Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

**Комп’ютерний практикум № 3**

з дисципліни «Технології паралельних обчислень»

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

|  |  |
| --- | --- |
| **Виконав:**  студент групи ІТ-04  Чапча Святослав  Дата здачі \_\_\_\_\_\_\_\_  Захищено з балом \_\_\_\_\_\_\_ | **Перевірила:**  ас. кафедри ІПІ  Дифучина О.Ю. |

Київ 2023

**Завдання:**

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**.

Скріншоти програми:

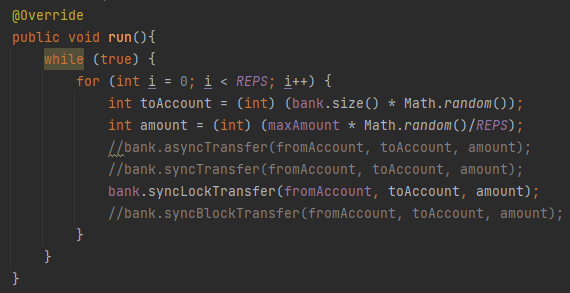


Рис. 1. TransferThread.java

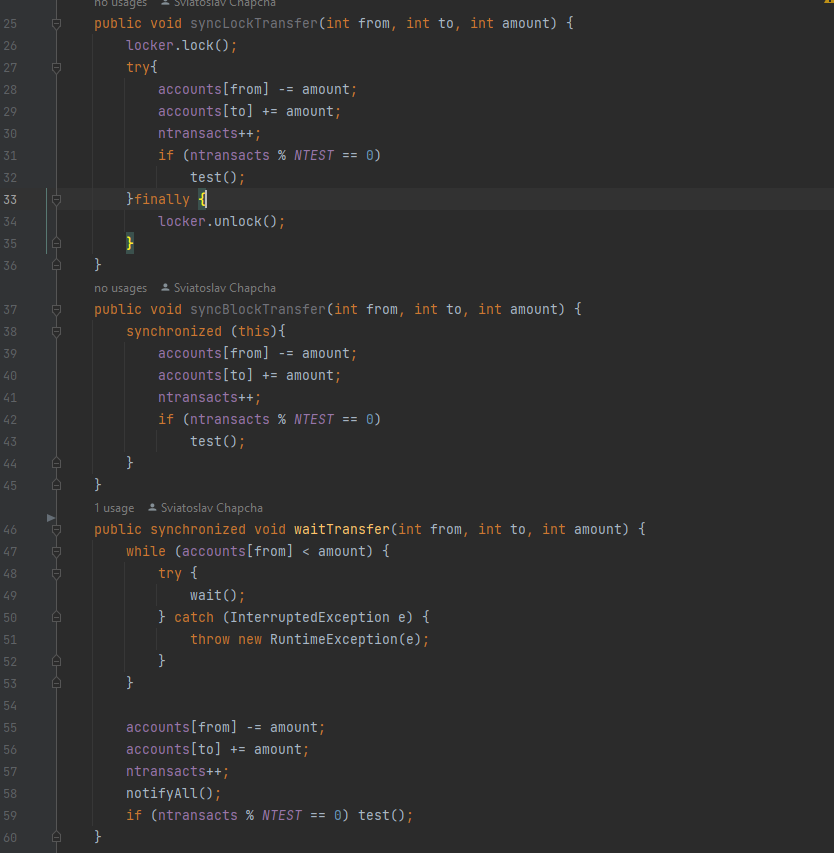


Рис. 2. Bank.java

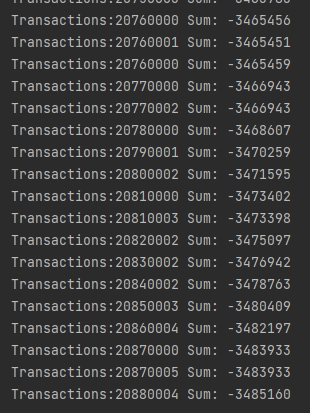


Рис. 3. Результат роботи асинхронізованої програми

Як бачимо, результат суми не зберігається і зменшується з кількістю транзакцій. Кількість транзакцій також обчислюється не точно.

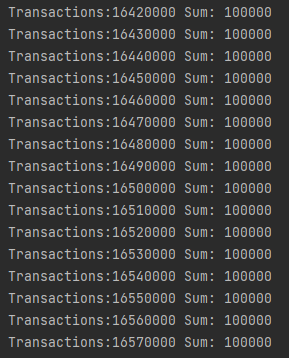


Рис. 4. Результати роботи синхронізованої програми

Після реалізації методів синхронізації, ми отримуємо стабільний та працюючий варіант. Результат у всіх випадках однаковий.

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**.

Скріншоти програми:



Рис. 5. Drop.java

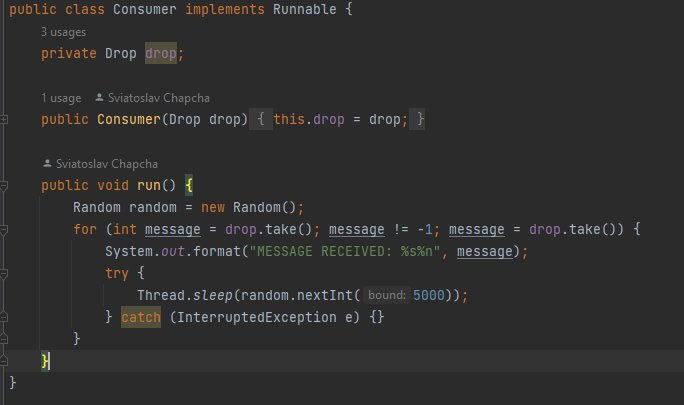


Рис. 6. Consumer.java

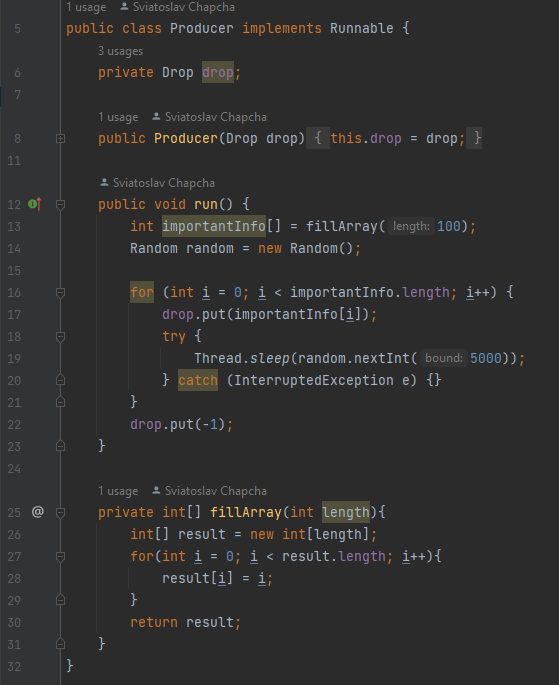


Рис. 7. Producer.java

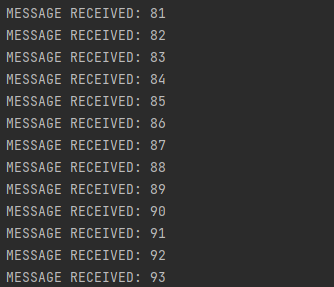


Рис. 8. Результат роботи

Як ми можемо побачити, завдяки тому, що ми додали синхронізований блок до методів put() та take(), повідомлення присилаються послідовно один за одним.

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.

Скріншоти програми:

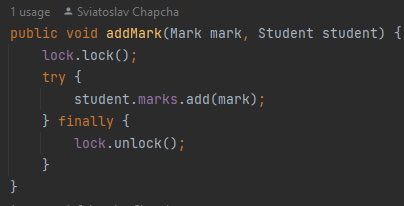


Рис. 9. Метод addMark()



Рис. 10. Teacher.java



Рис. 11. Результат програми при відсутності синхронізованого блокування

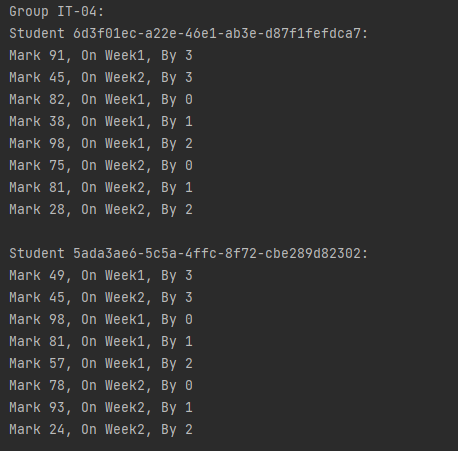
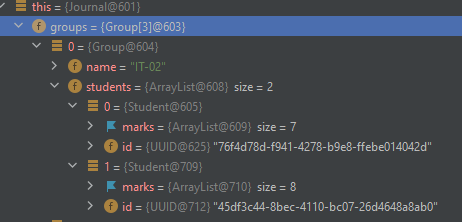


Рис. 12. Результат програми з синхронізованим блокуванням

Як ми можемо побачити, при відсутності синхронізованого блокування, з`являється помилка при виведенні самих оцінок. Якщо запустимо програму у debug режимі, то помітимо наступне:



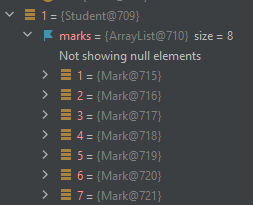


Рис. 13-14. Debug режим

Як ми можемо дізнатись, один із студентів залишився без оцінки, а інший студент отримав 8, але одна з оцінок просто не зберіглась і дорівнює Null.

Програмний код до всіх завдань наведений в репозиторії:

<https://github.com/whitetark/multithreading/tree/main/lab03>

**Висновки:**

Після виконання практичної роботи, можна зробити висновок, що, щоб отримати доступ до даних потрібно використовувати численні методи синхронізації, що є обов'язковим для правильної роботи програми з багатопотоковими додатками. Також, ми помітили, що можна використовувати різні методи, оскільки вони дають однаковий результат