Міністерство освіти і науки України

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Фізико-технічний інститут

ПРОЕКТУВАННЯ РОЗПОДІЛЕНИХ СИСТЕМ

Лабораторна робота

Робота №2

Виконав студент гр. ФБ-31мп:

Шевченко С. Ш.

Київ – 2024

# Загальна інформація:

[Репозиторій](https://github.com/dedkuzmich/distributed-systems-design-2024) Github з кодом.

# 1) Інсталяція Hazelcast:

Встановимо [JDK 21](https://www.oracle.com/java/technologies/downloads/#jdk21-windows) (C:\Program Files\Java\jdk-21)

Встановимо [Hazelcast Platform v5.4.0](https://hazelcast.com/community-edition-projects/downloads/) (C:\tools\hazelcast-5.4.0)

Створимо системну змінну:

JAVA\_HOME C:\Program Files\Java\jdk-21

Додамо 2 значення у системну змінну Path:

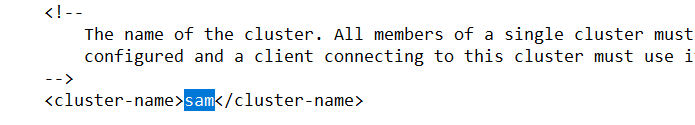
%JAVA\_HOME%\bin

C:\tools\hazelcast-5.4.0\bin

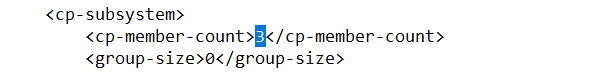
Налаштуємо сервер Hazelcast:

**$ notepad C:\tools\hazelcast-5.4.0\config\hazelcast.xml**

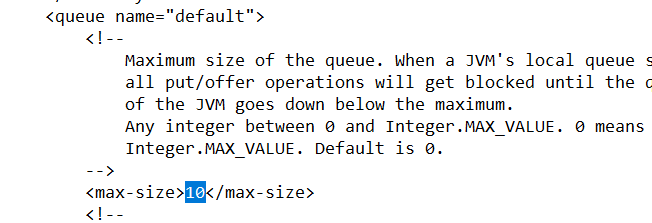
Змінимо ім'я кластера на "sam":



Увімкнемо підтримку CP Sysbsystem. Значення cp-member-count має бути > 0:



Обмежимо максимальний розмір черги 10-ма елементами:



# 2) Запуск 3 нод Hazelcast:

Створимо 7 вкладок (по 1 для кожного скрипту) в Windows Terminal за допомогою PowerShell 7 скрипту tab.ps1:

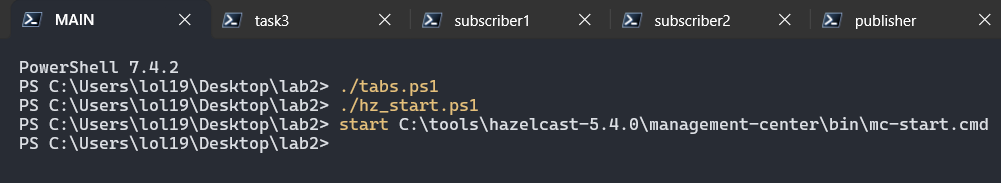
**$ ./tabs.ps1**

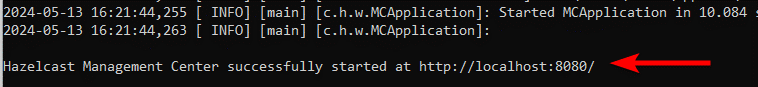
Запустимо 3 ноди Hazelcast:

**$ ./hz\_start.ps1**

Запустимо центр керування Hazelcast:

**$ start C:\tools\hazelcast-5.4.0\management-center\bin\mc-start.cmd**



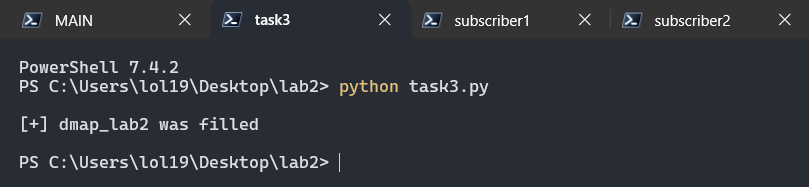


Центр керування доступний за адресою: <http://localhost:8080/>

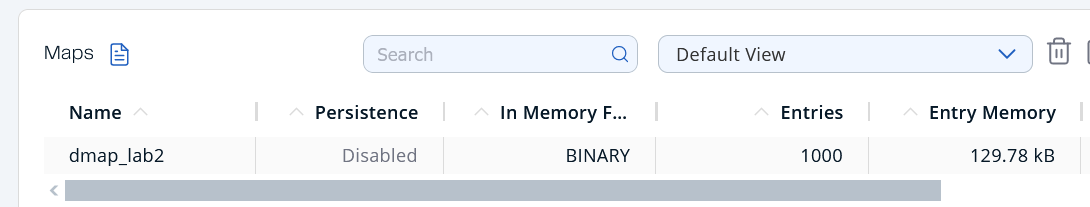
# 3) Distributed Map:

Запустимо скрипт:

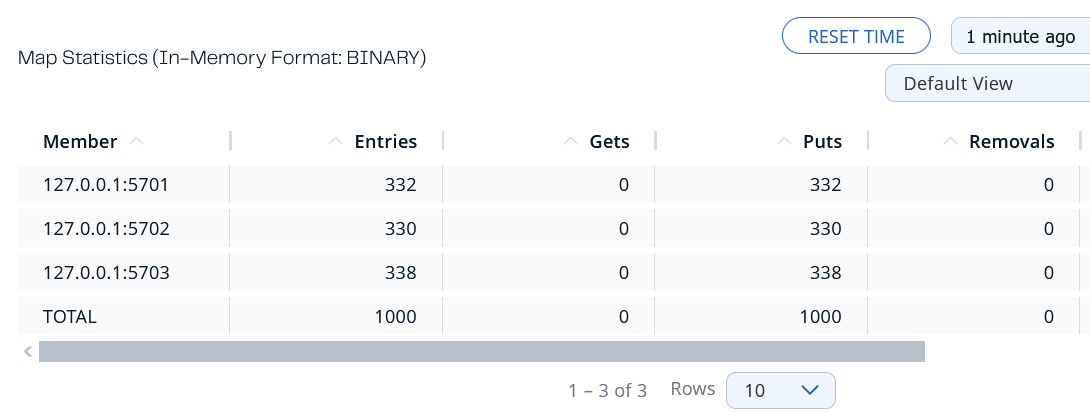
**$ python task3.py**



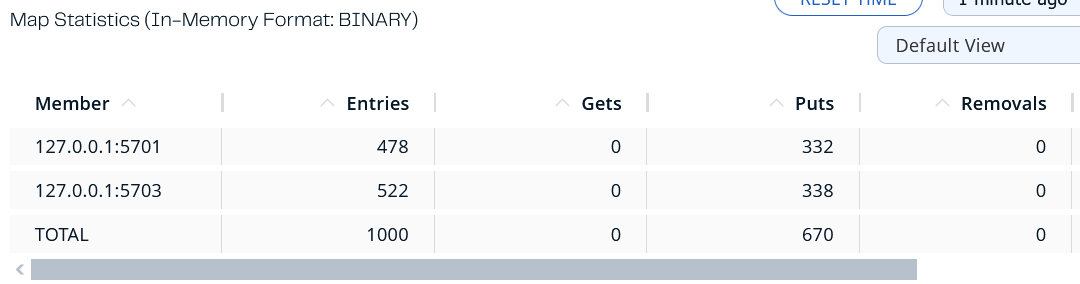
В центрі керування можемо побачити мапу, що містить 1000 значень:



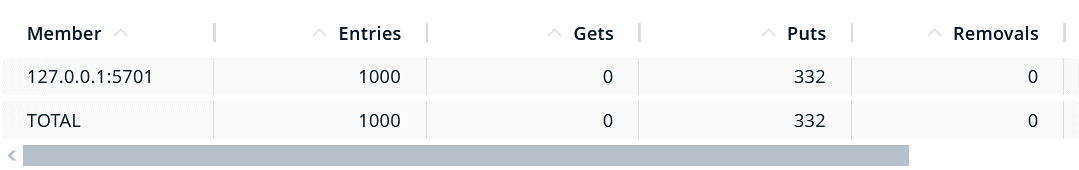
Розподіл значень по нодах якщо включені всі 3 ноди:



Розподіл значень по нодах, якщо відключити 1 ноду:



Розподіл значень по нодах, якщо відключити 2 ноди:



*- Чи буде втрата даних?*

Дані не втрачаються через реплікацію.

# 4) Topic:

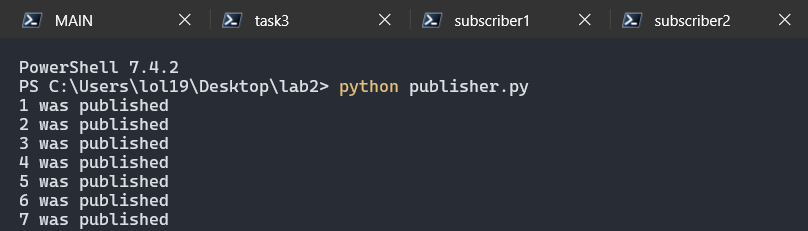
Запустимо скрипти в такій послідовності у відповідних вкладках:

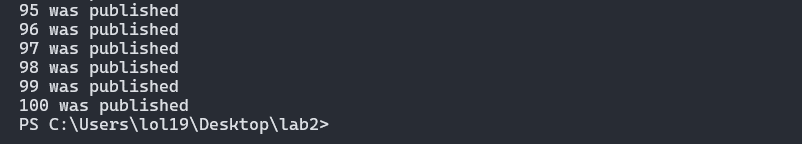
**$ python subscriber.py**

**$ python subscriber.py**

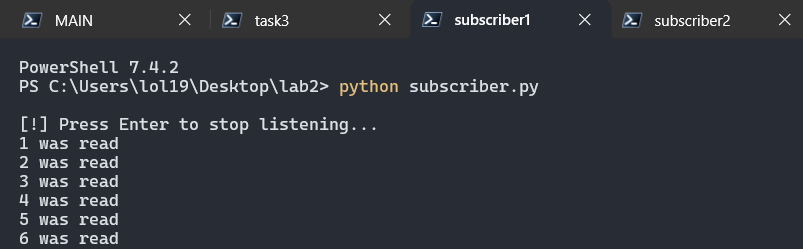
**$ python publisher.py**

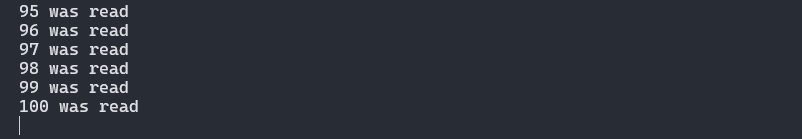
Publisher записує 100 значень в Topic:



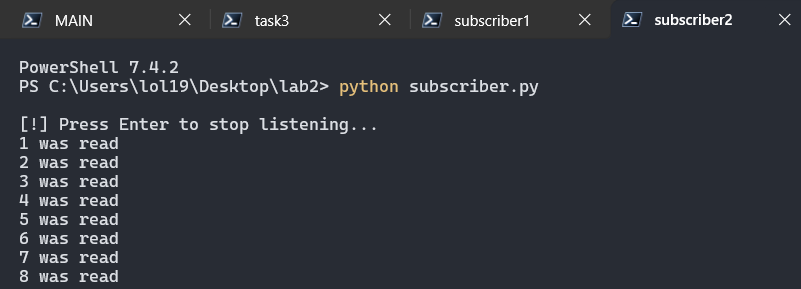


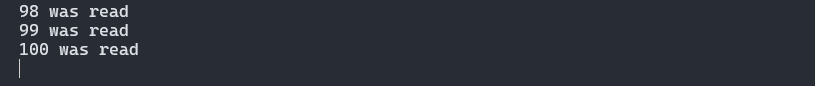
Subscriber1 зчитує 100 значень з Topic:



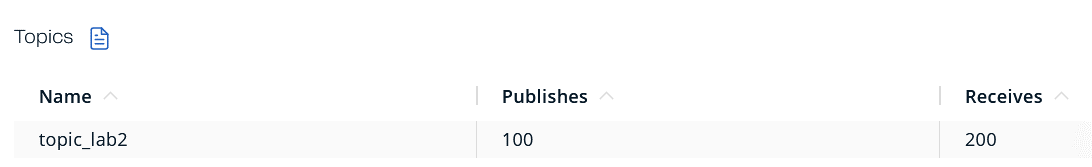


Subscriber2 зчитує 100 значень з Topic:





В центрі керування можемо побачити Topic:



*- Яким чином будуть вичитуватись значення з Topic двома клієнтами?*

Hazelcast використовує модель "publish and listen". Коли повідомлення публікується в Topic, воно надсилається всім підписникам (слухачам) цього Topic. Якщо два клієнти підписані на один і той же Topic, обидва отримають копію кожного опублікованого повідомлення.

*- Якщо один з читачів буде певний час неактивний, чи отримає він повідомлення які він пропустив?*

Опубліковані повідомлення не зберігаються. Якщо підписник не активний в момент публікації повідомлення, то він не отримає цього повідомлення.

# 5) Bounded Queue

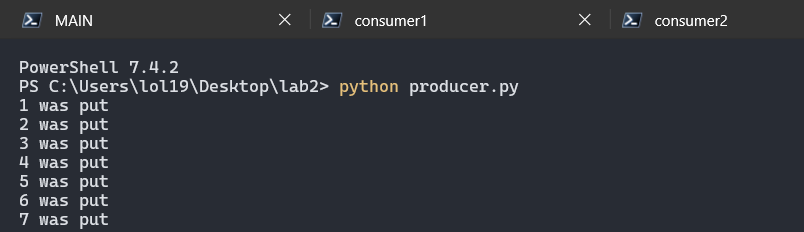
Запустимо скрипти в такій послідовності у відповідних вкладках:

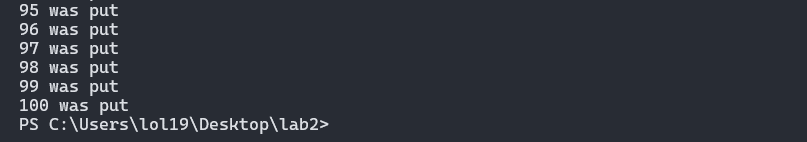
**$ python consumer.py**

**$ python consumer.py**

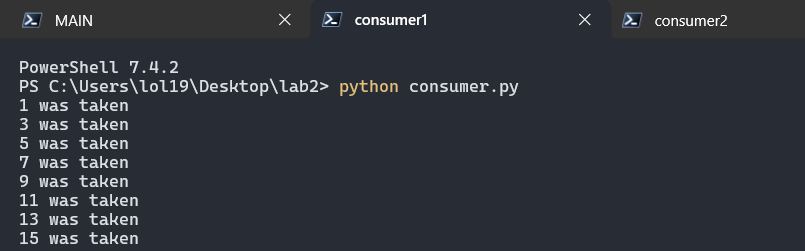
**$ python producer.py**

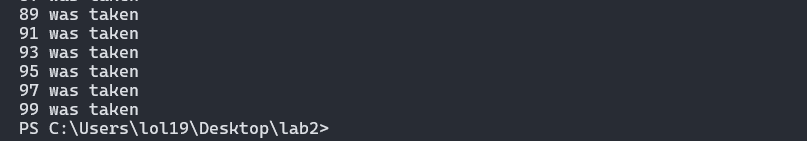
Producer записує 100 значень в чергу:



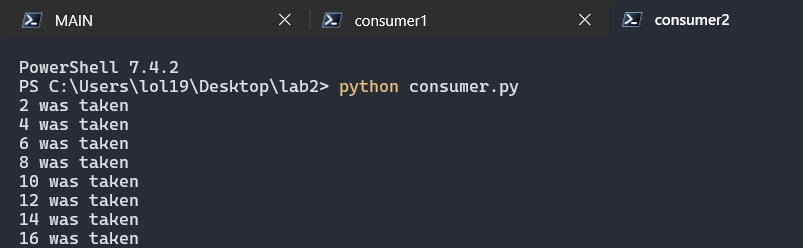


Consumer1 зчитує 50 непарних значень із черги:





Consumer2 зчитує 50 парних значень із черги:





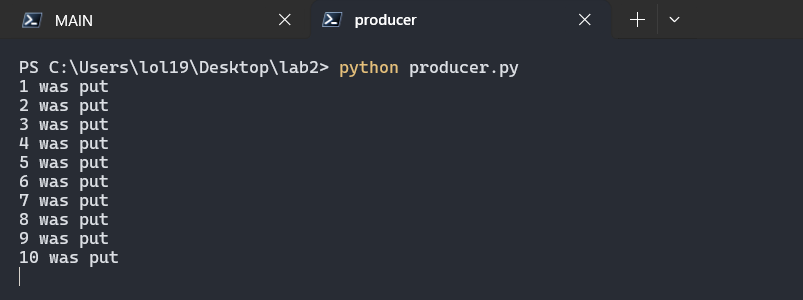
*- Яким чином будуть вичитуватись значення з черги двома клієнтами?*

Якщо 2 клієнти спробують вичитати значення з однієї черги, вони будуть отримувати повідомлення по черзі.

Consumer1 зчитує повідомлення з черги – це повідомлення видаляється з черги. Consumer2 зчитує наступне повідомлення – це повідомлення також видаляється.

Таким чином, кожне повідомлення в черзі обробляється лише один раз.

*- Перевірте яка буде поведінка на запис якщо відсутнє читання, і черга заповнена*



Запис просто зупиниться після 10-го значення. Переповнення черги не відбувається.

В цьому можна переконатися в центрі керування:

