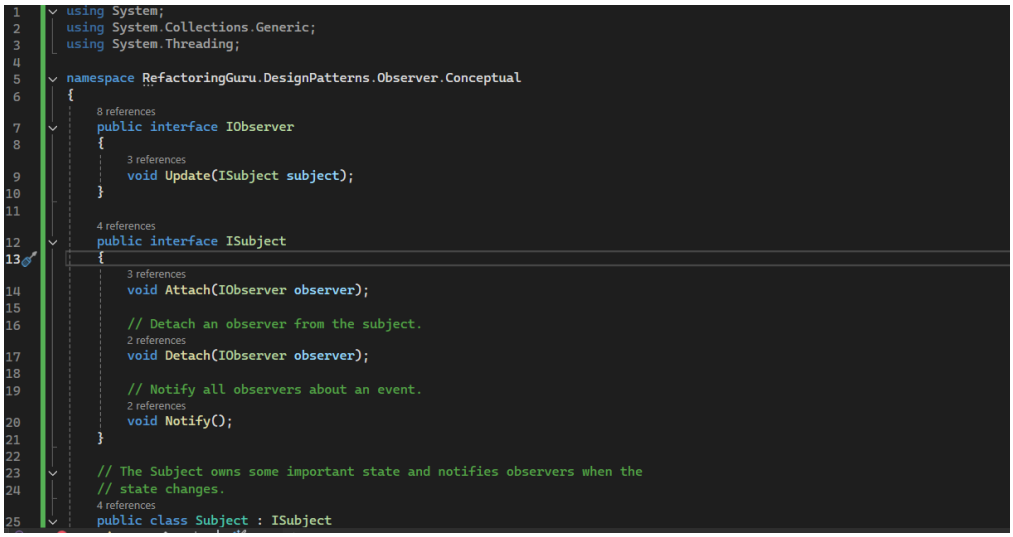
2311104068_ILHAMLIIASSIDAQ_TP13

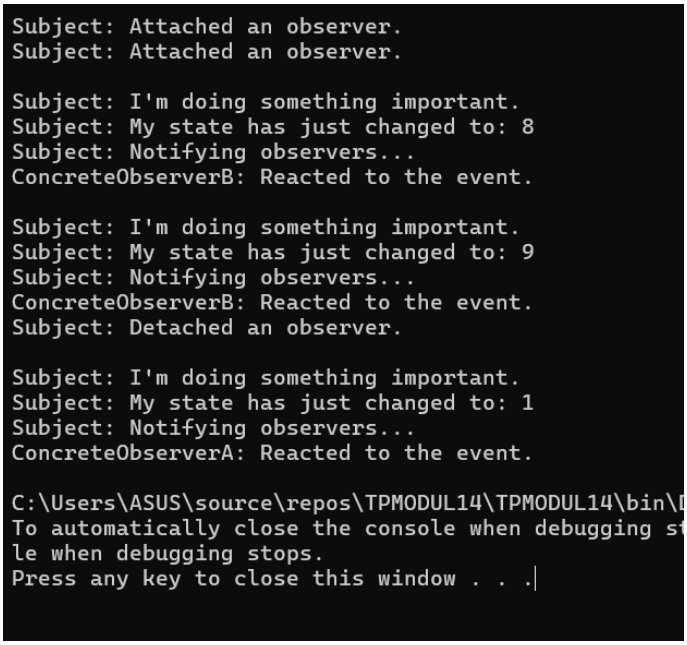
1. Link GitHub : https://github.com/ilhamlii21/KPL_ILHAM_LII_ASSIDAQ-2311104068_S1SE-07-02

2.



```
using System;
using System.Collections.Generic;
using System.Threading;

namespace RefactoringGuru.DesignPatterns.Observer.Conceptual
{
    8 references
    public interface IObserver
    {
        3 references
        void Update(ISubject subject);
    }

    4 references
    public interface ISubject
    {
        3 references
        void Attach(IObserver observer);

        // Detach an observer from the subject.
        2 references
        void Detach(IObserver observer);

        // Notify all observers about an event.
        2 references
        void Notify();
    }

    // The Subject owns some important state and notifies observers when the
    // state changes.
    4 references
    public class Subject : ISubject
```

Class program ini merupakan implementasi dari **Observer Design Pattern** dalam C#, di mana Subject bertindak sebagai objek yang diamati (publisher) dan ConcreteObserverA serta ConcreteObserverB adalah objek yang mengamati (subscribers atau observers). Ketika Subject melakukan perubahan pada state-nya melalui metode SomeBusinessLogic, ia akan memanggil Notify() untuk memberitahu semua observer yang telah terdaftar. Masing-masing observer akan merespons perubahan berdasarkan logika kondisi tertentu. Pola ini berguna untuk menciptakan hubungan satu-ke-banyak antara objek sehingga ketika satu objek berubah, semua yang bergantung padanya dapat secara otomatis diperbarui.



Hasil output ini menunjukkan cara kerja Observer Pattern dalam program. Pertama, dua observer (ConcreteObserverA dan ConcreteObserverB) ditambahkan ke dalam subject. Setiap kali method SomeBusinessLogic dijalankan, subject mengubah nilai State secara acak dan memberi notifikasi ke semua observer yang terdaftar. Pada dua pemanggilan awal, nilai State berubah menjadi angka yang memenuhi syarat reaksi ConcreteObserverB, sehingga hanya observer tersebut yang merespons. Setelah ConcreteObserverB dilepas (detached), hanya ConcreteObserverA yang tersisa, dan ketika nilai State berubah menjadi 1, observer tersebut merespons sesuai dengan logika yang telah ditentukan. Output ini menunjukkan bahwa setiap observer hanya bereaksi saat kondisi yang ditentukan terpenuhi.