**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ТАРАСА ШЕВЧЕНКА**

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра інтелектуальних технологій**

**ЛАБОРАТОРНА РОБОТА №2**

з дисципліни «Об'єктно-орієнтоване програмування»

Тема роботи: «Відношення між класами: агрегація та композиція.»

Варіант №5

Виконала студентка

групи АнД-21

Радоманова С.П.

Перевірила:

Москаленко Н.В.

**Київ-2024**

Варіант 5: Клас Fraction для подання скороченого дробу, що задається парою цілих чисел: чисельник та знаменник. Реалізувати операції додавання/віднімання, множення/ділення, порівняння дробу. Метод для скорочення дробу реалізувати як приватний.

**Завдання 1**

1. Описати мовою java клас, в якому реалізовані такі методи:

• конструктор без параметрів для початкової ініціалізації полів;

• конструктор з параметрами – значеннями полів;

• метод toString(), який формує та повертає в якості результату рядок зі

значеннями полів класу (метод toString() буде використовуватися за

замовчуванням під час виведення полів класу на екран);

• метод для введення значення полів (введення даних користувачем є

обов’язковим);

• функції відповідно до варіанту завдання.

Поля класу мають бути закритими (приватними), для доступу до полів

необхідно реалізувати відповідні методи («сеттери» та «геттери»). В цих

методах не варто здійснювати введення чи виведення даних!

**Завдання 2**

1. Описати клас відповідно до варіанту завдання 2, який в якості одного з полів включає поле – екземпляр класу з завдання 1. Реалізувати зв’язок між

класами шляхом агрегації або композиції (свій вибір обґрунтувати).

Відмінності в реалізації зв’язку агрегації та композиції див. в лекції «Зв’язки

між класами».

2. Клас Calculator з повним набором арифметичних операцій з використання в

якості операндів екземплярів класу відповідно до варіанту завдання.

У нашому випадку використовуємо зв’язок агрегацію, оскільки об’єкт класу Fraction може окремо існувати і використовуватись за межами Calculator. Тому об'єкти класу Fraction передаються в клас Calculator через конструктор і не створюються всередині самого класу Calculator. Це відповідає концепції агрегації, де класи можуть існувати окремо, а один клас лише "використовує" інший.

### **Опис структури класу Fraction**

**Клас:** Fraction  
**Опис:** Клас, що представляє дроби.

#### **Поля класу:**

* private int numerator — чисельник дробу.
* private int denominator — знаменник дробу.

#### **Методи класу:**

* **Fraction()** — конструктор без параметрів, ініціалізує поля значеннями 1 для чисельника та знаменника.
* **Fraction(int numerator, int denominator)** — конструктор з параметрами, ініціалізує поля значеннями, переданими в параметрах, і приводить знак до чисельника, якщо знаменник від'ємний. Якщо знаменник дорівнює нулю, кидає виключення.
* **int getNumerator()** — метод для отримання значення чисельника.
* **void setNumerator(int numerator)** — метод для встановлення значення чисельника і автоматичного спрощення дробу.
* **int getDenominator()** — метод для отримання значення знаменника.
* **void setDenominator(int denominator)** — метод для встановлення значення знаменника (перевіряє на нуль).
* **String toString()** — перевизначений метод для виведення дробу у форматі "чисельник/знаменник".
* **void createFraction()** — метод для вводу даних дробу користувачем, з перевіркою на нульовий знаменник.
* **Fraction add(Fraction other)** — метод для додавання двох дробів і повернення нового дробу.
* **Fraction subtract(Fraction other)** — метод для віднімання другого дробу від першого та повернення нового дробу.
* **Fraction multiply(Fraction other)** — метод для множення двох дробів та повернення нового дробу.
* **Fraction divide(Fraction other)** — метод для ділення одного дробу на інший; якщо другий дроб має нульовий чисельник, кидає виключення.
* **String compare(Fraction other)** — метод для порівняння двох дробів; повертає рядок, що вказує, який дріб більший або чи вони рівні.
* **private int findGCD(int a, int b)** — приватний метод для знаходження найбільшого спільного дільника (НСД) за алгоритмом Евкліда.
* **private void simplify()** — приватний метод, що спрощує дріб до найменших значень за допомогою НСД.

### **Опис структури класу Calculator**

**Клас:** Calculator  
**Опис:** Клас, що виконує арифметичні операції над дробами, представленими об'єктами класу Fraction.

#### **Поля класу:**

* private Fraction fraction1 — перший дріб, з яким працює калькулятор.
* private Fraction fraction2 — другий дріб, з яким працює калькулятор.

#### **Методи класу:**

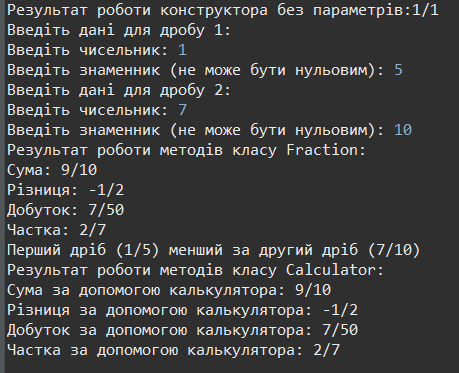
* **Calculator(Fraction fraction1, Fraction fraction2)** — конструктор, який приймає два дроби як параметри. Це реалізація агрегації, оскільки об'єкти класу Fraction передаються в конструктор.
* **Fraction addFractions()** — метод, що виконує додавання двох дробів, використовуючи метод add класу Fraction, і повертає новий об'єкт Fraction, що містить результат.
* **Fraction subtractFractions()** — метод, що виконує віднімання другого дробу від першого, використовуючи метод subtract класу Fraction, і повертає новий об'єкт Fraction, що містить результат.
* **Fraction multiplyFractions()** — метод, що виконує множення двох дробів, використовуючи метод multiply класу Fraction, і повертає новий об'єкт Fraction, що містить результат.
* **Fraction divideFractions()** — метод, що виконує ділення першого дробу на другий, використовуючи метод divide класу Fraction, і повертає новий об'єкт Fraction, що містить результат.
* **String toStringFractions()** — метод, що повертає рядок, який описує, з якими дробами працює калькулятор, включаючи представлення об'єктів fraction1 та fraction2.



| **Fraction** |  | **Calculator** |
| --- | --- | --- |
| * numerator: int * denominator: int |  | * fraction1: Fraction * ﻿﻿fraction2: Fraction |
| + Fraction()  + Fraction(int, int)  + add(Fraction): Fraction  + subtract(Fraction): Fraction + multiply(Fraction): Fraction + divide(Fraction): Fraction  + compare(Fraction): String  + toString(): String  + createFraction(): void  - findGDC(int, int): int  - simplify(): void |  | +Calculator(fraction1:Fraction, fraction2: Fraction)  + addFractions(): Fraction  + subtractFractions():Fraction  + multiplyFractions():Fraction  + divideFractions(): Fraction  + toStringFractions(): String |

UML - діаграма

**Тестові приклади:**



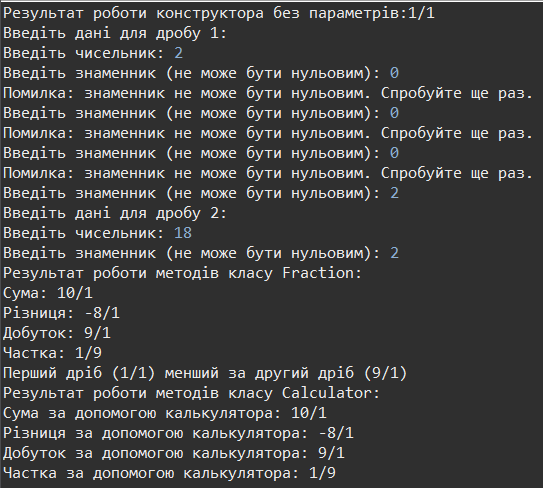
Обрахуємо вручну:

Сума:

Різниця:

Добуток:

Частка:



Обрахуємо вручну:

Сума:

Різниця:

Добуток:

Частка:

Працездатність доведено тестовими прикладами.

**Текст програми мовою Java:**

**(Fraction.java)**

package tasks;

import java.util.Scanner;

public class Fraction {

private int numerator; // чисельник

private int denominator; // знаменник

public Fraction() {

this.numerator = 1;

this.denominator = 1;

}

public Fraction(int numerator, int denominator)

{

// Переносимо знак до чисельника

if (denominator < 0)

{

this.numerator = -numerator;

this.denominator = -denominator;

} else {

this.numerator = numerator;

this.denominator = denominator;

}

simplify();

}

*@Override* //перезаписує метод toString() з класу Object, який є суперкласом для всіх класів у Java

public String toString() {

// Переносимо знак до чисельника

if (denominator < 0)

{

this.numerator = -numerator;

this.denominator = -denominator;

}

return numerator + "/" + denominator;

}

// Метод для вводу даних користувачем

public void createFraction() {

Scanner scanner = new Scanner(System.***in***);

System.***out***.print("Введіть чисельник: ");

this.numerator = scanner.nextInt();

do {

System.***out***.print("Введіть знаменник (не може бути нульовим): ");

this.denominator = scanner.nextInt();

if (this.denominator == 0) {

System.***out***.println("Помилка: знаменник не може бути нульовим. Спробуйте ще раз.");

}

} while (this.denominator == 0);

simplify(); // Спростити дроб

}

public void setNumerator(int numerator) {

this.numerator = numerator;

simplify();

}

public void setDenominator(int dominator) {

this.denominator = dominator;

simplify();

if (denominator == 0) {

throw new IllegalArgumentException("Знаменник не може бути нульовим.");

}

}

public int getNumerator() {

return numerator;

}

public int getDenominator() {

return denominator;

}

public Fraction add(Fraction other) //додавання

{

int newNumerator = this.numerator \* other.getDenominator() + other.getNumerator() \* this.denominator;

int newDenominator = this.denominator \* other.getDenominator();

Fraction result = new Fraction(newNumerator, newDenominator);

return result; // Виклик спрощення

}

public Fraction subtract(Fraction other) //віднімання

{

int newNumerator = this.numerator \* other.getDenominator() - other.getNumerator() \* this.denominator;

int newDenominator = this.denominator \* other.getDenominator();

return new Fraction(newNumerator, newDenominator);

}

public Fraction multiply(Fraction other) // множення

{

int newNumerator = this.numerator \* other.getNumerator();

int newDenominator = this.denominator \* other.getDenominator();

return new Fraction(newNumerator, newDenominator);

}

public Fraction divide(Fraction other) //ділення

{

if (other.getNumerator() == 0)

{

throw new IllegalArgumentException("Ділення на нуль недопустиме.");

}

int newNumerator = this.numerator \* other.getDenominator();

int newDenominator = this.denominator \* other.getNumerator();

return new Fraction(newNumerator, newDenominator);

}

public String compare(Fraction other) {

// Перевірка на нульовий знаменник

if (this.denominator == 0 || other.getDenominator() == 0) {

throw new IllegalArgumentException("Знаменник не може бути нульовим.");

}

// Зводимо обидва дроби до спільного знаменника

int firstNumerator = this.numerator \* other.getDenominator();

int secondNumerator = other.getNumerator() \* this.denominator;

// Порівнюємо чисельники

if (firstNumerator > secondNumerator) {

return "Перший дріб (" + this + ") більший за другий дріб (" + other + ")";

} else if (firstNumerator < secondNumerator) {

return "Перший дріб (" + this + ") менший за другий дріб (" + other + ")";

} else {

return "Обидва дроби (" + this + " і " + other + ") рівні";

}

}

private int findGCD(int a, int b) { // алгоритм Евкліда для пошуку НСД

if (b == 0) {

return a;

}

return findGCD(b, a % b);

}

private void simplify() {

if (denominator != 0) {

int gcd = findGCD(numerator, denominator);

numerator /= gcd;

denominator /= gcd;

}

}

}

**(main.java)**

**package tasks;**

**import java.util.Scanner;**

**public class main {**

**public static void main(String[] args)**

**{**

**Scanner scanner = new Scanner(System.*in*);**

**Fraction defaultfraction = new Fraction();**

**System.*out*.println("Результат роботи конструктора без параметрів:" + defaultfraction);**

**Fraction frac1 = new Fraction();**

**System.*out*.println("Введіть дані для дробу 1:");**

**frac1.createFraction();**

**Fraction frac2 = new Fraction();**

**System.*out*.println("Введіть дані для дробу 2:");**

**frac2.createFraction();**

**System.*out*.println("Результат роботи методів класу Fraction:");**

**Fraction resultAdd = frac1.add(frac2);**

**System.*out*.println("Сума: " + resultAdd);**

**Fraction resultSubtract = frac1.subtract(frac2);**

**System.*out*.println("Різниця: " + resultSubtract);**

**Fraction resultMultiply = frac1.multiply(frac2);**

**System.*out*.println("Добуток: " + resultMultiply);**

**Fraction resultDivide = frac1.divide(frac2);**

**System.*out*.println("Частка: " + resultDivide);**

**String resultCompare = frac1.compare(frac2); //Порівняння дробів**

**System.*out*.println(resultCompare);**

**// створюємо об'єкт класу Calculator, використовуючи конструктор**

**Calculator calculator = new Calculator(frac1, frac2);**

**System.*out*.println("Результат роботи методів класу Calculator:");**

**Fraction resultAddCalculator = calculator.addFractions();**

**System.*out*.println("Сума за допомогою калькулятора: " + resultAddCalculator);**

**Fraction resultSubtractCalculator = calculator.subtractFractions();**

**System.*out*.println("Різниця за допомогою калькулятора: " + resultSubtractCalculator);**

**Fraction resultMultiplyCalculator = calculator.multiplyFractions();**

**System.*out*.println("Добуток за допомогою калькулятора: " + resultMultiplyCalculator);**

**Fraction resultDivideCalculator = calculator.divideFractions();**

**System.*out*.println("Частка за допомогою калькулятора: " + resultDivideCalculator);**

**}**

**}**

**(Calculator.java)**

**package tasks;**

**public class Calculator {**

**private Fraction fraction1; // Перший дріб**

**private Fraction fraction2; // Другий дріб**

**public Calculator(Fraction fraction1, Fraction fraction2) // Агрегація: Fraction передається у Calculator**

**{**

**this.fraction1 = fraction1;**

**this.fraction2 = fraction2;**

**}**

**// Метод для додавання дробів**

**public Fraction addFractions()**

**{**

**return fraction1.add(fraction2);**

**}**

**public Fraction subtractFractions()**

**{**

**return fraction1.subtract(fraction2);**

**}**

**public Fraction multiplyFractions()**

**{**

**return fraction1.multiply(fraction2);**

**}**

**public Fraction divideFractions()**

**{**

**return fraction1.divide(fraction2);**

**}**

**public String toStringFractions()**

**{**

**return "Calculator working with: " + fraction1 + " and " + fraction2;**

**}**

**}**