Інтерфейси:

- IOffer
    - Властивості:
        - Ім’я
        - Опис
        - Категорія в меню (піца/десерт/напитки/гарячі страви/т.д.): Menu.Enum
        - Загальний час приготування: TimeOnly
- ICashRegister (Каса)//чи варто виділяти інтерфейс чи слід одразу класом
    - Властивості:
        - Замовлення
        - Перелік способів оплати
        - Кількість людей в черзі //уточнити чи потрібно
    - Методи:
        - Оплата (приймає ціну/замовлення і спосіб оплати)
        - Встати в чергу
        - Вийти з черги
        - Отримати контакт куди оплачувати
        - Взнати свою привілейованість
- IVisualization (придумати кращу назву)
    - Методи//це інтерфейс візуалізатора із визначиними загльними методами для взаємодії з симулятором
        - Відобразити
- IStaff
    - Делегати:
        - Повідомити наступного
        - Повідомити про готовність замовлення
        - Повідомити про те що я вільний
    - Властивості:
        - Інформація
- IMenu
    - Властивості:
        - Страви
        - Додатки
        - Акції

Абстрактні класи:

- MoneyType
    - Поля:
        - Баланс
        - ID
    - Властивості:
        - Баланс
        - ID
    - Методи:
        - Оплата
        - Додати кошти/встановити кошти
- Ingredient
    - Властивості:
        - Ім’я
- Recipe
    - Поля:

        - Етапи готування
    - Enum:
        - Етапи готовки

Класи:

- Chef: IStaff
    - Поля:
        - Сховище для інгредієнтів: class
        - Сховище для результату роботи: class
        - Перелік/черга страв/елементів_страв що має приготувати
        - Наступний кухар: Nullable<Chef>/замість кухара можна поставити інтерфейс для працівників кухні чи просто працівників
    - Методи:
        - Встановити категорію яку готує
        - Додати страву у чергу -> void
        - Готувати -> void
        - Положити приготоване у відповідне місце -> void
    - Події:
        - Сповіщення про завершення (повідомляємо наступному кухарю після того як поклали приготоване у сховище)
- ChefManager: IStaff
    - Поля:
        - Перелік кухарів у розпорядженні
        - Нерозподілені страви для приготування: Dictionary<Order, Dictionary<IOffer, uint>>
    - Методи:
        - Встановлення для кожного кухара виду діяльності
        - Встановлення зав’язків між кухарями (хто кого повідомляє про завершення своєї частини)
        - Отримання нового замовлення та розподілення між кухарями
    - **Події:**
        - **Появилось замовлення**
- Waiter: IStaff
    - Поля:
        - Склад з якого бере приготоване кухарями
        - Список невиконаних замовлень
    - Методи:
        - Утворення замовлення із приготованих блюд
        - **Передача замовлення (на касу чи деінде) / інформування про готове замовлення**
- Order
    - Поля:
        - ID
        - Перелік елементів меню і кількість: Dictionary<IOffer, uint>
    - Властивості:
        - Перелік меню
    - Методи:
        - Загальна ціна
- Menu: IMenu
    - Поля:

        - Перелік пропозицій з цінниками: Dictionary<IOffer, decimal>
        - Додатки до страв (на прикладі додатків до піци чи можливо інших страв): Dictionary< Enum, Dictionary<IOffer, decimal>> // тут IOffer можливо варто замінити
        - Акції
- Pizza : IOffer // за аналогією інші складні блюда (за їх наявності)
    - Enum:
        - Розміри
        - Тип основи
    - Поля:
        - Розмір: enum
        - Тип основи: enum
        - Етапи готовки / рецепт
- Juice : IOffer // за аналогією кава/чай/інші напитки
    - **Поля:**
        - **Об’єм: enum**
        - **Смак : enum**
- Customer //переглядати віп-чи ні можна при оплаті. Порівнюється ідентифікатор покупція і наявність такого у базі. Як аналог надати покупцю поле із списком привілейованих рівнів
    - Поля:
        - Ідентифікатор (код/ПІБ/т.д.)
        - Платіжна система // мабуть це не є потрібним
        - Поточне замовлення що формує покупець
        - Віп-статус
    - Методи:
        - Формування замовлення -> Order
        - Здійснення замовлення через касу/термінал //чи варто цей і попередній пункт розділяти
- Manager: IStaff
    - Поля:
        - Меню
- Admin
    - Поля:
        - Дані піцерії
    - Методи:
        - Видача інформації про працівника/менеджера
        - Додати Віп-клієнта
        - Вилучити Віп-клієнта
        - **Додати/вилучити касу**
        - Додати/вилучити працівника
- CustumerGenerator
    - Поля:
        - Кількість клієнтів
        - інтервали
    - Методи:
        - Генерація покупця/покупців із випадковою затримкою в заданих межах//через yield чи як там реалізується безперебійна робота методу
        - Термінова генерація покупця/покупців
        - Термінова генерація заданої кількості покупців
- Simulator
    - Поля:
        - Генератор покупців

        - Покупці
        - Піцерія
    - Методи:
        - Старт
        - Стоп
- Pizzeria:
    - Поля:
        - Дані піцерії
        - Адмін
    - Властивості:
        - Каси
- PizzeriaData
    - Поля:
        - Меннеджер
        - Персонал
        - Програма лояльноссті
        - Меню
        - Каси
        - Склад
    - Властивості:
        - Каси
    - Методи:
        - Видача інформації про працівника/менеджера
        - Додати Віп-клієнта
        - Вилучити Віп-клієнта
        - **Додати/вилучити касу**
        - Додати/вилучити працівника
    - Enum:
        - Рівні ВІП
        - Категорії страв
- LoyaltyProgram:
    - Поля:
        - Покупці та їх статуси
    - Властивості:
        - Покупці та їх статуси
    - Методи:
        - Додати покупця
        - Встановити статус покупця
- Storage
    - Поля:
        - Інгредієнти та їх кількість на складі
    - Методи:
        - Покласти
        - Взяти

- PizzaRecipe: Recipe
-

//Додаткові класи:

- Інформаційна дошка – на неї менеджер вішає меню, оголошує акції і з ней покупці можуть отримати інформацію про це
- Клас із енумом з назвами віпок
- Працівник для видачі замовлень. Чи його роботу виконуватиме пакувальник?
- Якщо надати кухарям білдери, то можна створити абстрактний клас білдер і від нього окремо під кожну страву новий білдер. Головний на кухні коли роздає кухарям ролі, може визначити хто яким білдером користуватиметься.