

**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<K, V> {

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

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

    private int numIndexes;
    private int elemCounts;
    protected LinkedList<Entry<K, V>>[] table;

    public HashTable(int numIndexes) {
        this.numIndexes = numIndexes;
        this.elemCounts = 0;
        table = new LinkedList[numIndexes];
        for (int i = 0; i < this.numIndexes; i++) {
            table[i] = new LinkedList<>();
        }
    }

    public HashTable() {
        this(8);
    }

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

    public int size() {
        return elemCounts;
    }
}
```

```

public int hash(K key) {
    return Math.abs(key.hashCode()) % this.numIndexes;
}

@Override
public boolean equals(Object obj) {
    if (this == obj) return true;
    if (obj == null || getClass() != obj.getClass()) return false;

    HashTable<?, ?> other = (HashTable<?, ?>) obj;
    return this.elemCounts == other.elemCounts;
}

public List<V> get(K key) {
    int index = hash(key);
    List<V> values = new ArrayList<>();

    for (Entry<K, V> 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;
}

public void put(K key, V value) {
    int index = hash(key);
    for (Entry<K, V> e : table[index]) {
        if (e.key.equals(key)) {
            e.value = value;
            return;
        }
    }
    table[index].add(new Entry<>(key, value));
    this.elemCounts++;
}

public void remove(K 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;

import java.util.List;

public class TaskOne {

```

```

public static void main(String[] args) {

    Hashtable<String, String> ht = new Hashtable<>(4);

    System.out.println("Table created. isEmpty(): " + ht.isEmpty());
    System.out.println("Initial size: " + ht.size());
    System.out.println();

    System.out.println("Adding elements...");
    ht.put("apple", "red");
    ht.put("banana", "yellow");
    ht.put("cherry", "red");
    ht.put("apple", "green"); // заменить "red"
    ht.put("melon", "green");
    System.out.println("Size after adds: " + ht.size());
    System.out.println();

    System.out.println("Getting values:");
    printList("apple", ht.get("apple"));
    printList("banana", ht.get("banana"));
    printList("melon", ht.get("melon"));
    printList("notInTable", ht.get("notInTable"));
    System.out.println();

    System.out.println("Removing key 'apple'...");
    ht.remove("apple");
    System.out.println("Size after removing 'apple': " + ht.size());
    printList("apple", ht.get("apple"));
    System.out.println();

    System.out.println("Removing key 'notHere' (should do nothing)...");
    ht.remove("notHere");
    System.out.println("Size still: " + ht.size());
    System.out.println();

    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());
}

private static <V> void printList(String key, List<V> list) {
    System.out.print("get(" + key + ") : ");
    if (list == null || list.isEmpty()) {
        System.out.println("[]");
        return;
    }
    System.out.print("[");
    for (int i = 0; i < list.size(); i++) {
        System.out.print(list.get(i));
        if (i < list.size() - 1) System.out.print(", ");
    }
    System.out.println("]");
}
}

```

## Элемент 2 — Код программы TaskOne

```

C:\Users\proto\.jdk\openjdk-25\bin\java.exe "-javaa
Table created. isEmpty(): true
Initial size: 0

Adding elements...
Size after adds: 4

Getting values:
get(apple) : [green]
get(banana) : [yellow]
get(melon) : [green]
Can't get notInTable, no matches in table
get(notInTable) : []

Removing key 'apple'...
Size after removing 'apple': 3
Can't get apple, no matches in table
get(apple) : []

Removing key 'notHere' (should do nothing)...
Size still: 3

Removing remaining keys...
Size after clearing: 0
isEmpty(): true

Process finished with exit code 0

```

### 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;

        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

```
C:\Users\proto\jdk\openjdk-25\bin\java.exe "-javaagent:E:\Programs\IntelliJ\IntelliJ IDEA Community Edition 2025.2.1\lib\idea_rt.jar"
Контакт добавлен: +1-202-555-0147 : Contact{name='Alice', gmail='alice@gmail.com', extra='коллега'}
Контакт добавлен: +1-202-555-0148 : Contact{name='Bob', gmail='bob@mail.com', extra='друг детства'}
Контакт добавлен: +1-202-555-0149 : Contact{name='Charlie', gmail='charlie@yahoo.com', extra='партнёр по проекту'}
Все контакты:
+1-202-555-0147 : Contact{name='Alice', gmail='alice@gmail.com', extra='коллега'}
+1-202-555-0148 : Contact{name='Bob', gmail='bob@mail.com', extra='друг детства'}
+1-202-555-0149 : Contact{name='Charlie', gmail='charlie@yahoo.com', extra='партнёр по проекту'}
Поиск контакта:
Найден: Contact{name='Bob', gmail='bob@mail.com', extra='друг детства'}
Удаление контакта:
Контакт с номером +1-202-555-0147 удалён.
После удаления:
+1-202-555-0148 : Contact{name='Bob', gmail='bob@mail.com', extra='друг детства'}
+1-202-555-0149 : Contact{name='Charlie', gmail='charlie@yahoo.com', extra='партнёр по проекту'}

Process finished with exit code 0
```

## 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>