

## Task 2 Results

### 3) Продемонструйте роботу Distributed Map

При відключенні однієї ноди, дані не втрачаються, а рівномірно розподіляються по двох інших нодах:

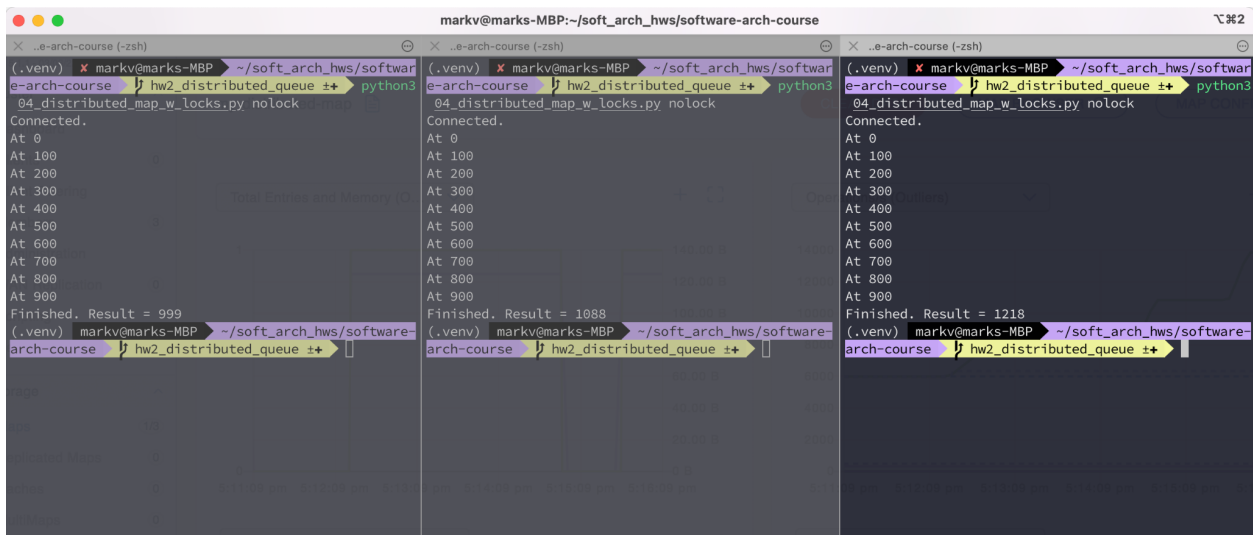
Member ^	Entries ^
127.0.0.1:5701	316
127.0.0.1:5702	340
127.0.0.1:5704	344
TOTAL	1000

Member ^	Entries ^
127.0.0.1:5701	483
127.0.0.1:5702	517
TOTAL	1000

### 4) Продемонструйте роботу Distributed Map with locks

a) no locking

При запуску скрипта 04\_distributed\_w\_locks.py одночасно з трьох клієнтів, маємо race condition:



b) pessimistic locking

## Результат вірний

```
markv@marks-MBP:~/soft_arch_hws/software-arch-course
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 04_distributed_map_w_locks.py pessimistic
Connected.
Finished, Result = 2701
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 hw2_distributed_queue.py
Connected.
Finished, Result = 2935
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 04_distributed_map_w_locks.py pessimistic
Connected.
Finished, Result = 3000
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 hw2_distributed_queue.py
Connected.
Finished, Result = 3000
```

## c) optimistic locking

## Результат також вірний

```
markv@marks-MBP:~/soft_arch_hws/software-arch-course
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 04_distributed_map_w_locks.py optimistic
Connected.
At 0
At 100
At 200
At 300
At 400
At 500
At 600
At 700
At 800
At 900
Finished, Result = 2324
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 hw2_distributed_queue.py
Connected.
At 0
At 100
At 200
At 300
At 400
At 500
At 600
At 700
At 800
At 900
Finished, Result = 3000
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 04_distributed_map_w_locks.py optimistic
Connected.
At 0
At 100
At 200
At 300
At 400
At 500
At 600
At 700
At 800
At 900
Finished, Result = 2916
(.venv) markv@marks-MBP ~/soft_arch_hws/software-arch-course python3 hw2_distributed_queue.py
Connected.
At 0
At 100
At 200
At 300
At 400
At 500
At 600
At 700
At 800
At 900
Finished, Result = 2916
```

## 5) Налаштуйте Bounded queue

```
python3 05_bounded_queue.py producer
Producing 4
Producing 5
Producing 6
Producing 7
Producing 8
Producing 9
Producing 10
Producing 11
Producing 12
Producing 13
Producing 14
Producing 15
Producing 16
Producing 17
Producing 18
Producing 19
Producing 20
Producing 21
Producing 22
Producing 23
Producing 24
Producing 25
Producing 26
Producing 27
Producing 28
Producing 29
Producing 30

python3 05_bounded_queue.py consumer
Consumed: 0
Consumed: 2
Consumed: 4
Consumed: 6
Consumed: 8
Consumed: 10
Consumed: 12
Consumed: 14
Consumed: 16
Consumed: 18
Consumed: 20
Consumed: 22
Consumed: 24
Consumed: 26
Consumed: 28
Consumed: 30

python3 05_bounded_queue.py consumer
Consumed: 1
Consumed: 3
Consumed: 5
Consumed: 7
Consumed: 9
Consumed: 11
Consumed: 13
Consumed: 15
Consumed: 17
Consumed: 19
Consumed: 21
Consumed: 23
Consumed: 25
Consumed: 27
Consumed: 29
```

- Якщо відсутнє читання і черга заповнена, виклик `.put()` блокується, поки черга не звільниться
- Якщо є декілька читачів, то значення зчитуються по черзі, причому ми використовуємо `queue.take()` -> якщо черга пуста, виклик блокується