МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**ЗВІТ**

**з лабораторної роботи № 2**

**«Практика використання сервера Redis»**

**Виконав:**

студент 3-го курсу, групи КП-83, спеціальності 121 – Інженерія програмного забезпечення

*Шаповалов Данило Дмитрович*

**Посилання:** [link](https://github.com/DeLion13/DataBases2/tree/master/lab2)

Київ – 2021

## Обгрунтування вибору структур даних redis

Для представлення користувачів в системі було обрано структуру даних SET, адже вона дозволяє за час О(1) здійснити пошук, чи є такий користувач, наприклад під час реєстрації чи аутентифікації. Так як маємо дві групи користувачів, які мають різні функціональні можливості, було вирішено створити два окремих SETа.

Так як адмін повинен мати можливість переглядати список наявних користувачів, а також їх порядок не важливий, то з цією метою також було обрано структуру даних SET.

Для того, щоби мати можливість спостерігати відсортований рейтинг користувачів відповідно до їх активності (надісланих повідомлень/спаму), було обрано структуру даних відсортований сет, тобто ZSET, адже саме він збергіає ключі у відсортованому вигляді за павним значенням.

Для зберігання даних про повідомлення, було вирішено для кожного екземпляра створювати HASH (бо він дозволяє запам'ятовувати набори ключ-значення) і в ньому зберігати інформацію про юзернейми відправника та отримувача, а також саме повідомлення. Для того, щоб генерувати для кожної такої структури унікальний ключ, було використано звичайну строку, яку кожен раз при додаванні нового повідомлення збільшуємо на одиницю.

Для журналу повідомлень було використано Pub/Sub, адже завдяки цьому будь-хто може відправляти повідомлення, які слухач відразу ж прочитає, адже він підписаний на відповідний канал. Це дає змогу створювати додатки, які в реалтаймі будуть оновлюватися, адже як тільки-но дані будуть опубліковані, клієнт зможе їх миттєво прочитати та відповідно опрацювати їх.

Для черги повідмлень було використано LIST, адже ця структура даних зберігає порядок, а операції вставляння в кінець та вилучення з початку займають О(1) часу.

Для відображення користувачу вхідних повідомлень (які надійшли саме йому) було обрано LIST, адже він дозволяє зберігати хронологічний порядок появи нових повідомлень.

Для того, щоб користувач зміг отримати статистику по статусам відправлених повідомлень, було відповідно сторено SET для кожного статусу, в яких зберігаються усі повідомлення в системі відповідно до їх статусу. Також щоб зберігати відправлені користувачем повідомлення був також створений SET. Щоби групувати, використуоються операції об'єднання множин статусів, які були обрані для того щоб їх переглянти, а потім робиться перетин утвореної множини з множиною відправлених повідомлень. Це і дає змогу швидко порахувати відповідну кількість.

## Опис умов експерименту

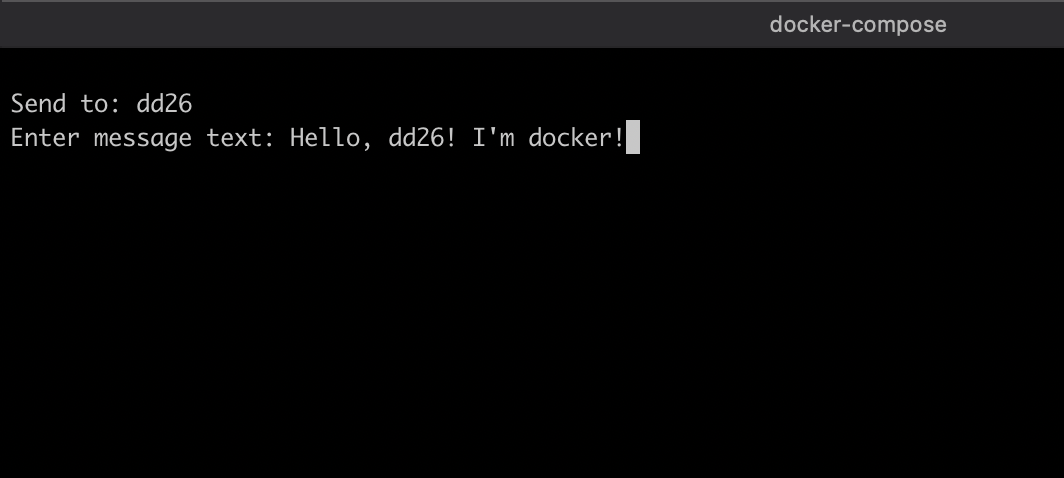
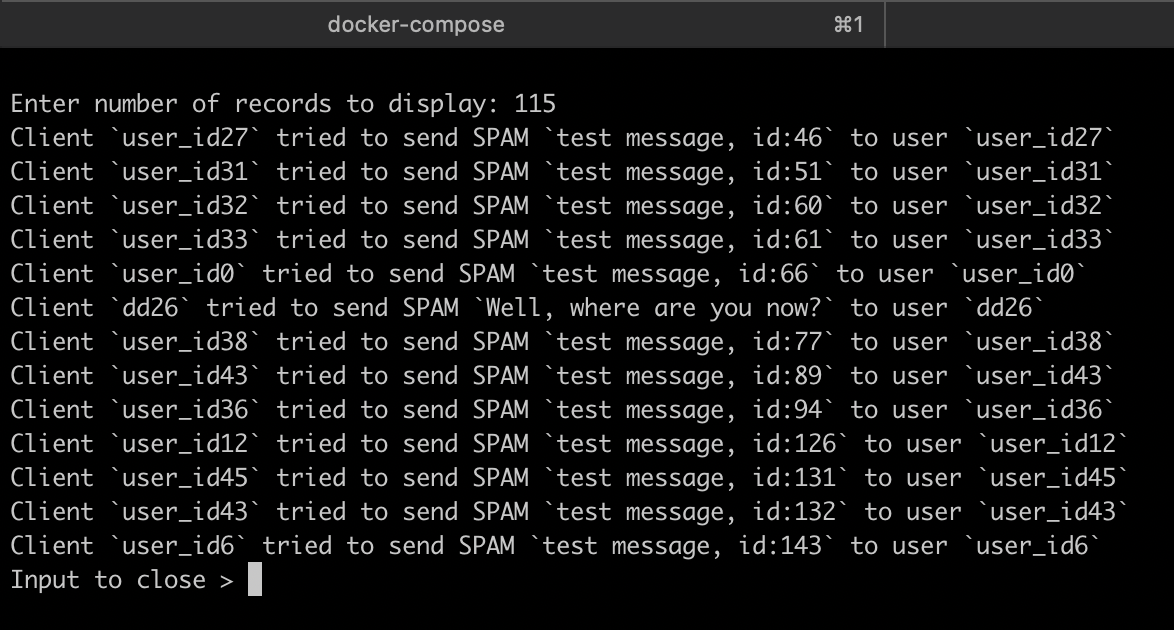
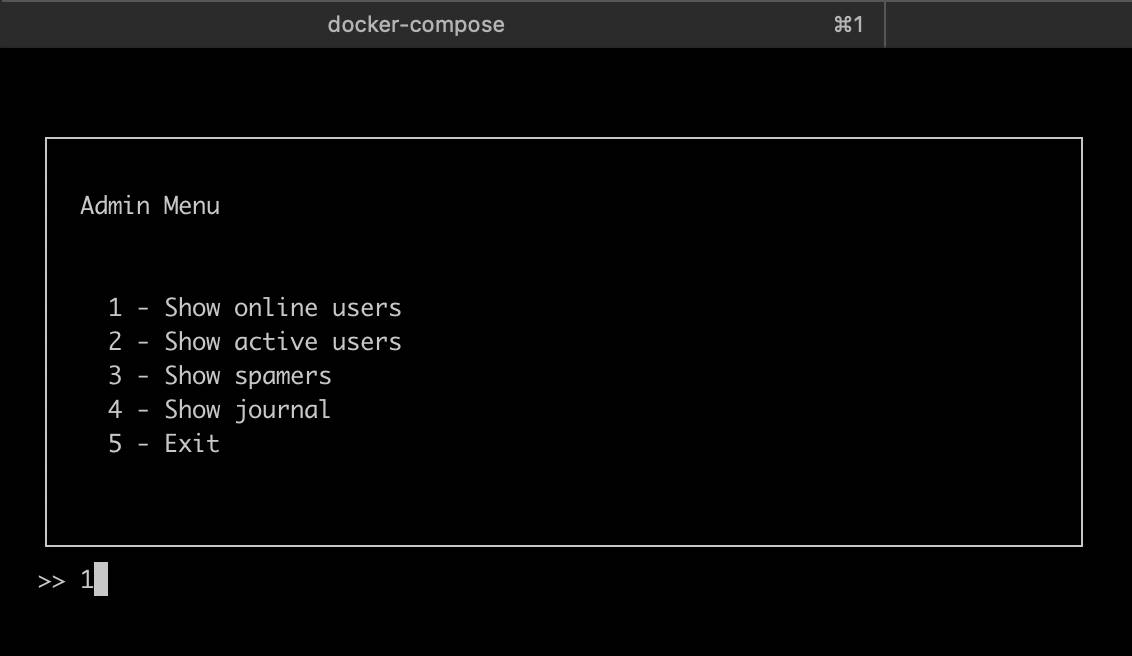
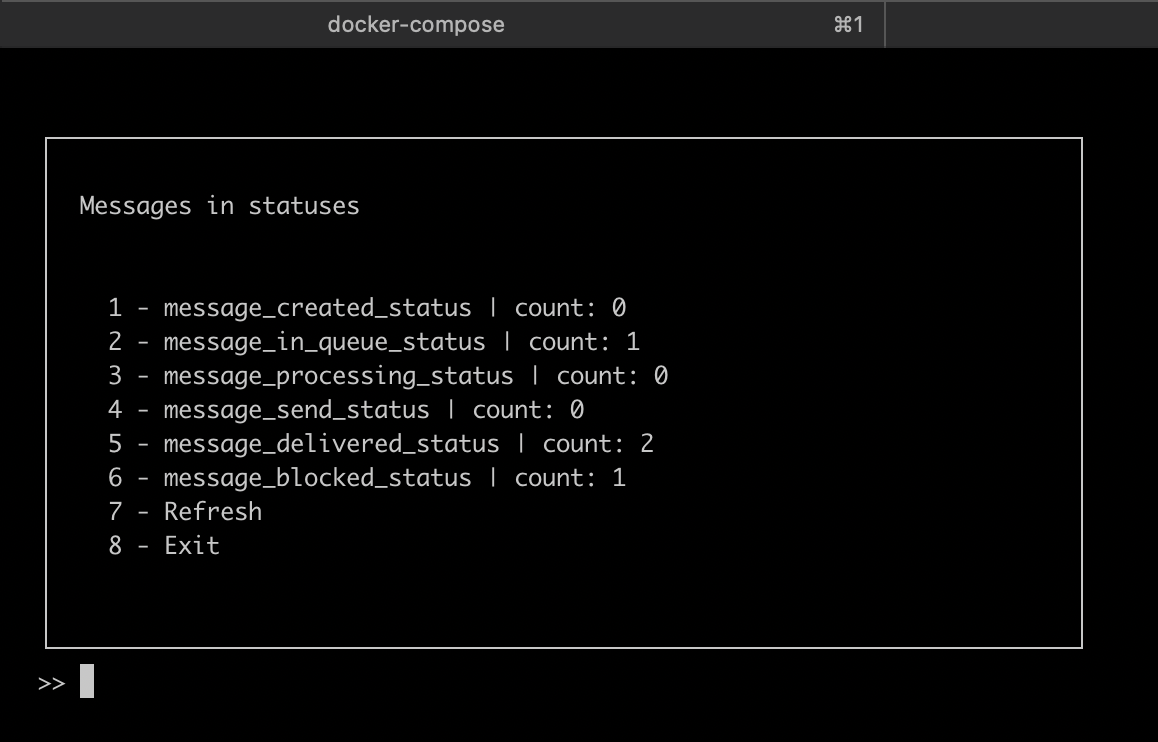
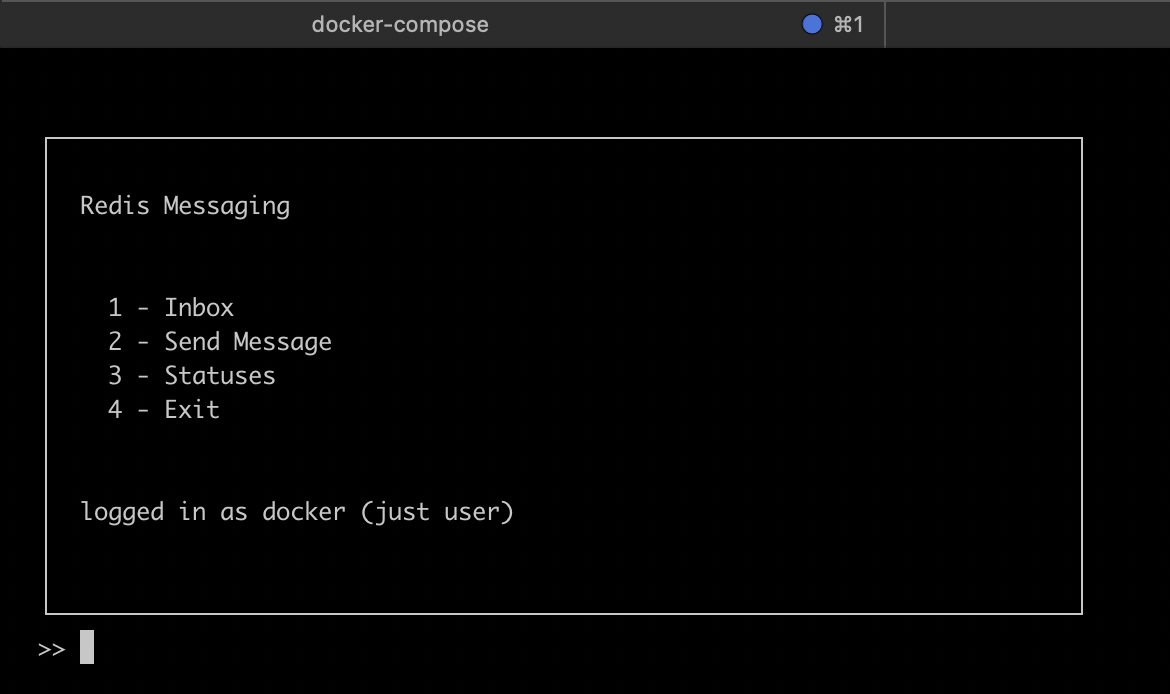
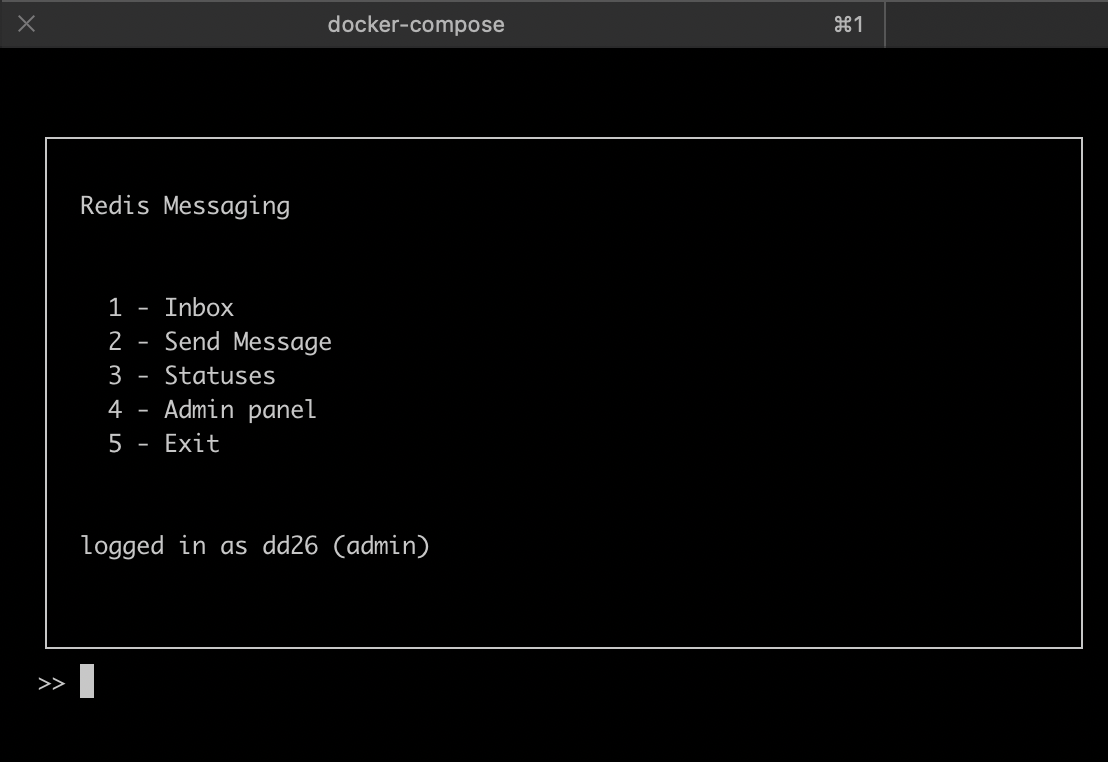
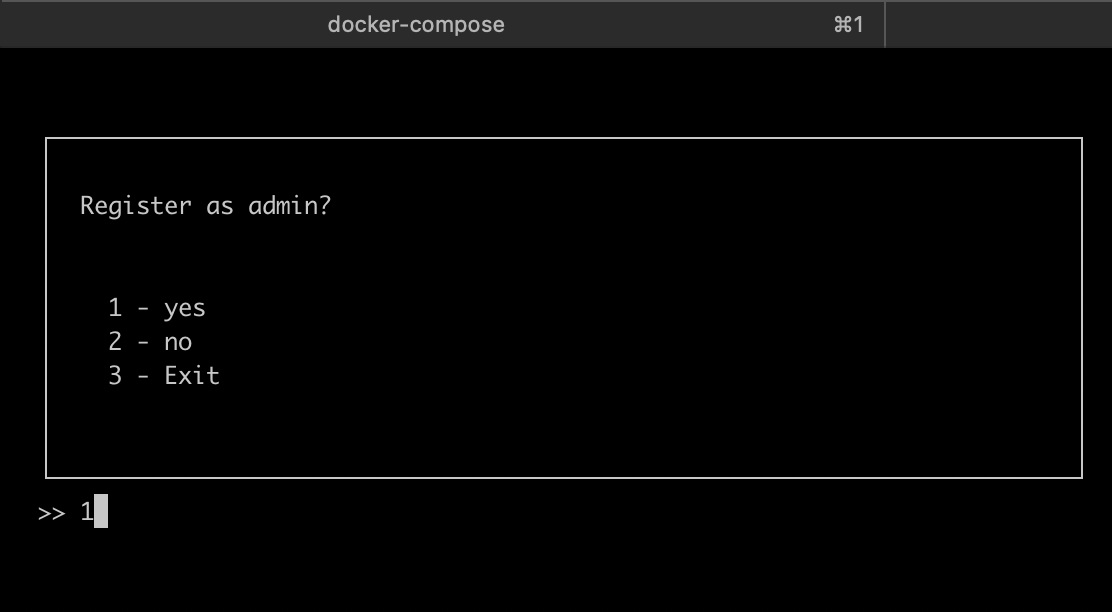
Запущено 4 програми. Виконані команди:

1. docker-compose up (запуск сервера)
2. python3 worker.py (виконувач для роботи з повідомленнями)
3. python3 emulation.py (режим емуляції)
4. python3 main.py (запуск програми з інтерфейсом)

Створено користувачів «dd26» у ролі адміністратора, та «docker» у ролі користувача. Відправлено кілька повідомлень з різних акаунтів одне одному.

Від ролі адміністратора можна подивитись різні статистики. Серед них спам, онлайн користувачі, активні користувачи, журнал.

**Скріншоти**



**Відповіді на запитання**

1. Визначити сфери застосування основних структури даних redis (List, Hash, Sorted List, Set, Pub/Sub).
   1. List – черги.
   2. Hash – об’єкти даних (сутності, що зберігають декілька атрибутів).
   3. Sorted list – підрахування score для кожного елемента із множини, побудування вісортованого за зростанням чи спаданням score списку елементі.
   4. Set – списки, у яких необхідно уникати дублювання.
   5. Pub/Sub – розсилка повідомлень.
2. Визначити основні переваги та недоліки redis.
   1. «+»:
      1. Широкий спектр типів даних
      2. Швидкість
      3. Простий в установці
      4. Активна спільнота розробників
      5. Не має залежностей
      6. Легкий в освоєнні
   2. «-»:
      1. Слабкофункціональна мова запитів.
      2. Велике використання оперативної пам’яті.

**Висновок**

При виконанні даної лабораторної роботт я навчився створювати ефективні програм, орієнтовані на використання сервера Redis за допомогою мови програмування Python.