**Початок**

1. В методі Main() ініціалізується магазин та інтерфейс додатку. В конструкторі магазину можна проініціалізувати його стенди. А в конструкторах кожного стенда можна проініціалізувати їхні продукти та продавців.
2. Користувач запускає роботу програми (викликає метод Start() даного додатку). В ньому викликається метод магазину Open() та викликається метод GenerateBuyers, який в циклі генерує покупців і працює в новому потоці.

**Послідовність роботи продавця**

1. В методі Shop.Open() для кожного стенда викликається метод Stand.Open(). B Stand.Open() генерується подія StandOpen, яку обробляє кожний продавець в методі stand\_OnStandOpen.
2. B методі stand\_OnStandOpen в кожному продавцеві запускається потік, який виконує метод DoWork, в якому він в циклі обробляє чергу стенда, якому він належить.
3. При отриманні покупця з черги, для нього продавцем виконується метод ServeBuyer(). Після цього записується кількість проданих товарів, які зберігаються в стенді. Потім для покупця викликається метод DoWork(), який запускає потік, що продовжує роботу покупця.

**Послідовність роботи покупця**

1. Після запуску всіх стендів і продавців, починаються генеруватися покупці.
2. В конструкторі кожного покупця виконується його ініціалізація та викликається метод DoWork(), що запускає новий потік .
3. В цьому потоці покупець шукає стенд, де найменша черга (обирає серед тих стендів, які ще він не відвідав), обирає в ньому товар і стає в чергу поточного стенда. Після цього потік завершує роботу.

**Закінчення**

1. Користувач завершує роботу і викликає метод Stop().
2. В цьому методі для додатку виставляється змінна isRunning = false, відповідно потік з методом GenerateBuyers() завершує свою роботу. Викликається для магазину метод Close(). Тут додаток має почекати, поки магазин не просигналізує, що завершив роботу. Після цього сигналу він виводить статистику продаж (shop. ShowStatistic()).
3. В методі shop.Close() для кожного стенда викликається метод Close(), де генерується подія CloseStand
4. Цю подію обробляє кожний продавець. Після її отримання продавці дообслуговують чергу стенда і завершують роботу, генеруючи при цьому подію WorkComplete.
5. Цю подію обробляє стенд. Якщо він отримає ці події від усіх його продавців, то він надсилає власну подію WorkComplete, яку обробляє магазин.
6. Якщо магазин отримав цю подію від усіх стендів, тоді він сигналізує подію WorkComplete.

**Beginning**

1. The Main() method initializes a Shop instance and an application interface. The Shop constructor initializes its Stands. Each Stand initializes its Product and Sellers in constructor.
2. The User starts program work by Start() method. In this method application calls method Shop.Open() and then method GenerateBuyers() that generates new Buyers and works in a new thread.

**Seller’s flow**

1. In Shop.Open() method, the Stand.Open() method is called for each Stand. In this method, the StandOpen event raises. Each Seller of this stand in the OnStartOpen() method handles that.
2. In stand\_OnStandOpen() method each Seller creates a new thread, that does DoWork() method, in which it serves the Buyers queue in the loop.
3. When Seller gets some Buyer from queue, it serve Buyer (it calls method ServeBuyer()). After that, Seller writes information about number of sold products (this information is stored in Stand). Then the Seller calls a DoWork() method for served Buyer, that creates new thread and continues the Buyer’s work.

**Buyer’s flow**

1. After Shop opening, the GenerateBuyers() method starts to generate new Buyers.
2. The constructor of each Buyer does initialization of instance and calls a DoWork() method, that creates a new thread.
3. In this thread, the Buyer chooses a Stand where is a smallest queue (it chooses among those Stand that it has not visited yet), chooses the product (number of products) and take the queue of selected Stand. After that, the Buyer’s Thread ends work.

**Ending**

1. The User calls a Stop() method.
2. In this method, the application variable isRunning set to false; accordingly, the thread that do a GenerateBuyers() method will stop. Then the Stop() method calls a Close() method for Store. Here, the application should waiting until the Store does not signal that it has completed the work. After that, the application displays statistics of Shop (shop.ShowStatistic()).
3. In the shop.Close() method, each Stant calls the Close() method, where the CloseStand event is raised.
4. Each Seller handles this event. After that, the Sellers serves the Buyers queue to the end and completes the work. Than it generates the WorkComplete event.
5. The Stand handles this event. If it receives these events from all its Sellers, it will raises the WorkComplete event.
6. The Store handles Stand’s WorkComplete event. If the Store receives these events from all its Stands, it will raises the WorkComplete event.