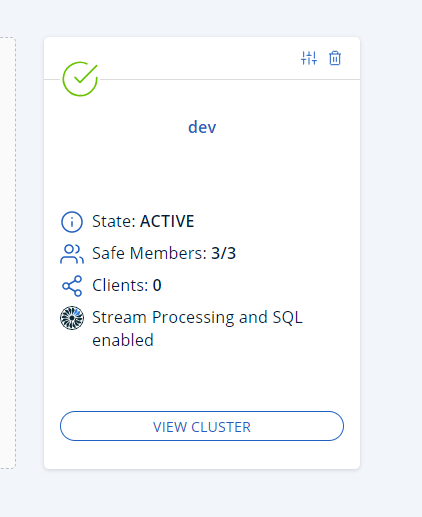
**Task 2**

**Розгортання і робота з distributed in-memory data structures на основі Hazelcast:** **Distributed Map**

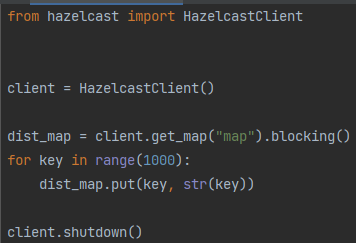
Кластер:



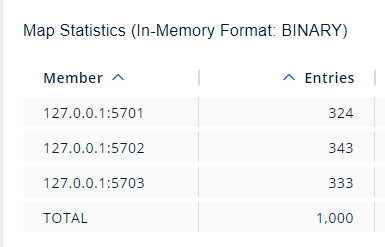
1. Продемонструйте роботу Distributed Map

<https://docs.hazelcast.com/hazelcast/5.3/data-structures/creating-a-map>

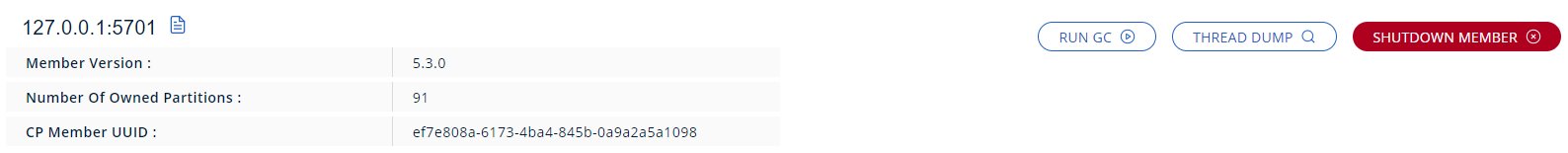
* використовуючи API створіть Distributed Map
* запишіть в неї 1000 значень з ключем від 0 до 1к
* за допомогою Management Center (<https://docs.hazelcast.com/management-center/5.3/getting-started/install#before-you-begin>) подивиться на розподіл значень по нодах
* подивитись як зміниться розподіл даних по нодах:
  + якщо відключити одну ноду
  + відключити дві ноди.
  + Чи буде втрата даних?
  + Яким чином зробити щоб не було втрати даних?

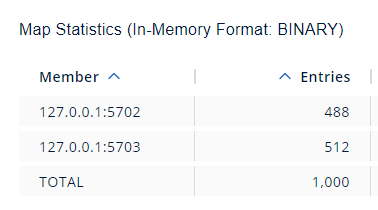


Розподіл значень по нодах після виконання коду:



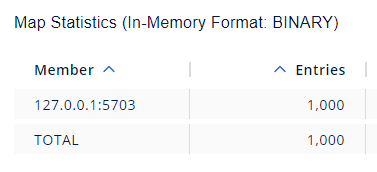
Розподіл значень по нодах після відключення першої ноди:





Розподіл значень після відключення другої ноди:

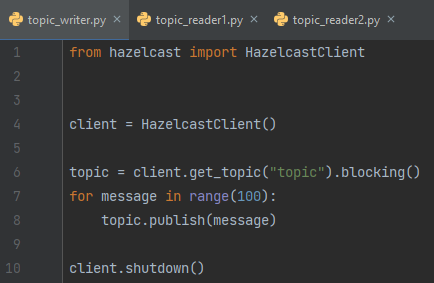


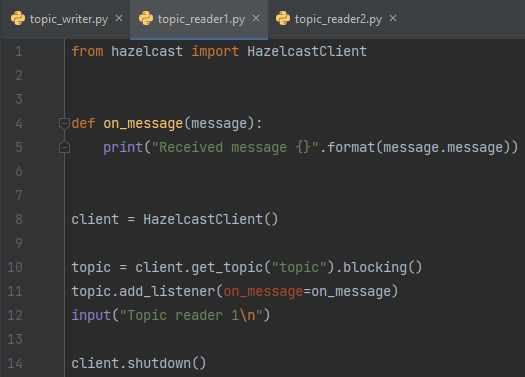


Втрат значень немає

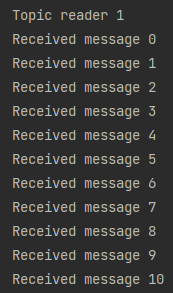
1. Продемонструйте роботу з Topic <https://docs.hazelcast.com/hazelcast/5.3/data-structures/topic>

* запустіть одного клієнта який буде писати в Topic значення 1..100, а двох інших які будуть читати з Topic
* яким чином будуть вичитуватись значення з Topic двома клієнтами?
* якщо один з читачів буде певний час неактивний, чи отримає він повідомлення які він пропустив?



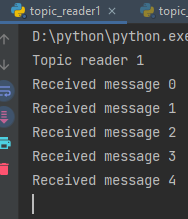
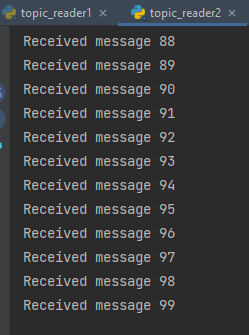


Спочатку запускаються читачі, після чого запускається той, хто буде писати. Результат виконання:



Тепер першому читачу додамо 3 секундну затримку для імітування його неактивності

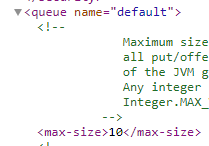
Читач 2 отримав всі повідомлення одразу, у той час як читач 1 продовжує отримувати повідомлення з найпершого:

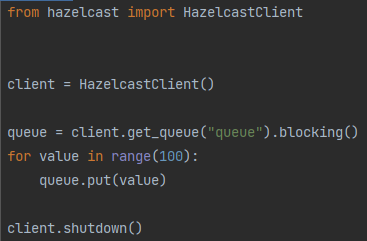


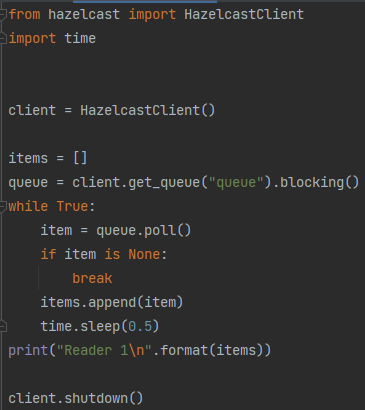
1. Робота з Bounded queue

* на основі Distributed Queue (<https://docs.hazelcast.com/hazelcast/5.3/data-structures/queue#creating-an-example-queue>) налаштуйте Bounded queue на 10 елементів (<https://docs.hazelcast.com/hazelcast/5.3/data-structures/queue#configuring-queue>)
* запустіть одного клієнта який буде писати в чергу значення 1..100, а двох інших які будуть читати з черги
* яким чином будуть вичитуватись значення з черги двома клієнтами?
* перевірте яка буде поведінка на запис якщо відсутнє читання, і черга заповнена

В конфігураційному файлі обмежили максимальний розмір черги:







Результат виконаня:





При відсутності читання з заповненої черги клієнт, що пише в чергу, буде очікувати до тих пір, поки в черзі не звільниться місце для додавання нового елементу і поки він не додасть всі вказані елементи в чергу.