

**Київський політехнічний інститут імені Ігоря Сікорського**  
**Фізико-технічний інститут**

**Проектування розподілених систем**  
**Лабораторна робота №2**  
Варіант 1

**Виконала:**  
Студентка групи ФБ-42мп  
Алькова Аліна

## Task 2

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

#### Завдання:

1. Встановити і налаштувати Hazelcast <https://hazelcast.com/open-source-projects/downloads/>
2. Сконфігурувати і запустити 3 ноди (інстанси) об'єднані в кластер або як частину Java-застосування, або як окремі застосування <https://docs.hazelcast.com/hazelcast/5.3/getting-started/get-started-binary#step-6-scale-your-cluster>

Попередньо сконфігуровано та запущено три ноди:

```
Windows PowerShell
PS D:\Documents\dist_systems\lab2> docker-compose --version
Docker Compose version v2.33.1-desktop.1
PS D:\Documents\dist_systems\lab2> docker-compose up -d
time="2025-03-25T18:03:58+02:00" level=warning msg="D:\Documents\dist_systems\lab2\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 17/17
  ✓ hazelcast-3 Pulled          33.6s
  ✓ hazelcast-2 Pulled          33.6s
  ✓ hazelcast-1 Pulled          33.6s
  ✓ 8a49fdb3b6a5 Pull complete 1.0s
  ✓ ae208aa222ae Pull complete 1.0s
  ✓ 706814bdbc39 Pull complete 29.6s
  ✓ fd653b4a11f6 Pull complete 29.7s
  ✓ 4f4fb700ef54 Pull complete 34.2s
  ✓ 00b7cdcb75ea Pull complete 29.7s
  ✓ management-center Pulled   39.1s
  ✓ 4d5d1cbd7ece Pull complete 6.8s
  ✓ 14dca4ce0c0d Pull complete 6.8s
  ✓ e40e0bb2b475 Pull complete 6.9s
  ✓ 5961e33a3161 Pull complete 34.1s
  ✓ 833a96e28ecf Pull complete 35.2s
  ✓ 7ab485b55ff9 Pull complete 35.2s
  ✓ 4226a4455703 Pull complete 35.2s
[+] Running 5/5
  ✓ Network lab2_hazelcast-network      Created      0.0s
  ✓ Container lab2-hazelcast-2-1        Started       1.0s
  ✓ Container lab2-hazelcast-3-1        Started       1.2s
  ✓ Container lab2-hazelcast-1-1        Started       1.1s
  ✓ Container lab2-management-center-1  Started       1.2s
PS D:\Documents\dist_systems\lab2>
```

#### Containers

View all your running containers and applications. [Learn more](#)

12.90% / 1.200% (12 CPUs available) 1.31GB / 0.35GB

☒ Only show running containers

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	management-center-1	081090ca7b83	hazelcast/management-ce	8080:8080	21.64%	2 minutes ago	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/>	hazelcast-1-1	57d1d2e66832	hazelcast/hazelcast:5.3	5701:5701	1.39%	2 minutes ago	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/>	hazelcast-3-1	a2aad4b3cbab	hazelcast/hazelcast:5.3	5703:5701	3.58%	2 minutes ago	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/>	hazelcast-2-1	62d3ebb661f8	hazelcast/hazelcast:5.3	5702:5701	3.36%	2 minutes ago	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Showing 5 items

**my-cluster**

State: **ACTIVE**

Safe Members: **3/3**

Clients: **0**

Stream Processing and SQL enabled

VIEW CLUSTER

Members

Search Default View

Member ^	Addi... ^	Scrip... ^	Cons... ^	Stre... ^	Slow... ^	Haze... ^	Owned Partitions	OS Committed ...	OS C...
172.18.0.2:5701	No Info	Disabled	Disabled	Enabled	No	5.3.8	91	7.20 GB	2.49 %
172.18.0.3:5701	No Info	Disabled	Disabled	Enabled	No	5.3.8	90	7.20 GB	2.21 %
172.18.0.4:5701	No Info	Disabled	Disabled	Enabled	No	5.3.8	90	7.20 GB	2.21 %

1 - 3 of 3 Rows 10

### 3. Продемонструйте роботу Distributed Map

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

- використовуючи API на будь-якій мові яка має клієнт для Hazelcast, створіть Distributed Map
- запишіть в неї 1000 значень з ключами від 0 до 1000
- за допомогою Management Center (<https://docs.hazelcast.com/management-center/5.3/getting-started/install#before-you-begin>) подивиться на розподіл ключів по нодах

```

PS D:\Documents\dist_systems\lab2> python map_demo.py
Записано 1000 значень у Distributed Map
Розмір мапи: 1000
PS D:\Documents\dist_systems\lab2>

```

Map Statistics (In-Memory Format: BINARY) RESET TIME 1 minute ago → now Default View

Member ^	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.18.0.2:5701	357	0	0	0	0	47.33 kB	0
172.18.0.3:5701	314	0	0	0	0	41.63 kB	0
172.18.0.4:5701	329	0	1 000	0	0	43.63 kB	0
TOTAL	1 000	0	1 000	0	0	132.60 kB	0

1 - 3 of 3 Rows 10

- подивитись як зміниться розподіл даних по нодах якщо відключити одну ноду (результати мають бути у протоколі)

```
PS D:\Documents\dist_systems\lab2> docker-compose stop hazelcast-3
time="2025-03-25T19:12:10+02:00" level=warning msg="D:\\Documents\\dist_systems\\lab2\\docker-compose.yml: the attribute
'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Stopping 1/1
✓Container lab2-hazelcast-3-1 Stopped
PS D:\Documents\dist_systems\lab2>
```

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.18.0.2:5701	527	0	0	0	0	69.88 kB	0
172.18.0.4:5701	473	0	1 000	0	0	62.72 kB	0
TOTAL	1 000	0	1 000	0	0	132.60 kB	0

Дані перерозподілилися між двома активними нодами. Дані не втрачаються, оскільки за замовчуванням є одна резервна копія (backup).

- відключити послідовно дві ноди (результати мають бути у протоколі)

```
PS D:\Documents\dist_systems\lab2> docker-compose stop hazelcast-2
time="2025-03-25T19:13:10+02:00" level=warning msg="D:\\Documents\\dist_systems\\lab2\\docker-compose.yml: the attribute
'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Stopping 1/1
✓Container lab2-hazelcast-2-1 Stopped
PS D:\Documents\dist_systems\lab2>
```

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.18.0.4:5701	1 000	0	1 000	0	0	132.60 kB	0
TOTAL	1 000	0	1 000	0	0	132.60 kB	0

Залишиться одна нода (hazelcast-1). У Management Center видно, що всі дані доступні на останній ноді.

- відключити одночасно дві ноди (емулюючи “падіння” серверів, чи використовуючи команду **kill -9**) (результати мають бути у протоколі). Чи буде втрата даних? Яким чином зробити щоб не було втрати даних?

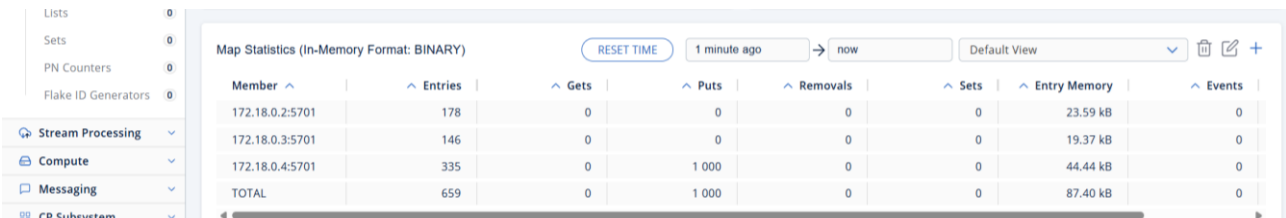
```
PS D:\Documents\dist_systems\lab2> docker-compose kill hazelcast-2 hazelcast-3
time="2025-03-25T19:14:43+02:00" level=warning msg="D:\\Documents\\dist_systems\\lab2\\docker-compose.yml: the attribute
'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Killing 2/2
✓Container lab2-hazelcast-2-1 Killed
✓Container lab2-hazelcast-3-1 Killed
PS D:\Documents\dist_systems\lab2>
```

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.18.0.4:5701	659	0	1 000	0	0	87.40 kB	0
TOTAL	659	0	1 000	0	0	87.40 kB	0

Дані втрачено

І після відновлення нод вони не повертаються:

```
PS D:\Documents\dist_systems\lab2> docker-compose start hazelcast-3
time="2025-03-25T19:15:52+02:00" level=warning msg="D:\\Documents\\dist_systems\\lab2\\docker-compose.yml: the attribute
'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 1/1
  ✓ Container lab2-hazelcast-3-1 Started 0.3s
PS D:\Documents\dist_systems\lab2> docker-compose start hazelcast-2
time="2025-03-25T19:15:55+02:00" level=warning msg="D:\\Documents\\dist_systems\\lab2\\docker-compose.yml: the attribute
'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 1/1
  ✓ Container lab2-hazelcast-2-1 Started 0.3s
PS D:\Documents\dist_systems\lab2>
```



The screenshot shows the Hazelcast Map Statistics interface. On the left is a sidebar with navigation links: Lists, Sets, PN Counters, Flake ID Generators, Stream Processing, Compute, Messaging, and CP Subsystem. The main area displays 'Map Statistics (In-Memory Format: BINARY)' with a 'RESET TIME' button and a time range selector set to '1 minute ago' to 'now'. Below this is a table with columns: Member, Entries, Gets, Puts, Removals, Sets, Entry Memory, and Events. The table contains three rows of data for different members and a total row.

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.18.0.2:5701	178	0	0	0	0	23.59 kB	0
172.18.0.3:5701	146	0	0	0	0	19.37 kB	0
172.18.0.4:5701	335	0	1 000	0	0	44.44 kB	0
TOTAL	659	0	1 000	0	0	87.40 kB	0

За замовчуванням Hazelcast має одну резервну копію (backup-count=1). Якщо дві ноди падають одночасно, а основна копія та резервна копія були на цих нодах, дані можуть бути втрачені. Щоб уникнути втрати даних, потрібно збільшити backup-count до 2 у конфігурації Hazelcast (у файлі hazelcast.xml).

#### 4. Продемонструйте роботу Distributed Map without locks

- використовуючи 3 клієнта, на кожному з них одночасно запустіть інкремент значення для одного й того самого ключа в циклі на 10K ітерацій:

```
map.putIfAbsent("key", 0);
for ( int k = 0; k < 10_000; k++ ) {
    var value = map.get( "key" );
    value++;
    map.put( "key", value );
}
```

- подивитись яке кінцеве значення для ключа "key" буде отримано (чи вийде 30K?)

```
PS D:\Documents\dist_systems\lab2> python map_no_locks.py
----->>>
Final result: 14433, time: 34.18 seconds
Expected result: 30000
PS D:\Documents\dist_systems\lab2>
```

#### 5. Зробіть те саме з використанням песимістичним блокування та поміряйте час:

<http://docs.hazelcast.org/docs/latest/manual/html-single/index.html#locking-maps>

```
PS D:\Documents\dist_systems\lab2> python map_optimistic.py
----->>>
Final result: 30000, time: 97.27 seconds
Expected result: 30000
PS D:\Documents\dist_systems\lab2>
```

6. Зробіть те саме з використанням оптимістичним блокуванням та поміряйте час::  
<http://docs.hazelcast.org/docs/latest/manual/html-single/index.html#locking-maps>

```
PS D:\Documents\dist_systems\lab2> python map_pessimistic.py  
  
----->>>  
Final result: 30000, time: 129.29 seconds  
Expected result: 30000  
PS D:\Documents\dist_systems\lab2>
```

## 7. Порівняйте результати кожного з запусків

- для реалізації без блокувань маєте спостерігати втрату даних; для реалізації з песимістичним та оптимістичним блокуванням мають бути однакові результати
- песимістичний чи оптимістичний підхід працює швидше?

### Для No locks:

Фінальне значення менше за 30,000, відбулась втрата даних через race condition.  
Час виконання: найшвидший.

### Для оптимістичного:

Фінальне значення: 30,000 (коректно). Дані завжди коректні завдяки перевірці replace\_if\_same.  
Час виконання: середній

### Для песимістичного:

#### Pessimistic lock:

Фінальне значення: 30,000 (коректно). Дані коректні, але блокування уповільнює виконання.  
Час виконання: найповільніший

## 8. 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
  - під час вичитування, кожне повідомлення має вичитуватись одразу
- яким чином будуть вичитуватись значення з черги двома клієнтами?
- перевірте яка буде поведінка на запис якщо відсутнє читання, і черга заповнена

```

Consumer 1 read: 91, Queue size: 0
Producer wrote: 91, Queue size: 0
Consumer 2 read: 92, Queue size: 0
Producer wrote: 92, Queue size: 0
Consumer 1 read: 93, Queue size: 0
Producer wrote: 93, Queue size: 0
Consumer 2 read: 94, Queue size: 0
Producer wrote: 94, Queue size: 0
Consumer 1 read: 95, Queue size: 0
Producer wrote: 95, Queue size: 0
Consumer 2 read: 96, Queue size: 0
Producer wrote: 96, Queue size: 0
Consumer 1 read: 97, Queue size: 0
Producer wrote: 97, Queue size: 0
Consumer 2 read: 98, Queue size: 0
Producer wrote: 98, Queue size: 0
Consumer 1 read: 99, Queue size: 0
Producer wrote: 99, Queue size: 0
Consumer 2 read: 100, Queue size: 0
Producer wrote: 100, Queue size: 0
Producer finished writing
Consumer 1 finished reading
Consumer 2 finished reading

Testing behavior when queue is full without consumers
Queue cleared
Producer wrote to full test: 1, Queue size: 1
Producer wrote to full test: 2, Queue size: 2
Producer wrote to full test: 3, Queue size: 3
Producer wrote to full test: 4, Queue size: 4
Producer wrote to full test: 5, Queue size: 5
Producer wrote to full test: 6, Queue size: 6
Producer wrote to full test: 7, Queue size: 7
Producer wrote to full test: 8, Queue size: 8
Producer wrote to full test: 9, Queue size: 9
Producer wrote to full test: 10, Queue size: 10
Attempting to write 11th item (queue is full, should block)...
Blocked for 5.01 seconds as expected (operation is still blocked)

Demo completed
PS D:\Documents\dist_systems\lab2>

```

Hazelcast Distributed Queue працює за принципом "один елемент — один споживач". Кожен елемент із черги забирає один із консюмерів випадковим чином (залежить від того, хто швидше викличе `take()`). У виводі видно, що значення розподіляються між `Consumer1` і `Consumer2` хаотично.

#### **Перевірка поведінки при заповненій черзі без читання:**

У коді функція `test_full_queue` заповнює чергу до 10 елементів, потім намагається додати 11-й із таймаутом у 5 секунд. Коли черга досягає ліміту в 10 елементів, спроба додати 11-й елемент блокується, доки не буде місця (або не спрацює таймаут), що підтверджує коректність `Bounded Queue`.

#### **С привиду порядку виведення повідомлень у консолі:**

У коді продюсер і консюмери працюють у різних потоках (`producer_thread`, `consumer1_thread`, `consumer2_thread`). Потоки виконуються паралельно, і порядок виведення повідомлень у консолі залежить від планувальника потоків операційної системи. Це означає, що повідомлення `Consumer X read: N` може з'явитися перед `Producer wrote: N`, якщо консюмер швидше обробив елемент і вивів повідомлення, ніж продюсер встиг вивести своє. Незважаючи на порядок виведення, черга працює коректно: Продюсер записує елемент у чергу за допомогою `queue.put(i)`. Консюмери читають елемент за допомогою `queue.poll(timeout=1.0)`.

У Hazelcast Distributed Queue елемент не може бути прочитаний, доки він не записаний. Тобто, хоча повідомлення консюмера з'являється першим у виводі, це лише особливість паралельного виконання потоків. Фактична операція читання (`poll()`) відбувається після операції запису (`put()`), інакше консюмер не отримав би значення.