ЛАБОРАТОРНА РОБОТА №4

***Реалізація багатопоточного додатку із використанням***

***можливостей STL***

**Завдання:**

***Частина 1***

Реалізувати паралельну обробку елементів масиву відповідно до варіанту, використовуючи механізм асинхронних викликів (std::async). Для цього розбити масив на *M* однакових частин та виконувати обробку кожної з них окремо. Масив початково заповнюється випадковими значеннями.

Виміряти час обробки масиву для декількох різних значень *M* (наприклад, 1, 2, 3, 4, 8, 16), а також порівняти його для випадків послідовної та паралельної обробки (використовувати політики запуску std::launch::async та std::launch::deferred). Результати дослідження представити у вигляді таблиці у звіті. Бажано також у звіті представити короткі висновки з аналізу отриманих результатів.

Розмір масиву обрати на власний розсуд в залежності від конфігурації апаратного забезпечення системи (орієнтуватися на час роботи не менший ніж 10 мс для забезпечення точності вимірів). Тип елементів масиву обрати також самостійно.

В якості додаткового завдання – реалізувати можливість задання розміру масиву, кількості частин (для розбиття) та політики запуску користувачем.

Номер варіанту визначається відповідно до номеру за списком за наступною формулою:

***НомерВаріанту*** = (***НомерЗаСписком – 1)* mod *5 + 1***

або за таблицею:

|  |  |
| --- | --- |
| **Номер варіанту** | **Номери за списком** |
| **1** | 1, 6, 11, 16, 21, 26 |
| **2** | 2, 7, 12, 17, 22, 27 |
| **3** | 3, 8, 13, 18, 23, 28 |
| **4** | 4, 9, 14, 19, 24, 29 |
| **5** | 5, 10, 15, 20, 25, 30 |

***Варіанти завдань:***

|  |  |
| --- | --- |
| **Варіант** | **Операція** |
| 1 | Піднесення значень елементів масиву до квадрату (з перезаписом значень масиву) |
| 2 | Розрахунок суми значень елементів масиву |
| 3 | Пошук найменшого елемента масиву |
| 4 | Пошук найбільшого елемента масиву |
| 5 | Збільшення значення кожного з елементів масиву на задане константне значення (з перезаписом значень масиву) |

***Частина 2***

Реалізувати систему, що зображена на рисунку.

Генератор 1

Генератор 2

Генератор *N*

…

Процесор

Система містить наступні сутності: *N* «генераторів» та один «процесор». Генератор генерує випадкове значення та передає його процесору, після чого «засинає» на випадковий проміжок часу (діапазон значень обрати самостійно). Процесор чекає значення від генераторів та виводить їх на екран (виводячи також ідентифікатор (наприклад, порядковий номер) генератора). Кожна з сутностей (як генератори, так і процесор) має виконуватися в окремому потоці.

Реалізувати механізм завершення роботи програми (наприклад, шляхом натиснення деякої клавіші).

В якості додаткового завдання може бути реалізовано, наприклад, процедури додавання/видалення генераторів, а також конфігурування кожного з них (діапазон часу затримки, діапазон генерованих значень).

**Питання для самоперевірки:**

1. Яке призначення та які особливості роботи функції std::async?
2. Яким чином може бути визначена та для чого використовується стратегія запуску функції std::async? Які є стратегії запуску?
3. Яким чином може бути створено потік виконання засобами STL?
4. Для чого використовуються та як пов’язані між собою обіцянки (std::promise) та ф’ючерси (std::future)?
5. Які засоби синхронізації потоків представлені в STL?
6. Для чого використовуються м’ютекси та яким чином реалізована робота з ними в STL?
7. Які засоби для роботи із м’ютексами відповідно до RAII представлено в STL та які їх особливості?
8. Що таке умовні змінні (conditional variable) та як роботу із ними реалізовано в STL?