Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти і газу

Кафедра ІПЗ

Лабораторна робота №5

Узагальнення та пакети

Виконав:

ст.групи ІП-22-1

Чурчак В.В.

Перевірив:

Романишин Т. Л.

Івано-Франківськ

2023

**Мета**: Навчитися використовувати узагальнення для створення алгоритмів, що можуть працювати з різними типами даних та використовувати пакети для керування простором імен.

**Завдання**:

1. Створити клас Set, в якому реалізувати два методи з назвами setSet та getSet. Перший метод повинен приймати на вхід множину у вигляді одновимірного масиву, а другий – повертати цей масив. Також передбачити у класі Set, конструктор, через який можна було б зв’язувати масив із відповідним об’єктом класу Set.

2. Створити клас Operation в якому реалізувати методи із назвами: unionSets (об'єднання множин), intersectionSets (перетин множин), diferenceSets (різниця множин), symmetricDiffSets (симетрична різниця множин).

3. Кожен із вище зазначених методів повинен приймати на вхід два об’єкта класу Set, та повертати відповідну множину у вигляді одновимірного масиву.

4. Усі класи та методи повинні вміти працювати із множинами різного типу (цілі числа, дробові числа, символи, логічні значення).

5. Забороняється в ролі контейнера для елементів множин використовувати будь-які класи колекцій.

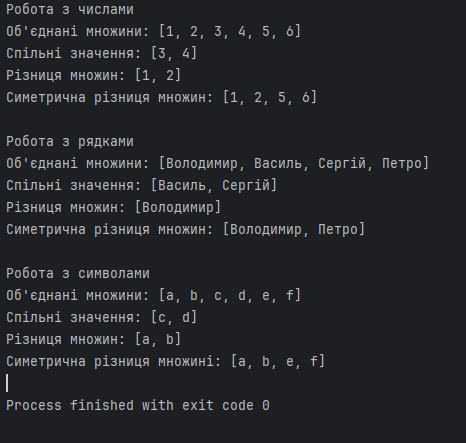
6. Усі створені класи повинні бути поміщені в індивідуальні пакети назва яких утворюється шляхом об’єднання назви факультету, назви групи та прізвища студента (наприклад, IT.PI\_10\_2.Petrenko).

Хід роботи}

package IT.PI\_22\_1.Churchak;  
  
import java.util.Arrays;  
import java.util.Objects;  
  
class Operation<T> {  
 // Об'єднання множин  
 public T[] unionSets(T[] set1, T[] set2) {  
 T[] result = Arrays.*copyOf*(set1, set1.length + set2.length);  
 int index = set1.length;  
  
 for (T element : set2) {  
 if (!contains(result, element)) {  
 result[index++] = element;  
 }  
 }  
  
 return Arrays.*copyOf*(result, index);  
 }  
  
 // Перетин множин  
 public T[] intersectionSets(T[] set1, T[] set2) {  
 int minSize = Math.*min*(set1.length, set2.length);  
 T[] result = Arrays.*copyOf*(set1, minSize);  
 int index = 0;  
  
 for (T element : set1) {  
 if (contains(set2, element)) {  
 result[index++] = element;  
 }  
 }  
  
 return Arrays.*copyOf*(result, index);  
 }  
  
 // Різниця множин  
 public T[] differenceSets(T[] set1, T[] set2) {  
 T[] result = Arrays.*copyOf*(set1, set1.length);  
 int index = 0;  
  
 for (T element : set1) {  
 if (!contains(set2, element)) {  
 result[index++] = element;  
 }  
 }  
  
 return Arrays.*copyOf*(result, index);  
 }  
  
 // Симетрична різниця множин  
 public T[] symmetricDiffSets(T[] set1, T[] set2) {  
 T[] unionResult = unionSets(set1, set2);  
 T[] intersectionResult = intersectionSets(set1, set2);  
  
 return differenceSets(unionResult, intersectionResult);  
 }  
  
 // Перевірка, чи містить масив певний елемент  
 private boolean contains(T[] array, T element) {  
 for (T value : array) {  
 if (Objects.*equals*(value, element)) {  
 return true;  
 }  
 }  
 return false;  
 }  
}

package IT.PI\_22\_1.Churchak;  
  
import java.util.Arrays;  
  
public class Set<T> {  
 private final T[] setArray;  
  
 public Set(T[] array) {  
 this.setArray = array;  
 }  
 public T[] getSet() {  
 return setArray;  
 }  
 public static void main(String[] args) {  
 // Приклад використання з цілими числами  
 Set<Integer> intSet1 = new Set<>(new Integer[]{1, 2, 3, 4});  
 Set<Integer> intSet2 = new Set<>(new Integer[]{3, 4, 5, 6});  
  
 Operation<Integer> intOperation = new Operation<>();  
  
 // Об'єднання множин  
 System.*out*.println("Робота з числами");  
  
 Integer[] unionResult = intOperation.unionSets(intSet1.getSet(), intSet2.getSet());  
 System.*out*.println("Об'єднані множини: " + Arrays.*toString*(unionResult));  
 // Перетин множин  
 Integer[] intersectionResult = intOperation.intersectionSets(intSet1.getSet(), intSet2.getSet());  
 System.*out*.println("Спільні значення: " + Arrays.*toString*(intersectionResult));  
  
 // Різниця множин  
 Integer[] differenceResult = intOperation.differenceSets(intSet1.getSet(), intSet2.getSet());  
 System.*out*.println("Різниця множин: " + Arrays.*toString*(differenceResult));  
  
 // Симетрична різниця множин  
 Integer[] symmetricDiffResult = intOperation.symmetricDiffSets(intSet1.getSet(), intSet2.getSet());  
 System.*out*.println("Симетрична різниця множин: " + Arrays.*toString*(symmetricDiffResult));  
  
 System.*out*.println();  
  
 // Приклад використання з рядками  
 Set<String> stringSet1 = new Set<>(new String[]{"Володимир", "Василь", "Сергій"});  
 Set<String> stringSet2 = new Set<>(new String[]{"Сергій", "Петро", "Василь"});  
  
 Operation<String> stringOperation = new Operation<>();  
  
 // Об'єднання множин  
 System.*out*.println("Робота з рядками");  
 String[] stringUnionResult = stringOperation.unionSets(stringSet1.getSet(), stringSet2.getSet());  
 System.*out*.println("Об'єднані множини: " + Arrays.*toString*(stringUnionResult));  
  
 // Перетин множин  
 String[] stringIntersectionResult = stringOperation.intersectionSets(stringSet1.getSet(), stringSet2.getSet());  
 System.*out*.println("Спільні значення: " + Arrays.*toString*(stringIntersectionResult));  
  
 // Різниця множин  
 String[] stringDifferenceResult = stringOperation.differenceSets(stringSet1.getSet(), stringSet2.getSet());  
 System.*out*.println("Різниця множин: " + Arrays.*toString*(stringDifferenceResult));  
  
 // Симетрична різниця множин  
 String[] stringSymmetricDiffResult = stringOperation.symmetricDiffSets(stringSet1.getSet(), stringSet2.getSet());  
 System.*out*.println("Симетрична різниця множин: " + Arrays.*toString*(stringSymmetricDiffResult));  
  
 System.*out*.println();  
  
 // Приклад використання з цілими символами  
 Set<Character> charSet1 = new Set<>(new Character[]{'a', 'b','c', 'd'});  
 Set<Character> charSet2 = new Set<>(new Character[]{'c', 'd', 'e', 'f'});  
  
 Operation<Character> charOperation = new Operation<>();  
  
 // Об'єднання множин  
 System.*out*.println("Робота з символами");  
 Character[] charUnionResult = charOperation.unionSets(charSet1.getSet(), charSet2.getSet());  
 System.*out*.println("Об'єднані множини: " + Arrays.*toString*(charUnionResult));  
  
 // Перетин множин  
 Character[] charIntersectionResult = charOperation.intersectionSets(charSet1.getSet(), charSet2.getSet());  
 System.*out*.println("Спільні значення: " + Arrays.*toString*(charIntersectionResult));  
  
 // Різниця множин  
 Character[] charDifferenceResult = charOperation.differenceSets(charSet1.getSet(), charSet2.getSet());  
 System.*out*.println("Різниця множин: " + Arrays.*toString*(charDifferenceResult));  
  
 // Симетрична різниця множин  
 Character[] charSymmetricDiffResult = charOperation.symmetricDiffSets(charSet1.getSet(), charSet2.getSet());  
 System.*out*.println("Симетрична різниця множині: " + Arrays.*toString*(charSymmetricDiffResult));  
  
 }  
}

**Результати:**



**Висновок:** в цій лабораторній роботі я ознайомився із узагальненням та пакетами, навчився використовувати використовувати їх під час написання Java-програм.

package Lab51.Operation;  
  
import java.util.Arrays;  
import java.util.Objects;  
  
public class Operation<T> {  
 // Об'єднання множин  
 public T[] unionSets(T[] set1, T[] set2) {  
 T[] result = Arrays.*copyOf*(set1, set1.length + set2.length);  
 int index = set1.length;  
  
 for (T element : set2) {  
 if (!contains(result, element)) {  
 result[index++] = element;  
 }  
 }  
  
 return Arrays.*copyOf*(result, index);  
 }  
  
 // Перетин множин  
 public T[] intersectionSets(T[] set1, T[] set2) {  
 int minSize = Math.*min*(set1.length, set2.length);  
 T[] result = Arrays.*copyOf*(set1, minSize);  
 int index = 0;  
  
 for (T element : set1) {  
 if (contains(set2, element)) {  
 result[index++] = element;  
 }  
 }  
  
 return Arrays.*copyOf*(result, index);  
 }  
  
 // Різниця множин  
 public T[] differenceSets(T[] set1, T[] set2) {  
 T[] result = Arrays.*copyOf*(set1, set1.length);  
 int index = 0;  
  
 for (T element : set1) {  
 if (!contains(set2, element)) {  
 result[index++] = element;  
 }  
 }  
  
 return Arrays.*copyOf*(result, index);  
 }  
  
 // Симетрична різниця множин  
 public T[] symmetricDiffSets(T[] set1, T[] set2) {  
 T[] unionResult = unionSets(set1, set2);  
 T[] intersectionResult = intersectionSets(set1, set2);  
  
 return differenceSets(unionResult, intersectionResult);  
 }  
  
 // Перевірка, чи містить масив певний елемент  
 private boolean contains(T[] array, T element) {  
 for (T value : array) {  
 if (Objects.*equals*(value, element)) {  
 return true;  
 }  
 }  
 return false;  
 }  
}