2025 Winter Semester

OSBucks

Operating System Final Project

Won Hajin, Kim Jinseo, Lee Ilseob, Kim Hyeongyeong

OSBucks

The Problem We Aim to Solve

- **Overview of the Problem**
  - Many cafes still rely on manual drink preparation, which can result in bottlenecks, inaccurate orders, and poor inventory control during peak times.
- **Challenges Faced:**
  - Unpredictable order surges can overwhelm staff.
  - Without automated systems, baristas may struggle to balance speed and order accuracy.
  - Mismanagement of ingredient stock can lead to delays or unavailability of popular drinks.
- **Customer Impact**
  - Inefficient workflows contribute to longer queues and decreased satisfaction, especially during busy hours.

OSBucks

Limitations of Current Solutions

- **Key Limitations:**
  - Manual distribution of tasks often causes worker overload.
  - Difficulty in monitoring ingredient levels in real-time.
  - No systematic way of distributing orders to optimize worker efficiency.
  - Lack of real-time data on preparation times and profits.

OSBucks

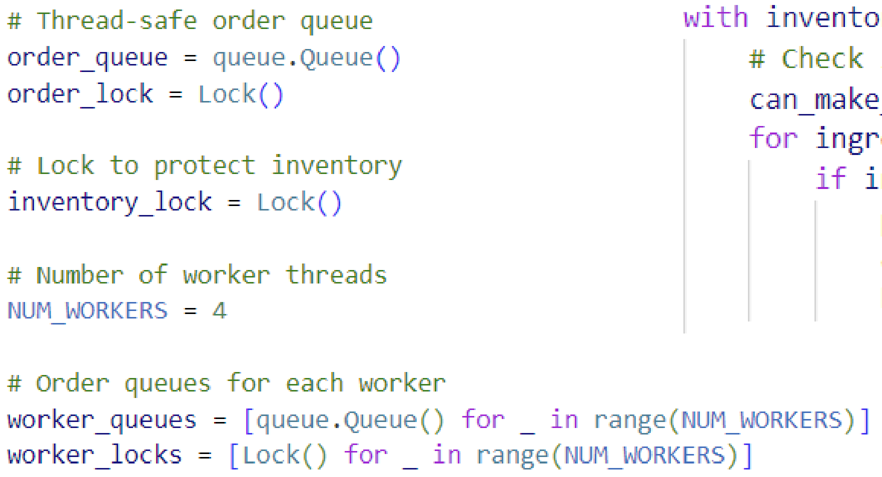
Our Approach: Automated Order Distribution System

- **Proposed Solution:**
  - Implement a Python-based server using SimpleXML RPCServer for real-time order communication.
  - Utilize multi-threading for concurrent processing of orders by worker threads.
  - Round-robin assignment to distribute tasks evenly among workers.
- **Key Features:**
  - Real-time monitoring of inventory
    - Supported by shared memory locks (inventory lock) to prevent race conditions when accessing stock data.
  - Dynamic worker assignment and parallel task handling: Each worker thread runs independently, simulating a CPU thread scheduler by using queues to handle multiple tasks concurrently.
  - Profit tracking integrated into the workflow: A centralized lock ensures accurate and atomic updates to shared variables, maintaining consistency.

OSBucks

Our Approach: Automated Order Distribution System

- **Key Metrics:**
  - Optimized worker load, ensuring no worker overload.
  - Improved ingredient monitoring, preventing sold-out issues.
  - The round-robin assignment evenly distributes tasks, balancing workloads across workers.
  - The inventory lock ensures accurate, concurrent access to stock data without conflicts.

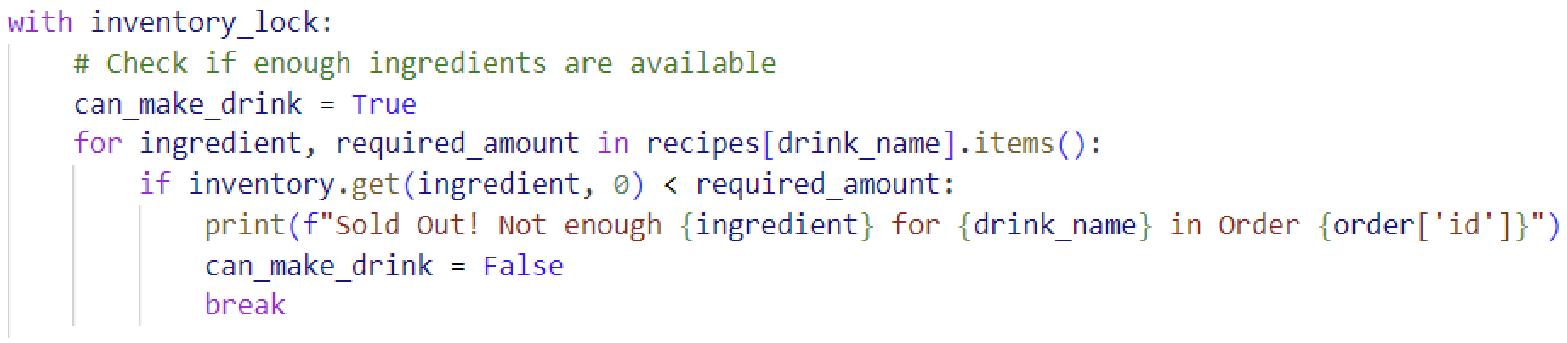


```python
# Thread-safe order queue
order_queue = queue.Queue()
order_lock = Lock()

# Lock to protect inventory
inventory_lock = Lock()

# Number of worker threads
NUM_WORKERS = 4

# Order queues for each worker
worker_queues = [queue.Queue() for _ in range(NUM_WORKERS)]
worker_locks = [Lock() for _ in range(NUM_WORKERS)]
```



```python
with inventory_lock:
    # Check if enough ingredients are available
    can_make_drink = True
    for ingredient, required_amount in recipes[drink_name].items():
        if inventory.get(ingredient, 0) < required_amount:
            print(f"Sold Out! Not enough {ingredient} for {drink_name} in Order {order['id']}")
            can_make_drink = False
            break
```

OSBucks

Solution Demo

https://youtu.be/dHzKQG4vRyY

THANK YOU

web	dfrc.korea.ac.kr
email	ruby100400@korea.ac.kr