

Звіт
Програмний проєкт №4

Виконав Кривушко Євгеній Олександрович; Група К - 24

Умова задачі:
кількість полів $m=2$

	Поле 0		Поле 1		
варіант	read	write	read	write	string
№15	10%	5%	50%	10%	25%

Схема захисту даних : Було використано два `shared_mutex`(для кожного поля свій `mutex`) та задіяно схему читач-письменник. Операції `read` та `string` використовують `shared_lock` - блокують операції `write`, але не блокують один одного. `String` блокує два `mutex` у фіксованому порядку, щоб уникнути `deadlock`. Операції `write` використовують `unique_lock` та блокують відповідні поля, якщо поля різні, то блокування не відбувається. Було використано саме такий підхід, через його ефективність робити з випадками, коли багато операцій читання.

Табличка розмірів 3x3 з усередненими результатами виконання (Release, optimization /O2, 100 запусків, кількість операцій у кожному файлі 1,000,000)

Перший стовпчик - час(мілісекунди), коли частоти відповідають умові;

Другий - всі частоти рівні;

Третій - частоти дуже сильно не відповідають умові і не є рівними

Рядки - кількість потоків(1-3)

22.5824	31.504	38.6616
53.7028	70.0692	76.2881
94.1513	120.285	129.942

Висновки: табличка з попереднього абзацу відповідає очікуванням, оскільки використовується схема читач-письменник, то програма буде ефективно працювати за умови великої кількості операцій читання. Якщо відсоток операцій рівний, або операцій писання значно перевищують операції читання, то схема показує свою неефективність.

Що реально було зроблено самостійно: зроблено демо-код для генерації файлів з відповідними розподілами операцій, та метод заміру часу для потоків. Для побудови структури даних, були використані приклади коду з `mutex`, як натхнення. Про схему читач-письменник було прочитано на `cppreference`.