**Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

з лабораторної роботи № 3 з дисципліни

«Технології паралельних обчислень»

«Розробка паралельних програм з використанням механізмів синхронізації: синхронізовані методи, локери, спеціальні типи**»**

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-14 Сергієнко Ю. В.*

**Перевірив**

(прізвище, ім'я, по батькові)

*Дифучина О. Ю.*

Київ 2024

Комп’ютерний практикум 3

**Тема:** Розробка паралельних програм з використанням механізмів синхронізації: синхронізовані методи, локери, спеціальні типи.

**Виконання:**

1. Реалізуйте програмний код, даний у лістингу, та протестуйте його при різних значеннях параметрів. Модифікуйте програму, використовуючи методи управління потоками, так, щоб її робота була завжди коректною. Запропонуйте три різних варіанти управління. **30 балів**.

Для виконання даного завдання необхідно модифікувати код програми для проведеня транзакцій. Основною проблемою є   
неправильний обрахунок трансферу – проблема Race Condition, за якої декілька потоків одночасно використовують один ресурс та неправильно записують результат. За параметрів NACCOUNTS = 10, INITIAL\_BALANCE = 1000, N\_LOG = 1000, REPS = 1000 бачимо, що транзакції перемішались (одночасно відбувався інкремент у декількох потоках), а сума відходить від добутку балансу та акаунтів (при додаванні одночасно в один ресурс записується нова сума, що призводить до втрати одного із значень) (рисунок 1).

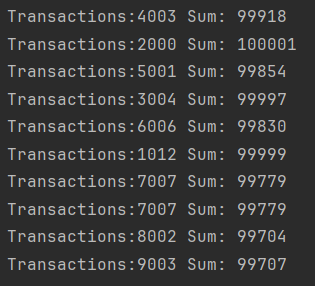


Рисунок 1 – Проблема Race Condition

Дану проблему можна вирішити за допомогою синхронізації роботи потоків, тому спробуємо зробити метод *transfer* синхронізованим. Як бачимо, програма працює правильно (рисунок 2).

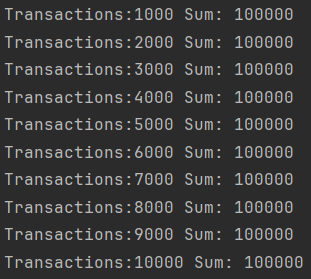


Рисунок 2 – Сихронізований метод

Спробуємо збільшити кількість операцій до 1000000 (рисунок 3).

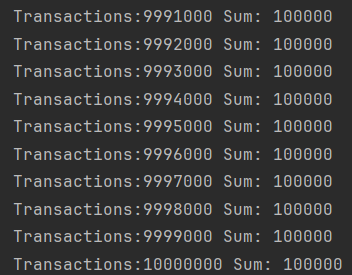


Рисунок 3 – Синхронізований метод (1000000 повторів)

Інше рішення проблеми – синхронне виконання потоків за допомогою локерів. У даному випадку використання ReentrantLock гарантує виконання лише однієї дії транферу одночасно, тому не перезапису даних не відбудеться (рисунок 4).

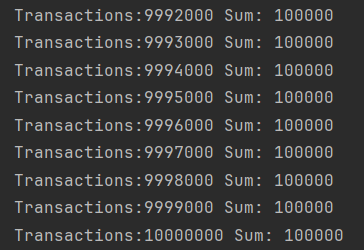


Рисунок 4 – Використання ReentrantLock

Спробуємо синхронізувати потоки по локеру Object. Фактично, ми досягнемо того ж ефекту, використовуючи блок synchronized (lock) (рисунок 5).

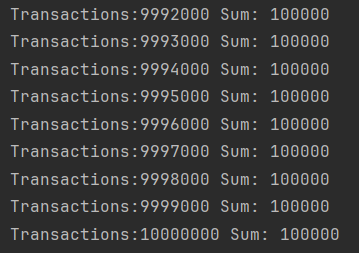


Рисунок 5 – Використання Object lock

1. Реалізуйте приклад Producer-Consumer application (див. https://docs.oracle.com/javase/tutorial/essential/concurrency/guardmeth.html ). Модифікуйте масив даних цієї програми, які читаються, у масив чисел заданого розміру (100, 1000 або 5000) та протестуйте програму. Зробіть висновок про правильність роботи програми. **20 балів**.

Основна задача – змінити тип масиву чисел що зчитується на числовий. Протестуємо це передаючи числа від 0 до Size у Consumer. Як бачимо, результат успішний (рисунок 6).

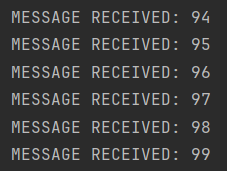


Рисунок 6 – Передача чисел

1. Реалізуйте роботу електронного журналу групи, в якому зберігаються оцінки з однієї дисципліни трьох груп студентів. Кожного тижня лектор і його 3 асистенти виставляють оцінки з дисципліни за 100-бальною шкалою. **40 балів**.

Для виконання даного завдання було створено клас GradeBook, де основним атрибутом є список студентів (об’єктів класу Student). Задачею є проставити поточні бали студентам, де максимальною оцінкою є WEEK \* MARK. Кожному асистенту належить одна група, де він може поставити WEEK / 2 оцінок, іншу половину кожній групі виставляє лектор.

Проблема виникає при додаванні балу в журнал студента: якщо лектор та асистент одночасно додають бали студенту, то інколи спрацьовує Memory Consistency Error – лише один результат записується. Для її вирішення необхідно синхронізувати роботу потоків (рисунок 7).

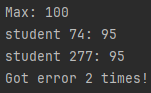


Рисунок 7 – Race condition

Найлегшим способом є використання синхронізованих методів, а результат роботи програми можна бачити на рисунку 8.



Рисунок 8 – Робота синхронізованих потоків

1. Зробіть висновки про використання методів управління потоками в java. **10 балів**.

**Висновок**

Під час виконання даного комп’ютерного практикуму я закріпив знання та навички щодо розробки паралельних алгоритмів. На прикладі декількох задач було змодельовано помилку Race Condition та знайдено її рішення у вигляді синхронізації потоків за допомогою синхронізованих методів, блоків та локерів. Також було модифіковано програму типу Producer – Consumer для надислання чисел замість текстових повідомлень.

Код програми доступний на [Github](https://github.com/sergienkoyura/kpi_tpo).