Il y a deux incohérences dans le code :

-Les assiettes se lavent mais ne s'essuient pas. Les numéros d'assiettes et de piles s'accumulent. Le code ne se crash pas.

-Pourquoi les il n'y a pas d'incohérence dans le nombre d'assiettes.

Explication :

Il n'y a pas de crash car les assiettes se remplissent de manière instantané. Il n'y a pas d'accès simultané à la pile d'assiette pour laver et essuyer. Personne ne bloque la pile, il n'y a pas de concurrence . Lorsqu'on modifie la fonction push, la ressource est bloquée, quelqu'un d'autre remplit la pile. Il y a concurrence pour laver et essuyer les assiettes. Cela produit un crash.

D'autre part, la pile se remplit à l'infini car il n'y a pas de vérification de la fonction isFull()

Solution :

La classe Assiette encapsule une pile avec différentes fonctions, tel que push et pop qui remplisse et enlève des assiettes de la pile, respectivement

On voit que normalement, lorsque la pile est vide, aucune assiette n'est essuyée. Il doit attendre que la pile se remplisse (comme pour le laveur). Cependant, l'essuyeur est bloqué au démarrage.
En effet, il n'est pas notifié que la pile d'assiette est finie par les threads, qui vérifient les conditions. D'autre part, il manque certaines conditions à la fonction, notamment énoncer que la pile est pleine.

```
    synchronized public void push(Assiette assiette) throws
InterruptedException {
        while (isFull()) {
            wait();
        }
        assert !isFull();
        myList.add(assiette);
        System.out.printf("la pile contient %d assiettes%n",
        myList.size());
        notifyAll();
    }
```