**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

Класс Object. Работа с хэш-таблицами

По дисциплине: Информационные технологии и программирование

Выполнил: студент группы БВТ-2401

Язалиев Тимур Исламович

Руководитель: Харрасов Камиль Раисович

Москва, 2025

Вариант: 1

1. Цель работы

Изучить основы работы с классом Object, методами equals(), hashCode() и toString(), а также реализовать хэш-таблицу для хранения информации о студентах с использованием номера зачетной книжки в качестве ключа.

2. Индивидуальное задание

Реализовать хэш-таблицу для хранения информации о студентах. Ключом является номер зачетной книжки, а значением — объект класса Student, содержащий поля: имя, фамилия, возраст и средний балл. Необходимо реализовать операции вставки, поиска и удаления студента по номеру зачетки.

3. Основная часть

**Задание 1.Реализация хэш-таблицы с методом цепочек.**

import java.util.LinkedList;

public class HashTable<K, V> {

private static class Entry<K, V> {

K key;

V value;

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[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<>();

}

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

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

entry.setValue(value);

return;

}

}

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

size++;

}

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;

}

public void remove(K key) {

int index = hash(key);

if (table[index] != null) {

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

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

table[index].remove(entry);

size--;

return;

}

}

}

}

public int size() {

return size;

}

public boolean isEmpty() {

return size == 0;

}

}

**Задание 2.Работа с HashMap для хранения информации о студентах.**

import java.util.HashMap;

import java.util.Map;

public class Student {

private String firstName;

private String lastName;

private int age;

private double averageGrade;

public Student(String firstName, String lastName, int age, double averageGrade) {

this.firstName = firstName;

this.lastName = lastName;

this.age = age;

this.averageGrade = averageGrade;

}

public String getFirstName() {

return firstName;

}

public String getLastName() {

return lastName;

}

public int getAge() {

return age;

}

public double getAverageGrade() {

return averageGrade;

}

@Override

public String toString() {

return String.format("Студент: %s %s, Возраст: %d, Средний балл: %.2f",

firstName, lastName, age, averageGrade);

}

}

public class StudentManagement {

private Map<String, Student> studentMap;

public StudentManagement() {

studentMap = new HashMap<>();

}

public void addStudent(String recordBookNumber, Student student) {

studentMap.put(recordBookNumber, student);

}

public Student getStudent(String recordBookNumber) {

return studentMap.get(recordBookNumber);

}

public void removeStudent(String recordBookNumber) {

studentMap.remove(recordBookNumber);

}

public void displayAllStudents() {

for (Map.Entry<String, Student> entry : studentMap.entrySet()) {

System.out.println("Зачётка: " + entry.getKey() + ", " + entry.getValue());

}

}

public static void main(String[] args) {

StudentManagement management = new StudentManagement();

Student student1 = new Student("Иван", "Иванов", 20, 4.5);

Student student2 = new Student("Петр", "Петров", 21, 4.2);

management.addStudent("12345", student1);

management.addStudent("67890", student2);

management.displayAllStudents();

System.out.println("\nПоиск студента с зачёткой 12345:");

System.out.println(management.getStudent("12345"));

management.removeStudent("67890");

System.out.println("\nПосле удаления студента с зачёткой 67890:");

management.displayAllStudents();

}

}

Заключение

-В ходе выполнения лабораторной работы:

-Изучены методы класса Object: equals(), hashCode(), toString().

-Реализована хэш-таблица с использованием метода цепочек для разрешения коллизий.

-Реализованы операции вставки, поиска и удаления элементов.

-Продемонстрирована работа хэш-таблицы на примере хранения данных о студентах.

Результаты работы подтверждают корректность реализации хэш-таблицы и методов класса Object.

*Ссылка на гит: https://github.com/Maggistr/LABWORK\_2\_sem.git*