1. Демонстрація HashSet та hashCode()

java

Copy

Download

import java.util.HashSet;

import java.util.Objects;

class Student {

private String name;

private String dateOfBirth;

private String email;

public Student(String name, String dateOfBirth, String email) {

this.name = name;

this.dateOfBirth = dateOfBirth;

this.email = email;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

Student student = (Student) o;

return Objects.equals(email, student.email);

}

@Override

public int hashCode() {

return Objects.hash(email);

}

@Override

public String toString() {

return name + " (" + email + ")";

}

}

public class HashSetDemo {

public static void main(String[] args) {

HashSet<Student> students = new HashSet<>();

*// Додаємо студентів (деякі з однаковими email)*

students.add(new Student("Іван Петров", "2000-05-15", "ivan@example.com"));

students.add(new Student("Марія Сидорова", "2001-03-22", "maria@example.com"));

students.add(new Student("Петро Іванов", "1999-11-10", "ivan@example.com")); *// Дублікат по email*

System.out.println("Студенти в HashSet:");

students.forEach(System.out::println);

System.out.println("Розмір HashSet: " + students.size());

}

}

2. Демонстрація TreeSet та Comparable

java

Copy

Download

import java.util.TreeSet;

class Group implements Comparable<Group> {

private String name;

private String specialty;

private int course;

public Group(String name, String specialty, int course) {

this.name = name;

this.specialty = specialty;

this.course = course;

}

@Override

public int compareTo(Group other) {

*// Спочатку сортуємо за курсом, потім за спеціальністю*

if (this.course != other.course) {

return Integer.compare(this.course, other.course);

}

return this.specialty.compareTo(other.specialty);

}

@Override

public String toString() {

return name + " (" + specialty + ", " + course + " курс)";

}

}

public class TreeSetDemo {

public static void main(String[] args) {

TreeSet<Group> groups = new TreeSet<>();

groups.add(new Group("Група А", "Комп'ютерні науки", 3));

groups.add(new Group("Група Б", "Математика", 2));

groups.add(new Group("Група В", "Комп'ютерні науки", 2));

groups.add(new Group("Група Г", "Фізика", 3));

System.out.println("Групи в TreeSet (відсортовані):");

groups.forEach(System.out::println);

}

}

3. Демонстрація TreeMap та Comparable

java

Copy

Download

import java.util.TreeMap;

class Teacher implements Comparable<Teacher> {

private int id;

private String firstName;

private String lastName;

public Teacher(int id, String firstName, String lastName) {

this.id = id;

this.firstName = firstName;

this.lastName = lastName;

}

@Override

public int compareTo(Teacher other) {

*// Сортуємо за прізвищем, потім за ім'ям*

int lastNameCompare = this.lastName.compareTo(other.lastName);

if (lastNameCompare != 0) {

return lastNameCompare;

}

return this.firstName.compareTo(other.firstName);

}

@Override

public String toString() {

return lastName + " " + firstName + " (ID: " + id + ")";

}

}

public class TreeMapDemo {

public static void main(String[] args) {

TreeMap<Integer, Teacher> teachers = new TreeMap<>();

teachers.put(101, new Teacher(101, "Іван", "Петренко"));

teachers.put(102, new Teacher(102, "Олена", "Іваненко"));

teachers.put(103, new Teacher(103, "Петро", "Сидоренко"));

System.out.println("Викладачі в TreeMap (відсортовані за ключем - ID):");

teachers.forEach((id, teacher) -> System.out.println(id + ": " + teacher));

}

}

4. Демонстрація LinkedList

java

Copy

Download

import java.util.LinkedList;

import java.util.ListIterator;

class Discipline {

private String name;

private int idTaschet;

public Discipline(String name, int idTaschet) {

this.name = name;

this.idTaschet = idTaschet;

}

@Override

public String toString() {

return name + " (ID табелю: " + idTaschet + ")";

}

}

public class LinkedListDemo {

public static void main(String[] args) {

LinkedList<Discipline> disciplines = new LinkedList<>();

disciplines.add(new Discipline("Математика", 101));

disciplines.add(new Discipline("Фізика", 102));

disciplines.add(new Discipline("Програмування", 103));

System.out.println("Дисципліни в LinkedList:");

disciplines.forEach(System.out::println);

*// Демонстрація ітерації з можливістю зміни*

ListIterator<Discipline> iterator = disciplines.listIterator();

while (iterator.hasNext()) {

Discipline d = iterator.next();

if (d.toString().contains("Математика")) {

iterator.set(new Discipline("Вища математика", 101));

}

}

System.out.println("Оновлений LinkedList:");

disciplines.forEach(System.out::println);

}

}

5. Демонстрація ArrayList

java

Copy

Download

import java.util.ArrayList;

import java.util.Collections;

class Department {

private int id;

private String name;

private String manager;

public Department(int id, String name, String manager) {

this.id = id;

this.name = name;

this.manager = manager;

}

@Override

public String toString() {

return name + " (ID: " + id + ", керівник: " + manager + ")";

}

}

public class ArrayListDemo {

public static void main(String[] args) {

ArrayList<Department> departments = new ArrayList<>();

departments.add(new Department(1, "Кафедра математики", "Петренко І.І."));

departments.add(new Department(2, "Кафедра фізики", "Сидоренко О.П."));

departments.add(new Department(3, "Кафедра інформатики", "Іваненко М.М."));

System.out.println("Кафедри в ArrayList:");

departments.forEach(System.out::println);

*// Сортування за назвою*

Collections.sort(departments, (d1, d2) -> d1.toString().compareTo(d2.toString()));

System.out.println("Відсортовані кафедри:");

departments.forEach(System.out::println);

}

}

6. Демонстрація Queue

java

Copy

Download

import java.util.LinkedList;

import java.util.Queue;

class Schedule {

private int id;

private String date;

private String time;

private String audience;

public Schedule(int id, String date, String time, String audience) {

this.id = id;

this.date = date;

this.time = time;

this.audience = audience;

}

@Override

public String toString() {

return "Розклад #" + id + ": " + date + " о " + time + " в ауд. " + audience;

}

}

public class QueueDemo {

public static void main(String[] args) {

Queue<Schedule> scheduleQueue = new LinkedList<>();

scheduleQueue.add(new Schedule(1, "2023-09-01", "09:00", "101"));

scheduleQueue.add(new Schedule(2, "2023-09-01", "10:30", "202"));

scheduleQueue.add(new Schedule(3, "2023-09-02", "14:00", "105"));

System.out.println("Розклад у Queue:");

while (!scheduleQueue.isEmpty()) {

System.out.println("Обробляємо: " + scheduleQueue.poll());

}

}

}

7. Демонстрація PriorityQueue

java

Copy

Download

import java.util.PriorityQueue;

class Grade implements Comparable<Grade> {

private int id;

private int idStudent;

private int idDiscipline;

private int grade;

private String date;

public Grade(int id, int idStudent, int idDiscipline, int grade, String date) {

this.id = id;

this.idStudent = idStudent;

this.idDiscipline = idDiscipline;

this.grade = grade;

this.date = date;

}

@Override

public int compareTo(Grade other) {

*// Сортуємо за оцінкою (спочатку вищі оцінки)*

return Integer.compare(other.grade, this.grade);

}

@Override

public String toString() {

return "Оцінка #" + id + ": студент " + idStudent + ", дисципліна " + idDiscipline +

", оцінка " + grade + ", дата " + date;

}

}

public class PriorityQueueDemo {

public static void main(String[] args) {

PriorityQueue<Grade> gradesQueue = new PriorityQueue<>();

gradesQueue.add(new Grade(1, 101, 201, 90, "2023-05-10"));

gradesQueue.add(new Grade(2, 102, 201, 85, "2023-05-11"));

gradesQueue.add(new Grade(3, 101, 202, 95, "2023-05-12"));

gradesQueue.add(new Grade(4, 103, 201, 88, "2023-05-13"));

System.out.println("Оцінки в PriorityQueue (відсортовані за спаданням):");

while (!gradesQueue.isEmpty()) {

System.out.println(gradesQueue.poll());

}

}

}

Комплексний приклад з усіма кроками

java

Copy

Download

import java.util.\*;

class Student implements Comparable<Student> {

private int id;

private String name;

private String email;

public Student(int id, String name, String email) {

this.id = id;

this.name = name;

this.email = email;

}

@Override

public int compareTo(Student other) {

return this.name.compareTo(other.name);

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

Student student = (Student) o;

return id == student.id;

}

@Override

public int hashCode() {

return Objects.hash(id);

}

@Override

public String toString() {

return name + " (ID: " + id + ", " + email + ")";

}

public int getId() { return id; }

public String getName() { return name; }

public String getEmail() { return email; }

}

public class CollectionsDemo {

public static void main(String[] args) {

*// 1. Невпорядкований список*

List<Student> studentsList = new ArrayList<>();

studentsList.add(new Student(1, "Іван Петров", "ivan@example.com"));

studentsList.add(new Student(2, "Марія Сидорова", "maria@example.com"));

studentsList.add(new Student(3, "Петро Іванов", "petro@example.com"));

studentsList.add(new Student(1, "Іван Петров", "ivan@example.com")); *// Дублікат*

System.out.println("Невпорядкований список:");

studentsList.forEach(System.out::println);

*// 2. Структура з унікальними елементами (HashSet)*

Set<Student> studentsSet = new HashSet<>(studentsList);

System.out.println("\nHashSet (унікальні студенти):");

studentsSet.forEach(System.out::println);

*// 3. Сортування списку*

Collections.sort(studentsList);

System.out.println("\nВідсортований список за іменем:");

studentsList.forEach(System.out::println);

*// 4. Впорядкована структура з унікальними елементами (TreeSet)*

Set<Student> studentsTreeSet = new TreeSet<>(studentsList);

System.out.println("\nTreeSet (унікальні та відсортовані студенти):");

studentsTreeSet.forEach(System.out::println);

*// 5. Відображення (Map)*

Map<Integer, Student> studentsMap = new HashMap<>();

studentsList.forEach(s -> studentsMap.put(s.getId(), s));

System.out.println("\nMap (ID -> Студент):");

studentsMap.forEach((id, student) -> System.out.println(id + " -> " + student));

}

}