицу  
 table[i] = new LinkedList<Entry>();  
 }  
 }  
  
 public HashTable(){  
 this(8);  
 }  
  
  
 public boolean isEmpty(){  
 return elemCounts == 0;  
 }  
 public int size(){  
 return elemCounts;  
 }  
  
 public int hash(String input){  
 return Math.*abs*(input.hashCode()) % this.numIndexes;  
 }  
  
  
  
 public String[] get(String key){  
 int index = hash(key);  
 List<String> values = new ArrayList<>();  
  
 for (Entry e: table[index]){  
 if (e.key.equals(key)){  
 values.add(e.value);  
 }  
 }  
 if (values.isEmpty()) {  
 System.*out*.println("Can't get " + key + ", no matches in table");  
 }  
 return values.toArray(new String[0]);  
  
 }  
  
 public void put(String key, String value){ // change string  
 int index = hash(key);  
 table[index].add(new Entry(key, value));  
 this.elemCounts+=1;  
 }  
  
 public void remove(String key){  
 int index = hash(key);  
 int before = table[index].size();  
 table[index].removeIf(e -> e.key.equals(key));  
 elemCounts -= (before - table[index].size());  
 }  
}

Элемент 1 ― Код программы HashTable

package Lab\_3;  
  
public class TaskOne {  
 public static void main(String[] args) {  
  
 HashTable ht = new HashTable(4);  
  
 System.*out*.println("Table created. isEmpty(): " + ht.isEmpty());  
 System.*out*.println("Initial size: " + ht.size());  
 System.*out*.println();  
  
 // --- Test put() ---  
 System.*out*.println("Adding elements...");  
 ht.put("apple", "red");  
 ht.put("banana", "yellow");  
 ht.put("cherry", "red");  
 ht.put("apple", "green"); // duplicate key (to test multiple values for same key)  
 ht.put("melon", "green"); // potential collision due to small table size  
 System.*out*.println("Size after adds: " + ht.size());  
 System.*out*.println();  
  
 // --- Test get() ---  
 System.*out*.println("Getting values:");  
 *printArray*("apple", ht.get("apple"));  
 *printArray*("banana", ht.get("banana"));  
 *printArray*("melon", ht.get("melon"));  
 *printArray*("notInTable", ht.get("notInTable")); // non-existent key  
 System.*out*.println();  
  
 // --- Test remove() ---  
 System.*out*.println("Removing key 'apple'...");  
 ht.remove("apple");  
 System.*out*.println("Size after removing 'apple': " + ht.size());  
 *printArray*("apple", ht.get("apple")); // should now be empty  
 System.*out*.println();  
  
 // --- Test remove non-existing key ---  
 System.*out*.println("Removing key 'notHere' (should do nothing)...");  
 ht.remove("notHere");  
 System.*out*.println("Size still: " + ht.size());  
 System.*out*.println();  
  
 // --- Test isEmpty() after all removals ---  
 System.*out*.println("Removing remaining keys...");  
 ht.remove("banana");  
 ht.remove("cherry");  
 ht.remove("melon");  
 System.*out*.println("Size after clearing: " + ht.size());  
 System.*out*.println("isEmpty(): " + ht.isEmpty());  
  
 System.*out*.println("\n=== Tests Complete ===");  
 }  
  
 // Helper method to print arrays  
 private static void printArray(String key, String[] arr) {  
 System.*out*.print("get(" + key + ") : ");  
 if (arr == null || arr.length == 0) {  
 System.*out*.println("[]");  
 return;  
 }  
 System.*out*.print("[");  
 for (int i = 0; i < arr.length; i++) {  
 System.*out*.print(arr[i]);  
 if (i < arr.length - 1) System.*out*.print(", ");  
 }  
 System.*out*.println("]");  
 }  
}

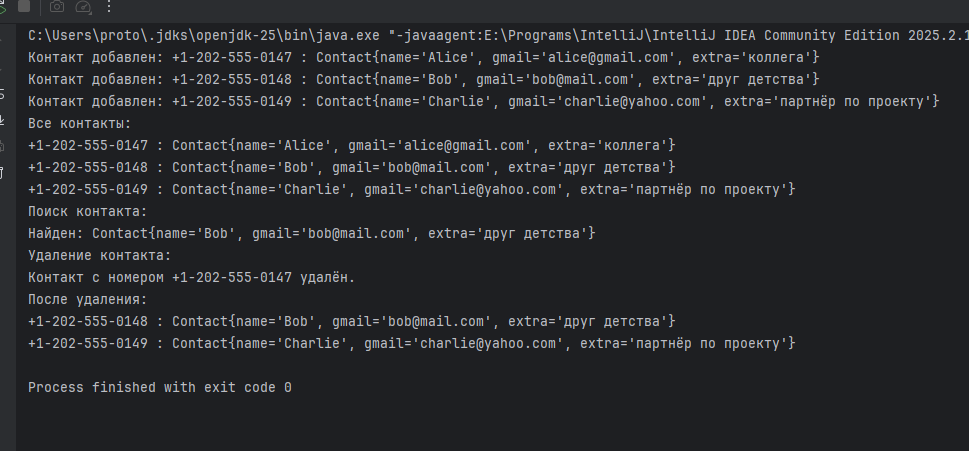
Элемент 2 ― Код программы TaskOne



Элемент 3 ― Вывод работы программы TaskOne

package Lab\_3;  
  
import java.util.\*;  
  
public class TaskTwo {  
  
 public static class Contact {  
 String name;  
 String gmail;  
 String extra;  
  
 // Constructor to initialize the contact  
 public Contact(String name, String gmail, String extra) {  
 this.name = name;  
 this.gmail = gmail;  
 this.extra = extra;  
 }  
  
 @Override  
 public String toString() {  
 return "Contact{name='" + name + "', gmail='" + gmail + "', extra='" + extra + "'}";  
 }  
 }  
  
 public static void insertContact(HashMap<String, Contact> hm, String phone, Contact contact) {  
 hm.put(phone, contact);  
 System.*out*.println("Контакт добавлен: " + phone + " : " + contact);  
 }  
 public static Contact findContact(HashMap<String, Contact> hm, String phone) {  
 if (hm.containsKey(phone)) {  
 return hm.get(phone);  
 } else {  
 System.*out*.println("Контакт с номером " + phone + " не найден.");  
 return null;  
 }  
 }  
 public static void removeContact(HashMap<String, Contact> hm, String phone) {  
 if (hm.containsKey(phone)) {  
 hm.remove(phone);  
 System.*out*.println("🗑 Контакт с номером " + phone + " удалён.");  
 } else {  
 System.*out*.println("Невозможно удалить — контакт с номером " + phone + " не найден.");  
 }  
 }  
 public static void main(String[] args){  
 HashMap<String, Contact> phoneBook = new HashMap<>();  
  
 *insertContact*(phoneBook, "+1-202-555-0147", new Contact("Alice", "alice@gmail.com", "коллега"));  
 *insertContact*(phoneBook, "+1-202-555-0148", new Contact("Bob", "bob@mail.com", "друг детства"));  
 *insertContact*(phoneBook, "+1-202-555-0149", new Contact("Charlie", "charlie@yahoo.com", "партнёр по проекту"));  
  
 System.*out*.println("Все контакты:");  
 for (String phone : phoneBook.keySet()) {  
 System.*out*.println(phone + " : " + phoneBook.get(phone));  
 }  
  
 System.*out*.println("Поиск контакта:");  
 Contact c = *findContact*(phoneBook, "+1-202-555-0148");  
 if (c != null) System.*out*.println("Найден: " + c);  
  
 System.*out*.println("Удаление контакта:");  
 *removeContact*(phoneBook, "+1-202-555-0147");  
  
 System.*out*.println("После удаления:");  
 for (String phone : phoneBook.keySet()) {  
 System.*out*.println(phone + " : " + phoneBook.get(phone));  
 }  
 }  
}

Элемент 4 ― Код программы TaskTwo



Элемент 5 ― Вывод работы программы TaskTwo

## Заключение

В ходе выполнения лабораторной работы был реализован класс HashTable с методом цепочек для разрешения коллизий и проверены его методы: put, get, remove, size и isEmpty. Также создана телефонная книга на основе хэш-таблицы (HashMap), позволяющая добавлять, искать и удалять контакты по номеру телефона. Работа позволила закрепить практические навыки работы с хэш-таблицами и объектами, применяемыми в реальных приложениях.

**Github:** [**https://github.com/Prototype721/Java**](https://github.com/Prototype721/Java)