

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

до комп'ютерного практикуму № 1 з дисципліни
«Технології паралельних обчислень»
на тему:
«Розробка потоків та дослідження пріоритету запуску потоків»

Виконав(ла)

ІП-92 Кулик Володимир Маркович
(шифр, прізвище, ім'я, по батькові)

Прийняв

Дифучина Олександра Юріївна
(прізвище, ім'я, по батькові)

Київ 2022

Комп'ютерний практикум №1

Тема: Розробка потоків та дослідження пріоритету запуску потоків

Завдання:

1. Реалізуйте програму імітації руху більярдних кульок, в якій рух кожної кульки відтворюється в окремому потоці (див. презентацію «Створення та запуск потоків в java» та приклад). Спостерігайте роботу програми при збільшенні кількості кульок. Поясніть результати спостереження. Опишіть переваги потокової архітектури програм. 10 балів.
2. Модифікуйте програму так, щоб при потраплянні в «лузу» кульки зникали, а відповідний потік завершував свою роботу. Кількість кульок, яка потрапила в «лузу», має динамічно відображатись у текстовому полі інтерфейсу програми. 10 балів.
3. Виконайте дослідження параметру `priority` потоку. Для цього модифікуйте програму «Більярдна кулька» так, щоб кульки червоного кольору створювалися з вищим пріоритетом потоку, в якому вони виконують рух, ніж кульки синього кольору. Спостерігайте рух червоних та синіх кульок при збільшенні загальної кількості кульок. Проведіть такий експеримент. Створіть багато кульок синього кольору (з низьким пріоритетом) і одну червоного кольору, які починають рух в одному й тому ж самому місці більярдного стола, в одному й тому ж самому напрямку та з однаковою швидкістю. Спостерігайте рух кульки з більшим пріоритетом. Повторіть експеримент кілька разів, значно збільшуючи кожного разу кількість кульок синього кольору. Зробіть висновки про вплив пріоритету потоку на його роботу в залежності від загальної кількості потоків. 20 балів.
4. Побудуйте ілюстрацію для методу `join()` класу `Thread` з використанням руху більярдних кульок різного кольору. Поясніть результат, який спостерігається. 10 балів.

5. Створіть два потоки, один з яких виводить на консоль символ ‘-’, а інший – символ ‘|’. Запустіть потоки в основній програмі так, щоб вони виводили свої символи в рядок. Виведіть на консоль 100 таких рядків. Поясніть виведений результат. 10 балів. Використовуючи найпростіші методи управління потоками, добийтесь почергового виведення на консоль символів. 15 балів.

6. Створіть клас Counter з методами increment() та decrement(), які збільшують та зменшують значення лічильника відповідно. Створіть два потоки, один з яких збільшує 100000 разів значення лічильника, а інший – зменшує 100000 разів значення лічильника. Запустіть потоки на одночасне виконання. Спостерігайте останнє значення лічильника. Поясніть результат. 10 балів. Використовуючи синхронізований доступ, добийтесь правильної роботи лічильника при одночасній роботі з ним двох і більше потоків. Опрацюйте використання таких способів синхронізації: синхронізований метод, синхронізований блок, блокування об’єкта. Порівняйте способи синхронізації. 15 балів.

Результати виконання:

1. На мові Java була розроблена програма імітації руху кульок, де рух кожної кульки прораховується в окремому потоці. При збільшенні кількості потоків до декількох сотень, рух кульок сповільнюється і програма стає повільно реагувати на дії користувача. Це відбувається через те, що кожен потік намагається отримати долю ресурсів процесора, які є обмеженими. На рисунку можна побачити приклад роботи програми.

Потокові програми створюється лише з метою пришвидшення виконання обчислень. Це дозволяє у повній мірі використовувати обчислювальні ресурси системи.

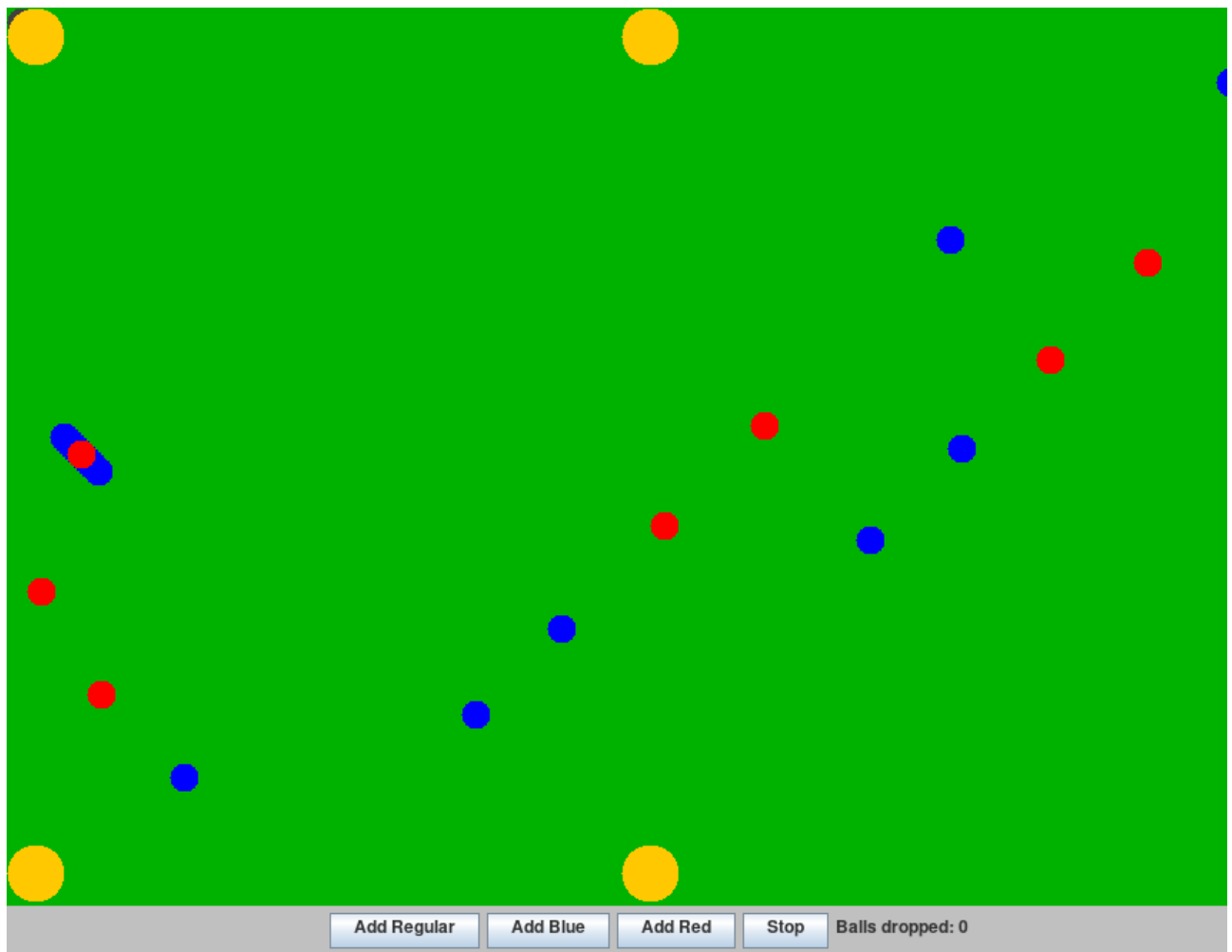


Рисунок 1 — робота програми

2. Метод `move` кульки повертає `boolean`, що означає чи впала кулька у лузу. Відповідний потік перевіряє це значення та завершує свій цикл, якщо це значення є `true`. Кількість кульок, що впали відображається в інтерфейсі програми.

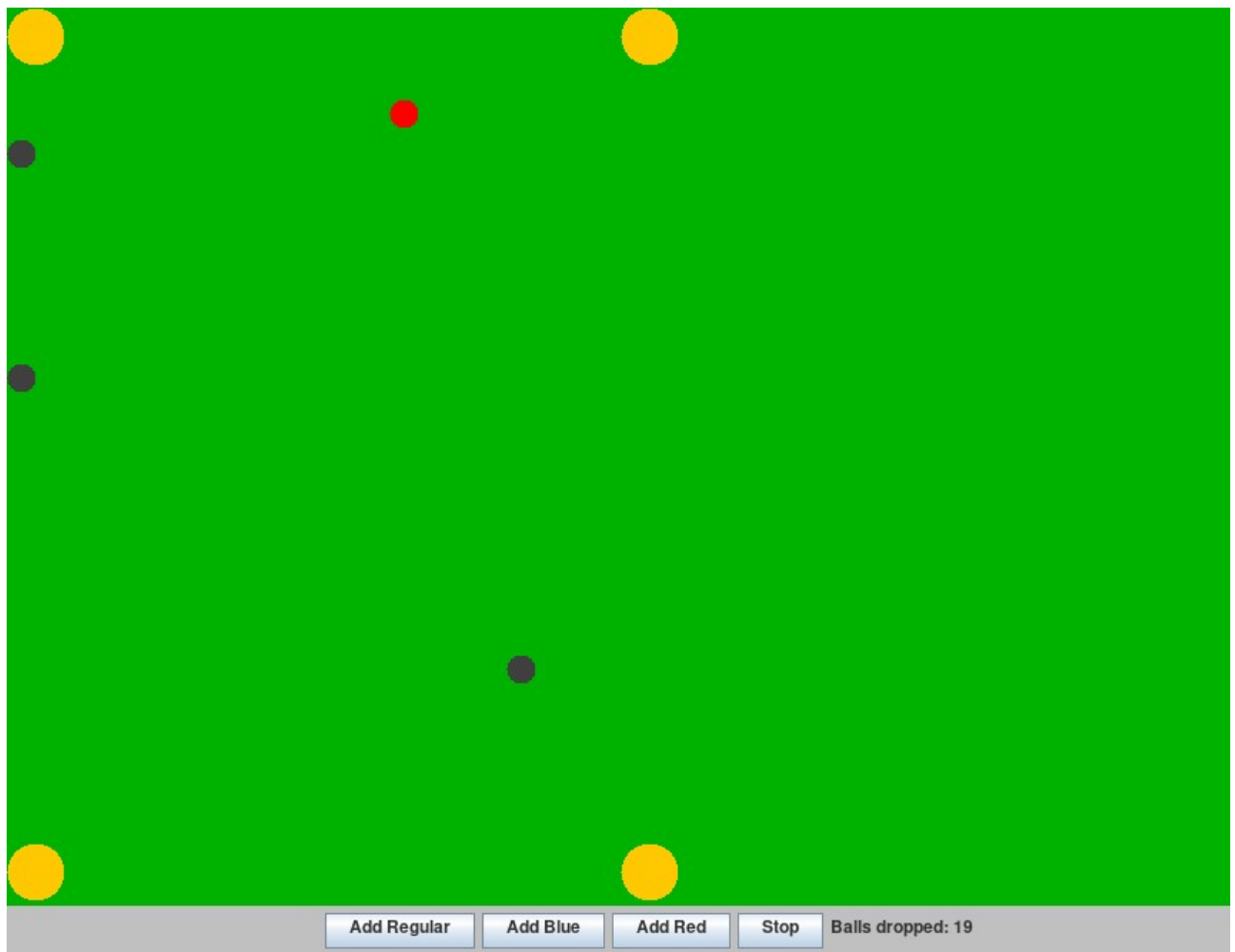


Рисунок 2 — демонстрація кульок, що впали

3. Червоні кульки створюються з пріоритетом потоку 10, що є максимальним, а сині з мінімальним пріоритетом 1. Планувальник потоків віддає перевагу потоку з більшим пріоритетом, тому при невеликій кількості кульок, червоні рухаються швидше синіх. При дуже великій кількості синіх кульок та одній червоній, різниці в їх русі майже немає, адже ресурс процесора ділиться між багатьма потоками і всім дістається тільки дуже маленька частка ресурсу. На рисунку можна побачити, що червона кулька рухається посередині серед синіх кульок.

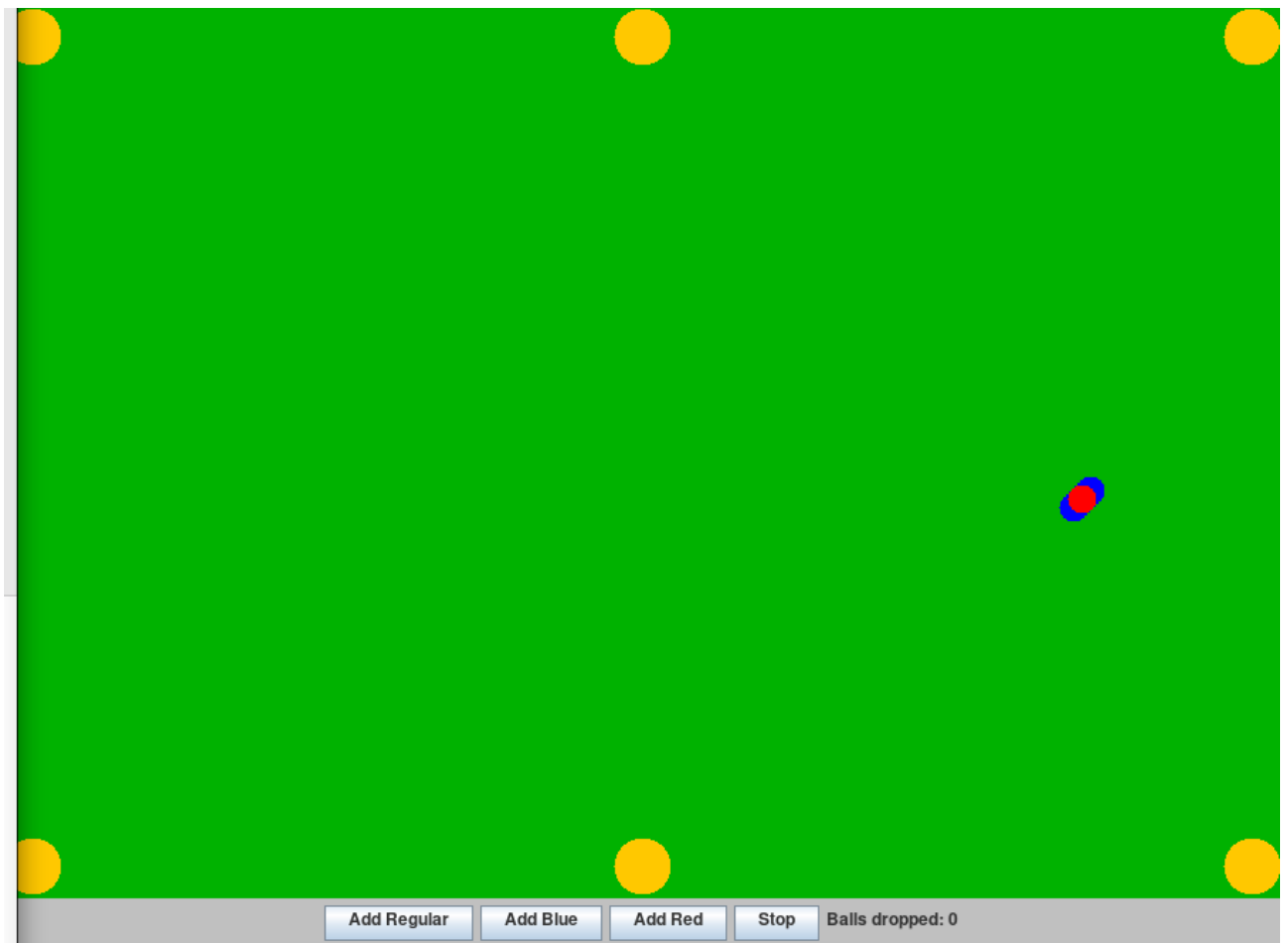


Рисунок 3 — Рух кульок з різним пріоритетом

4. Для демонстрації `join()` потоків, кульки сірого кольору чекають на завершення потоку попередньої кульки викликавши на ньому метод `join()`. Отже, сірі кульки будуть нерухомі до того моменту, коли кулька, що була створена перед ними не впаде у лузу, що можна побачити на рисунку.

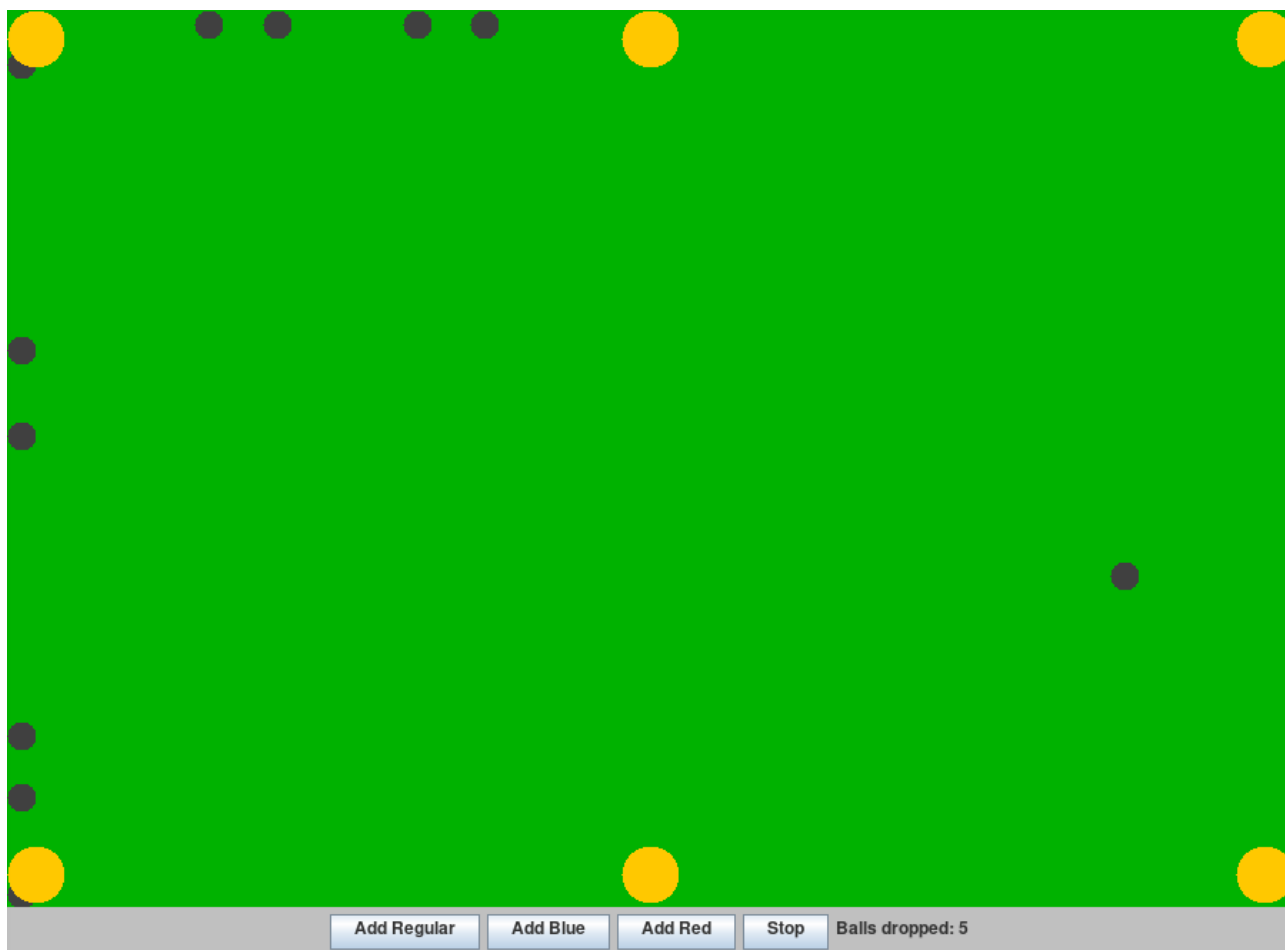


Рисунок 5 — метод join потоків

5. Після запуску двох потоків можна спостерігати хаотичне виведення двох символів, що пояснюється асинхронним виконанням двох потоків, а також роботою самого терміналу.

[illegible]

Рисунок 6 — вивід потоків без синхронізації

Синхронізація потоків досягається блоком `synchronized` і використання спільного об'єкту `lock` та його методами `lock` та `notify`. Після синхронізації бачимо чергування виведених символів.

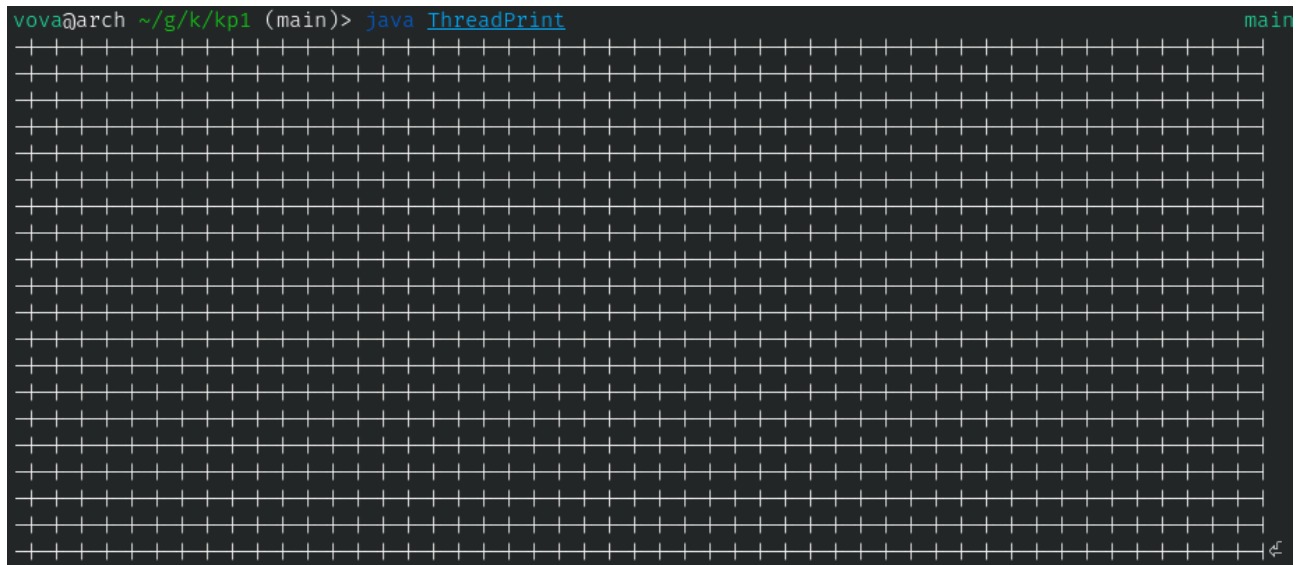


Рисунок 7 — вивід синхронізованих потоків

6. При виконанні операції без синхронізації, отримуємо значення `count`, яке не дорівнює 0, а може бути додатнім або від'ємним. Це пояснюється тим, що операція `count++` не є атомарною, а складається з трьох дій, що призводить до стану гонки, коли цю операцію виконують декілька потоків.

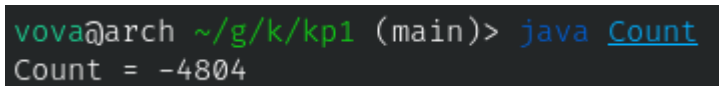


Рисунок 8 — значення `count` без синхронізації

Синхронізація була досягнута трьома способами: синхронізований метод, синхронізований блок та використання `java ReentrantLock`. Кожний з цих методів гарантує значення `count = 0` після виконання програми.

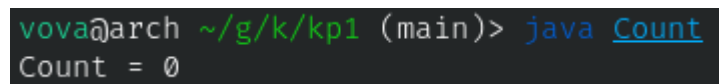


Рисунок 9 — значення `count` з синхронізацією

Висновок:

Після виконання комп'ютерного практикуму були засвоєні методи роботи з класом Thread у Java, використання пріоритетів потоків, методи синхронізації потоків.