Іванова Ангеліна К-27

№9	10%	10%	10%	10%	40%	5%	15%

Для структури даних було використано три окремі м’ютекси — по одному на кожне поле. Операції read(i) та write(i) блокують лише м’ютекс, що відповідає конкретному полю i, тому доступ до інших полів залишається неконфліктним.

Тип послідовності / Потоки	1 потік	2 потік	3 потік
Частоти варіанта №9	1.18236	1.74085	2.67271
Рівні частоти	1.64638	1.70606	2.48124
Дуже інші частоти	0.03405	0.02832	0.02855

Отримані результати показують, що для послідовностей типу (а) та (б) збільшення кількості потоків не призводить до прискорення, а навпаки — час виконання зростає. Це логічно, оскільки при такому профілі навантаження накладні витрати на створення потоків, синхронізацію та блокування м’ютексів перекривають потенційний виграш від паралелізму, особливо коли значну частку становлять операції читання з одного й того самого поля та виклики string. Натомість для послідовностей типу (в) час виконання є значно меншим і при переході від одного потоку до двох і трьох спостерігається невелике, але відчутне прискорення. Це свідчить про те, що в цьому штучному сценарії навантаження на поля й типи операцій розподілено таким чином, що потоки менше заважають один одному, а блокування відпрацьовують коротко, тому паралельне виконання дає виграш. У цілому таблиця частково підтверджує очікування: структура даних і схема м’ютексів виявляються найбільш ефективними для певних типів навантажень, а в інших випадках накладні витрати на паралельність можуть переважати користь.

Самостійно було реалізовано структуру даних з трьома цілими полями та окремим м’ютексом для кожного поля, а також методи читання, запису та отримання рядкового подання з урахуванням багатопотокового доступу. Було розроблено генерацію трьох типів послідовностей дій, організовано запуск програми в одному, двох і трьох потоках, реалізовано вимірювання часу виконання та збирання результатів.