НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра прикладної математики

Звіт
із лабораторної роботи №4
з дисципліни «Інформаційні системи»
на тему
“ Знайомство з принципом роботи брокера повідомлень”
Варіант 7

Виконав:
студент групи КМ-01
Іваник Ю. П.

Викладач:
Доцент кафедри ПМА
Саяпіна І. О.

Київ — 2023

Зміст

Завдання 1

1. Змініть код Producer'a, щоб він генерував інший текст повідомлень за Вашим вибором.
2. Змініть назву черги повідомлень як у коді Producer'a, так і в коді Consumer'a та переконайтеся, що вони використовують однакову назву черги.
3. Змініть код Producer'a, щоб генерувати повідомлення з іншим інтервалом.
4. Змініть код Consumer'a, щоб роздрукувати кількість повідомлень, які він отримав, на додаток до вмісту повідомлення.

Код для програми Producer’a:

```
def task_1_producer():
    counter = 1
    QUEUE = 'Ivanyk'

    while True:
        time_to_sleep = random.randint(500, 5000) / 1000  # 0.5 - 5 секунд
        time.sleep(time_to_sleep)

        connection =
pika.BlockingConnection(pika.ConnectionParameters('localhost'))
        channel = connection.channel()

        channel.queue_declare(queue=QUEUE, durable=False, exclusive=False,
auto_delete=False)

        message = f'Завдання 1 | Повідомлення {counter}'
        body = message.encode('utf-8')

        channel.basic_publish(exchange='', routing_key=QUEUE, body=body)

        print(f'[LOG] Повідомлення N {counter}')
        counter += 1


try:
    task_1_producer()
except KeyboardInterrupt:
    print('[LOG] Зупинено')
```

Код для програми Consumer’а:

```
import random
import time

import pika


def task_1_consumer():
    global msg_counter
    msg_counter = 1
    QUEUE = 'Ivanyk'

    connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
    channel = connection.channel()

    def callback(ch, method, properties, body):
        global msg_counter
        print(f'Отримано повідомлення N {msg_counter}: {body.decode("utf-8")}')
        msg_counter += 1

    channel.queue_declare(queue=QUEUE, durable=False, exclusive=False,
auto_delete=False)

    channel.basic_consume(queue=QUEUE, on_message_callback=callback, auto_ack=True)

    print('\n[LOG] Початок відстеження черги')
    channel.start_consuming()


try:
    task_1_consumer()
except KeyboardInterrupt:
    print('[LOG] Зупинено')
```

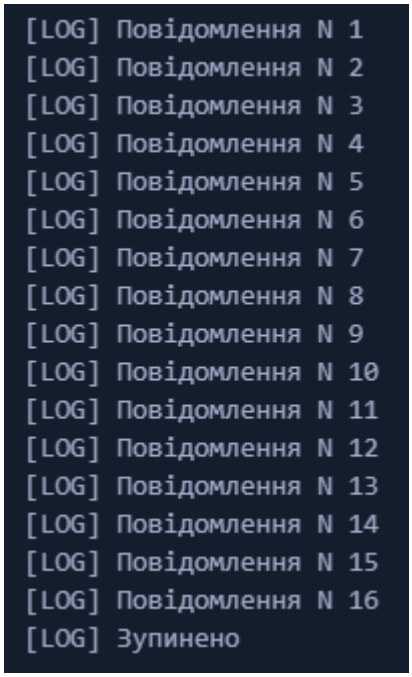

Рис. 1 – Завдання 1 Producer

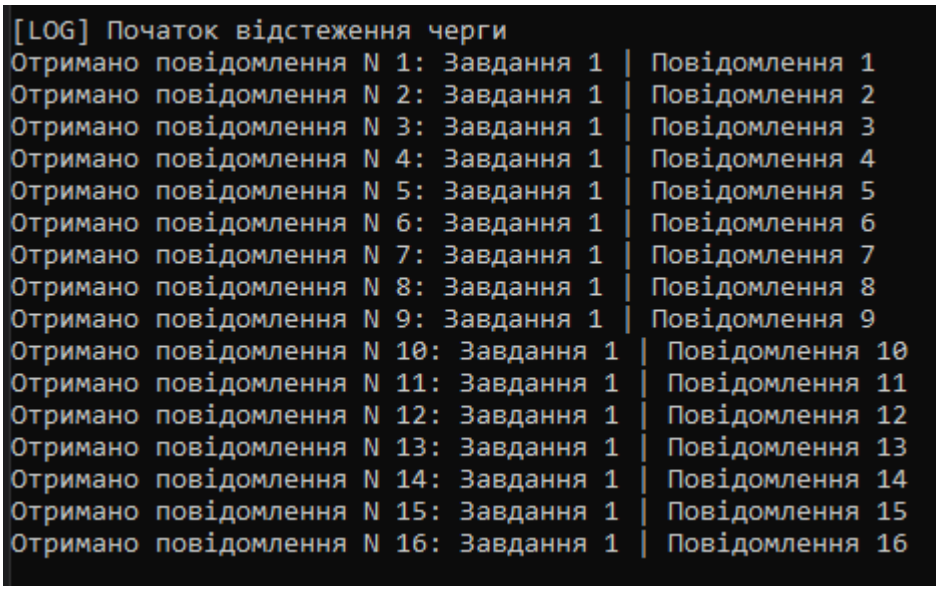

Рис. 2 – Завдання 1 Consumer

Спочатку було запущено consumer, програма повідомила, що почала відстежувати чергу. Далі запущено producer, програма розпочала генерування повідомлень і їх додавання до черги.

Завдання 2

Створіть прямий обмін із трьома прив’язаними до нього чергами.

Перша черга повинна отримувати повідомлення з ключем маршрутизації "production", друга черга повинна отримувати повідомлення з ключем маршрутизації "testing", а третя черга повинна отримувати повідомлення з ключем маршрутизації "development". Додайте нову прив’язку до обмінника, щоб усі повідомлення з ключем маршрутизації «debug» також надсилалися до черги «development».

Код для програми Producer’a:

```
def task_2_producer():
    counter = 1
    exchange_name = 'direct_routing'

    connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
    channel = connection.channel()

    # Створення прямого обміну
    channel.exchange_declare(exchange=exchange_name, exchange_type='direct')

    while True:
        time_to_sleep = 3
        time.sleep(time_to_sleep)

        # Генерування повідомлення з відповідним ключем маршрутизації
        message = f"   \t Завдання 2 | Повідомлення {counter}"
        routing_key = random.choice(['production', 'testing', 'development',
'debug'])
        body = message.encode('utf-8')

        # Надсилання повідомлення до прямого обміну з вказаним ключем маршрутизації
        channel.basic_publish(exchange=exchange_name, routing_key=routing_key,
body=body)

        print(f'[LOG] Надіслано повідомлення {counter} | {routing_key}')
        counter += 1


try:
    task_2_producer()
except KeyboardInterrupt:
    print('[LOG] Зупинено')
```

Код для програми Consumer’a:

```
import random
import time

import pika


def task_2_consumer():
    connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
    channel = connection.channel()

    exchange_name = 'direct_routing'

    channel.exchange_declare(exchange=exchange_name, exchange_type='direct')

    def production_callback(ch, method, properties, body):
        print(f"Отримано повідомлення з ключем маршрутизації 'production':
{body.decode('utf-8')}")

    def testing_callback(ch, method, properties, body):
        print(f"Отримано повідомлення з ключем маршрутизації 'testing':
{body.decode('utf-8')}")

    def development_callback(ch, method, properties, body):
        print(f"Отримано повідомлення з ключем маршрутизації 'development':
{body.decode('utf-8')}")

    # Черга для 'production'
    production_queue = 'production_queue'
    channel.queue_declare(queue=production_queue, durable=False)
    channel.queue_bind(exchange=exchange_name, queue=production_queue,
routing_key='production')
    channel.basic_consume(queue=production_queue,
on_message_callback=production_callback, auto_ack=True)

    # Черга для 'testing'
    testing_queue = 'testing_queue'
    channel.queue_declare(queue=testing_queue, durable=False)
    channel.queue_bind(exchange=exchange_name, queue=testing_queue,
routing_key='testing')
    channel.basic_consume(queue=testing_queue,
on_message_callback=testing_callback, auto_ack=True)

    # Черга для 'development'
    development_queue = 'development_queue'
    channel.queue_declare(queue=development_queue, durable=False)
    channel.queue_bind(exchange=exchange_name, queue=development_queue,
routing_key='development')
    channel.queue_bind(exchange=exchange_name, queue=development_queue,
routing_key='debug')
```

```
    channel.basic_consume(queue=development_queue,
on_message_callback=development_callback, auto_ack=True)

    print('\n[LOG] Початок відстеження черг')
    channel.start_consuming()


try:
    task_2_consumer()
except KeyboardInterrupt:
    print('[LOG] Зупинено')
```

```
[LOG] Надіслано повідомлення 1 | debug
[LOG] Надіслано повідомлення 2 | debug
[LOG] Надіслано повідомлення 3 | testing
[LOG] Надіслано повідомлення 4 | debug
[LOG] Надіслано повідомлення 5 | debug
[LOG] Надіслано повідомлення 6 | production
[LOG] Надіслано повідомлення 7 | production
[LOG] Надіслано повідомлення 8 | production
[LOG] Надіслано повідомлення 9 | development
[LOG] Надіслано повідомлення 10 | testing
[LOG] Надіслано повідомлення 11 | development
[LOG] Надіслано повідомлення 12 | development
[LOG] Надіслано повідомлення 13 | debug
[LOG] Надіслано повідомлення 14 | production
[LOG] Надіслано повідомлення 15 | debug
[LOG] Надіслано повідомлення 16 | development
[LOG] Надіслано повідомлення 17 | development
[LOG] Надіслано повідомлення 18 | testing
[LOG] Надіслано повідомлення 19 | testing
[LOG] Надіслано повідомлення 20 | debug
[LOG] Зупинено
```

Рис. 3 – Завдання 2 Producer

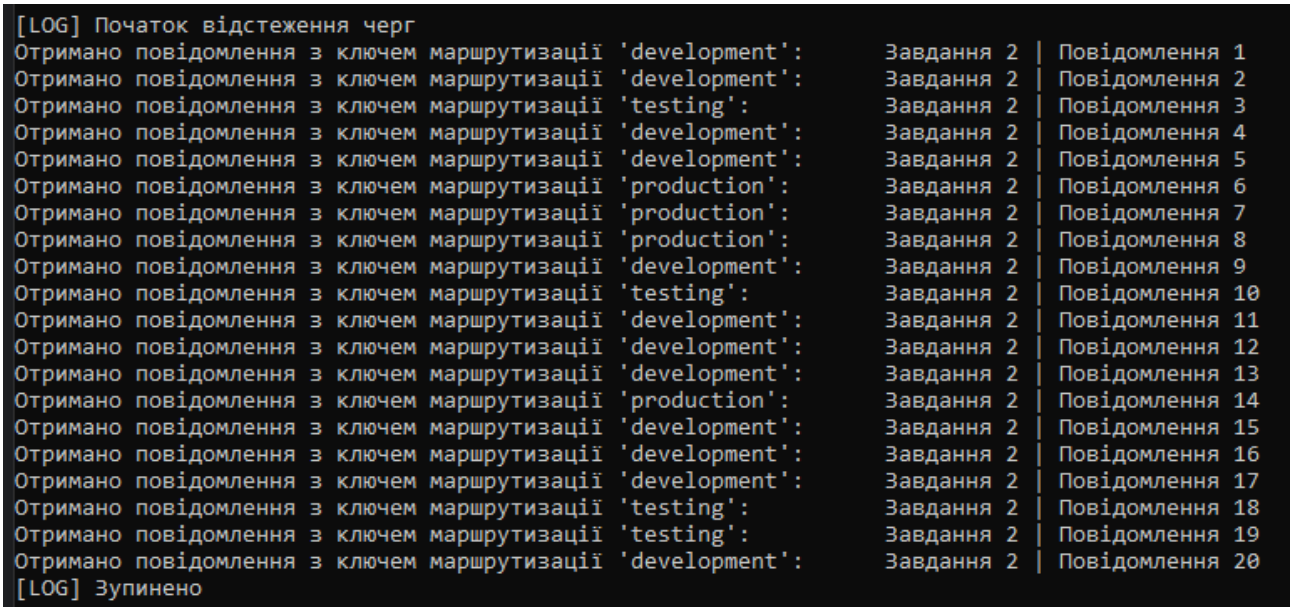

Рис. 4 – Завдання 2 Consumer

Спочатку був запущений споживач (consumer), і програма вивела повідомлення про те, що вона почала відстежування черги. Після цього був запущений виробник (producer), який розпочав генерувати повідомлення і додавати їх до черги. У виводі програми виробника також можна помітити, що після надсилання повідомлення в консоль виводиться інформація про це повідомлення у вигляді списку та його ключа маршрутизації.

Завдання 3

Створіть тематичний обмін із чергою для кожної категорії новин (наприклад, sports, politics, entertainment тощо). Прив’яжіть кожну чергу до обмінника за допомогою ключа маршрутизації "news.{category}".

Код для програми Producer’a:

```
def task_3_producer():
    counter = 1
    connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
    channel = connection.channel()

    channel.exchange_declare(exchange='news_exchange', exchange_type='topic')

    while True:
        time_to_sleep = random.randint(1, 3)
        time.sleep(time_to_sleep)

        category = random.choice(['sports', 'politics', 'entertainment'])  # список можливих категорій

        routing_key = f"news.{category}"
        message = f'Новина в категорії {category}'
        body = message.encode('utf-8')

        channel.basic_publish(exchange='news_exchange', routing_key=routing_key, body=body)

        print(f'[LOG] Надіслано повідомлення {counter}\t| {routing_key}   \t| {message}')
        counter += 1


try:
    task_3_producer()
except KeyboardInterrupt:
    print('[LOG] Зупинено')
```

Код для програми Consumer’а:

```
import random
import time

import pika


def task_3_consumer():
    connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
    channel = connection.channel()

    def callback(ch, method, properties, body):
        print(f'Отримано повідомлення: {body.decode("utf-8")}')

    channel.exchange_declare(exchange='news_exchange', exchange_type='topic')
    channel.queue_declare(queue='news')

    # Видалення всіх існуючих зв'язків для queue
    for rk in ['news.*']:
        channel.queue_unbind(queue='news', exchange='news_exchange',
routing_key=rk)

    channel.queue_bind(exchange='news_exchange', queue='news',
routing_key='news.*')

    channel.basic_consume(queue='news', on_message_callback=callback,
auto_ack=True)

    print('\n[LOG] Початок відстеження черги')
    channel.start_consuming()

try:
    task_3_consumer()
except KeyboardInterrupt:
    print('[LOG] Зупинено')
```

```
[LOG] Надіслано повідомлення 1  | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 2  | news.politics        | Новина в категорії politics
[LOG] Надіслано повідомлення 3  | news.politics        | Новина в категорії politics
[LOG] Надіслано повідомлення 4  | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 5  | news.entertainment   | Новина в категорії entertainment
[LOG] Надіслано повідомлення 6  | news.entertainment   | Новина в категорії entertainment
[LOG] Надіслано повідомлення 7  | news.entertainment   | Новина в категорії entertainment
[LOG] Надіслано повідомлення 8  | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 9  | news.entertainment   | Новина в категорії entertainment
[LOG] Надіслано повідомлення 10 | news.politics        | Новина в категорії politics
[LOG] Надіслано повідомлення 11 | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 12 | news.politics        | Новина в категорії politics
[LOG] Надіслано повідомлення 13 | news.entertainment   | Новина в категорії entertainment
[LOG] Надіслано повідомлення 14 | news.politics        | Новина в категорії politics
[LOG] Надіслано повідомлення 15 | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 16 | news.politics        | Новина в категорії politics
[LOG] Надіслано повідомлення 17 | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 18 | news.sports          | Новина в категорії sports
[LOG] Надіслано повідомлення 19 | news.entertainment   | Новина в категорії entertainment
[LOG] Надіслано повідомлення 20 | news.politics        | Новина в категорії politics
[LOG] Зупинено
```

Рис. 5 – Завдання 3 Producer

```
[LOG] Початок відстеження черги
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії politics
Отримано повідомлення: Новина в категорії politics
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії entertainment
Отримано повідомлення: Новина в категорії entertainment
Отримано повідомлення: Новина в категорії entertainment
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії entertainment
Отримано повідомлення: Новина в категорії politics
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії politics
Отримано повідомлення: Новина в категорії entertainment
Отримано повідомлення: Новина в категорії politics
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії politics
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії sports
Отримано повідомлення: Новина в категорії entertainment
Отримано повідомлення: Новина в категорії politics
[LOG] Зупинено
```

Рис. 6 – Завдання 3 Consumer

Висновки

Під час виконання лабораторної роботи я отримав знання про основні принципи функціонування брокера повідомлень RabbitMQ та його застосування для асинхронного обміну повідомленнями. Мною були освоєні ключові концепції, такі як Producer, Consumer, обмінники та черги.

В процесі виконання завдань я створював прив'язані черги та працював із ключами маршрутизації. Це дозволило мені здійснити практичні вправи, що стосуються реальних сценаріїв використання брокера повідомлень для організації асинхронного обміну повідомленнями у розподілених системах. Отримані в ході цієї роботи навички та досвід використання RabbitMQ виявляться корисними при подальшій роботі з розподіленими системами та асинхронною комунікацією.