Machine Coding

Dynamic menu

Coding Duration: 150 min

End time: 1:55 PM

**Context**

Dynamic menu cards are just like any other menu card that displays dishes offered by the restaurants. Restaurants’ owners are looking for a dynamic menu card instead of traditional static menu cards. Customers can choose any item from the menu card, whereas items can be prepared only if required ingredients for the selected dish are available in that restaurant.

Restaurants’ owners don't want to disappoint their customers by canceling an already placed order. Instead if it can’t be prepared due to unavailability of required ingredients, the item should not be displayed in the menu card itself.

**Requirements**

Dynamic menu cards should support the following functionalities.

* API to add ingredients with quantity.
* API to get available ingredients with respective quantities.
* API to add dishes with required ingredients. Dishes can be added without availability of the ingredients but should not display in the menu card until the ingredient is available.
* API to fetch menu cards with the available dishes.
* API to order dishes in one or more quantities. The order may contain 1 or more different types of dishes. If there are not enough ingredients to make all the dishes then the whole order should fail with an appropriate error message.

Eg 1: If X quantity of a dish d1 can be prepared with available ingredients, and a customer is placing an order for Y quantity (Y > X), the order should be failed with a proper message stating only X qty is available for the selected dish.

Eg 2: If the order contains 2 dishes (in quantity Y1, Y2) which require the same ingredients and the second dish cannot be fulfilled as it is short of ingredients, then the order should fail with a message stating the exact amount X of dish2 that can be fulfilled. Here X will be less than Y2.

If an ingredient is common between two dishes and only one of the dishes can be prepared with the available ingredients. Customers should be able to book any of the dishes. Though once the order is placed for any of the dishes both the dishes should be unavailable from the menu card. For example: Dishes D1 and D2 require 1 quantity of ingredients I1, the menu should display both the dish D1 and D2. As soon as one of the dishes is being ordered both the dishes should be removed from the menu. (Assuming there is only one quantity of ingredient I1 available in the restaurant. )

If only one quantity of a dish can be prepared with available ingredients. And two customers are trying to book this dish. Only one order should succeed whereas the other should fail with a proper error message. (There should be a unit test case to verify concurrency is being handled properly)

**Assumptions**:

* You can assume one dish required only 1 type of ingredients though the quantity of ingredient would differ as per the dish.
* You can also assume that there is only 1 restaurant.
* Bonus:
* Note: Do not attempt the bonus section if the base requirements haven’t been met.
* Add support for when the dishes require more than 1 type of ingredients.
* More than one restaurant and their specific menu and ingredients.
* Guidelines
* Please discuss the solution with an interviewer
* Please do not access internet for anything EXCEPT syntax
* You are free to use the language of your choice
* All code should be your own
* Please focus on the Bonus questions only after ensuring the required features are complete and demoable.

**Expectations**

* We are not expecting you to write REST API or create UI. It's just a functional APIs are expected here.
* Create the sample data yourself. You can put it into a file, test case or main driver program itself.
* Code should be demo-able. Either by using a main driver program or test cases.
* Code should be modular. Code should have basic OO design. Please do not jam the responsibilities of one class into another.
* Code should be extensible. Wherever applicable, use interfaces and contracts between different methods. It should be easy to add/remove functionality without re-writing the entire codebase.
* Code should handle edge cases properly and fail gracefully.
* Code should be legible, readable and DRY

**Sample test cases:**

Note: The sample below describes the text input as a list of commands. It is one out of the many ways we can interact with your program. You can also instead write unit tests to showcase the correctness.

Input.txt

Add ingredients 3

I1 2 // <Ingredient name, Ingredient quantity>

I2 2

I3 1

Add dishes 4

D1 I1 2 // <Dish name, Ingredient name, Ingredient quantity>

D2 I1 1

D3 I2 2

D4 I3 1

Get menu

Order D1 1 // <Dish name, Quantity>

Get menu

Order D3 2, D4 5

Get ingredients

Ouput

Ingredient(s) added

Dishes added

Available dishes

D1 // optionally show the ingredient requirements against dishes

D2

D3

D4

Order placed successfully

Available dishes

D3

D4

Couldn’t place your order, only 1 qty of D3 is available

IngredientName

I2 x 2

I3 x 1