Teoria Współbieżności

Laboratorium 1 - Współbieżność w Javie

Magdalena Pastuła

6.10.2020

1. Zadania do wykonania.

Na laboratorium do wykonania były następujące zadania:

1. Napisanie programu uruchomiającego 100 000 razy dwa wątki. Przeznaczeniem jednego z nich było zwiększanie wartości zmiennej, a drugiego zmniejszanie. Następnie należało sprawdzić wynik takiego działania.
2. Narysowanie histogramu wyniku końcowego programu z punktu 1. po wywołaniu go 100 razy.
3. Wprowadzenie własnego mechanizmu do programu z punktu 1., który pozwoliłby przewidzieć wartość końcową zmiennej, na której są wykonywane operacje.

2. Koncepcja rozwiązania.

Zadanie 3

Wprowadzony mechanizm naśladuje działanie semafora, to znaczy klasa MySemaphore zawiera zmienną typu boolean, która zawiera informację, czy jakiś wątek aktualnie posiada dostęp do modyfikowania zmiennej Counter czy nie.

3. Implementacja i wyniki.

Zadanie 1

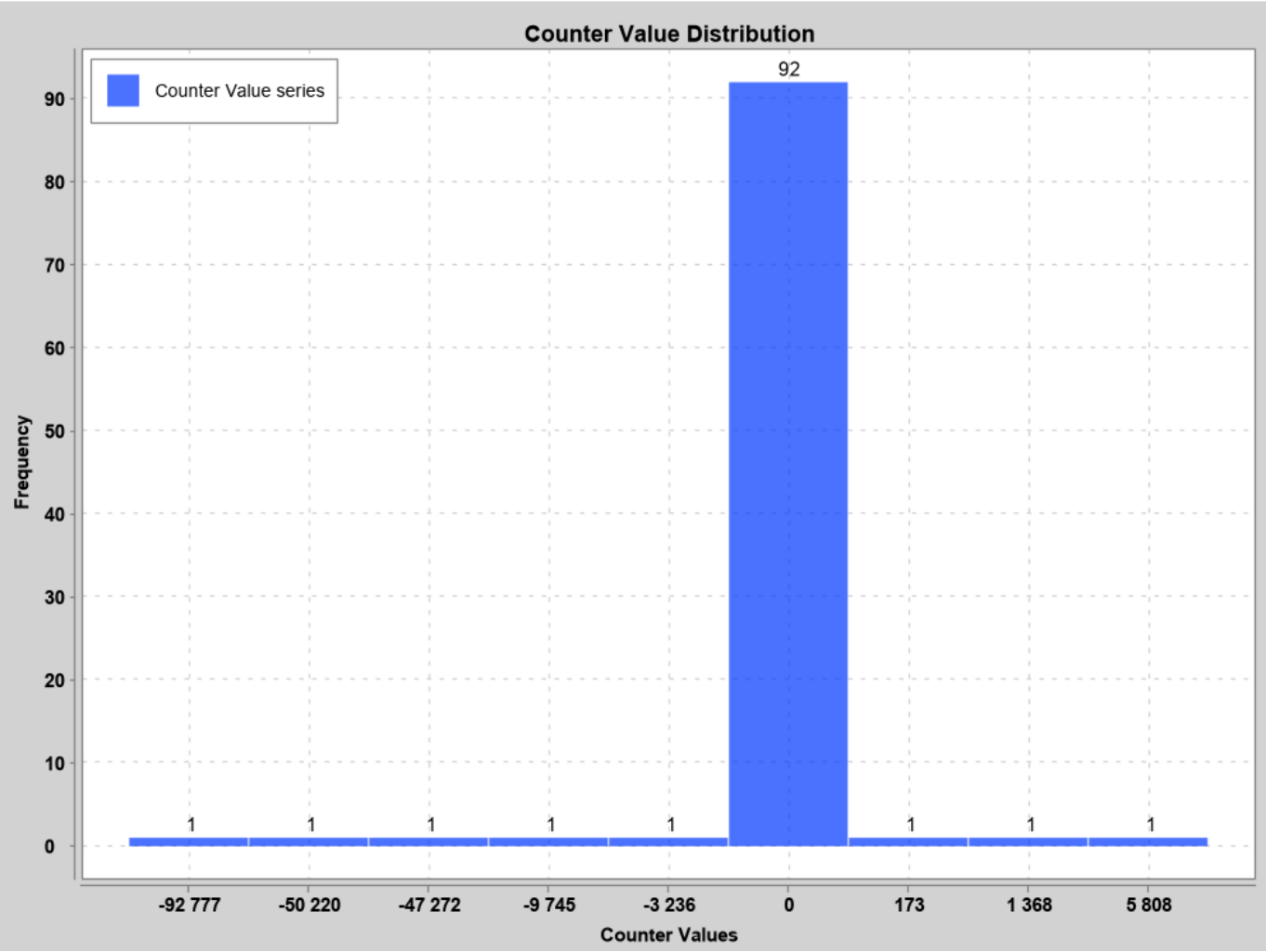
Kod zadania pierwszego znajduje się w pliku Race1.java.
Przy prawie każdym wywołaniu programu z zadania 1. wypisywana wartość końcowa jest inna i różna od zera. Przykładowe wartości końcowe zmiennej Counter:

- 47 029
- -358
- 16 064

Zadanie 2

Kod zadania drugiego znajduje się w pliku Race2.java. Do narysowania histogramu wykorzystano bibliotekę Apache oraz XChart.

Wykreślony histogram na podstawie wywołania programu z zadania 1. 100 razy:



Zadanie 3

Kod z zadania trzeciego znajduje się w pliku Race3.java.
Niestety, wprowadzony mechanizm nie wprowadził poprawy wyników programu z zadania 1.

4. Wnioski z ćwiczenia.

Wnioski wyciągnięte z poszczególnych zadań:

Zadanie 1

Niezsynchronizowane wątki przy próbie dostępu do tego samego zasobu lub zmiennej zachowują się niedeterministycznie. Wartość końcowa większa od zera oznacza, że w większej ilości przypadków wątek inkrementujący pobierał wartość zmiennej zanim wątek dekrementujący zdążył ją nadpisać. W przypadku wartości końcowej mniejszej od zera większość przypadków była odwrotna, niż w przypadku wartości końcowej dodatniej.

Zadanie 2

Histogram potwierdza stwierdzenie, że niezsynchronizowane wątki działają niedeterministycznie. Mimo, że większość wywołań programu z zadania 1. kończy się tak, jak powinna przy poprawnym działaniu wątków (wartość końcowa zero), to jednak zdarza się, że jest ona inna i świadczy o jednoczesnej próbie dostępu do zmiennej przez dwa wątki.

Zadanie 3

Problem nie został do końca rozwiązany, ponieważ podczas odczytu wartości semafora następuje ta sama sytuacja, co podczas zmiany wartości zmiennej: zanim pierwszy wątek zdąży zabrać semafor, drugi odczytał, że może dostęp do zmiennej jest wolny.

5. Bibliografia.

1. Przykład rysowania histogramu.
2. Dokumentacja HashMapy w Javie.