Week1 revision Exercise: Coffee Shop Order System

Scenario

You've been hired to help a small coffee shop modernize how they manage customer orders. Right now, everything is done manually, but they would like a simple Java program to keep track of menu items, customer orders, and the queue of waiting customers.
Your job is to build a small console-based simulation of this system.

Learning Goals

This exercise will help you revise:

- Java basics: variables, if-statements, loops
- Collections: ArrayList, HashMap, Queue
- Object-oriented concepts: class, constructor, constructor chaining
- Static factory methods
- Singleton design pattern

Tasks

1. **Create a class called Order**

   Each order should have:
   - a unique order ID (e.g., Ord1, Ord2, …)
   - a drink name
   - a size (e.g., Small, Medium, Large – please use an **enum**)
   - a price

Add at least two constructors:
   - one that takes all parameters (*order id, drink name, size, price*)
   - one that assumes default values
     (*constructor chaining – default values of coffee, Small*)

Include a toString() method to describe an order.

   (The **toString** method returns a string that textually represents an object)

```
public String toString() {
        return " information about an object";
}
```

2. **Create a CoffeeShopManager class**
   This class will be a **Singleton** (only one instance).

   It should contain:
   - a HashMap<String, Double> for the menu (drink name → base price)
   - an ArrayList<Order> for all completed orders
   - a Queue<Order> for the orders waiting to be ade

Include methods to:

- add a new order to the queue
- process (remove) an order from the queue and add it to completed orders
- show the total number of orders completed
- show total revenue so far

3. **Create a static OrderFactory class**
   - Add a createOrder(String drinkName, String size) method.
   - It should calculate price based on the drink type and size (use simple logic or multipliers).
   - Return a new Order object.

4. **Create a Main class (e.g., CoffeeShopApp)**
   In main():
   - Get the singleton CoffeeShopManager instance.
   - Add several drinks to the menu (HashMap).
   - Use the factory to create several Order objects (simulate customers).
   - Add orders to the manager's queue.
   - Process a few orders.
   - Print out the current state (pending queue, completed orders, total revenue).

5. **Optional Extensions (if you finish early)**
   - Add a loyalty system (after every 3 orders, next coffee is free).
   - Add new menu items dynamically.
   - Allow different types of drinks (e.g., Tea, Latte, Espresso).
   - Use random generation to simulate incoming customers.

Expected Output Example (sample idea but really up to you how you present the information**)**

--- Coffee Shop Menu ---
Latte: £3.00
Espresso: £2.50
Mocha: £3.50

New order added: Ord1 - Latte (Medium)
New order added: Ord2 - Espresso (Large)

Processing order: Ord1
Processing order: Ord2

Total orders completed: 2
Total revenue: £6.50