NAMA: AMMAR DZAKI NANDANA
KELAS: SE 07 02
NIM: 2311104071

TP MODUL 13 KPL

A. Contoh Kondisi Penggunaan Design Pattern "Observer"

Contohnya adalah notifikasi status build produk.
Dalam konteks kode Builder, kita ingin memberikan notifikasi otomatis (misalnya ke UI atau log sistem) setiap kali bagian produk selesai dibangun (PartA, PartB, PartC). Maka dapat digunakan Observer Pattern agar setiap observer (UI, logger, monitor) mendapatkan update tanpa mengubah ConcreteBuilder.

B. Langkah-langkah Implementasi Observer Pattern

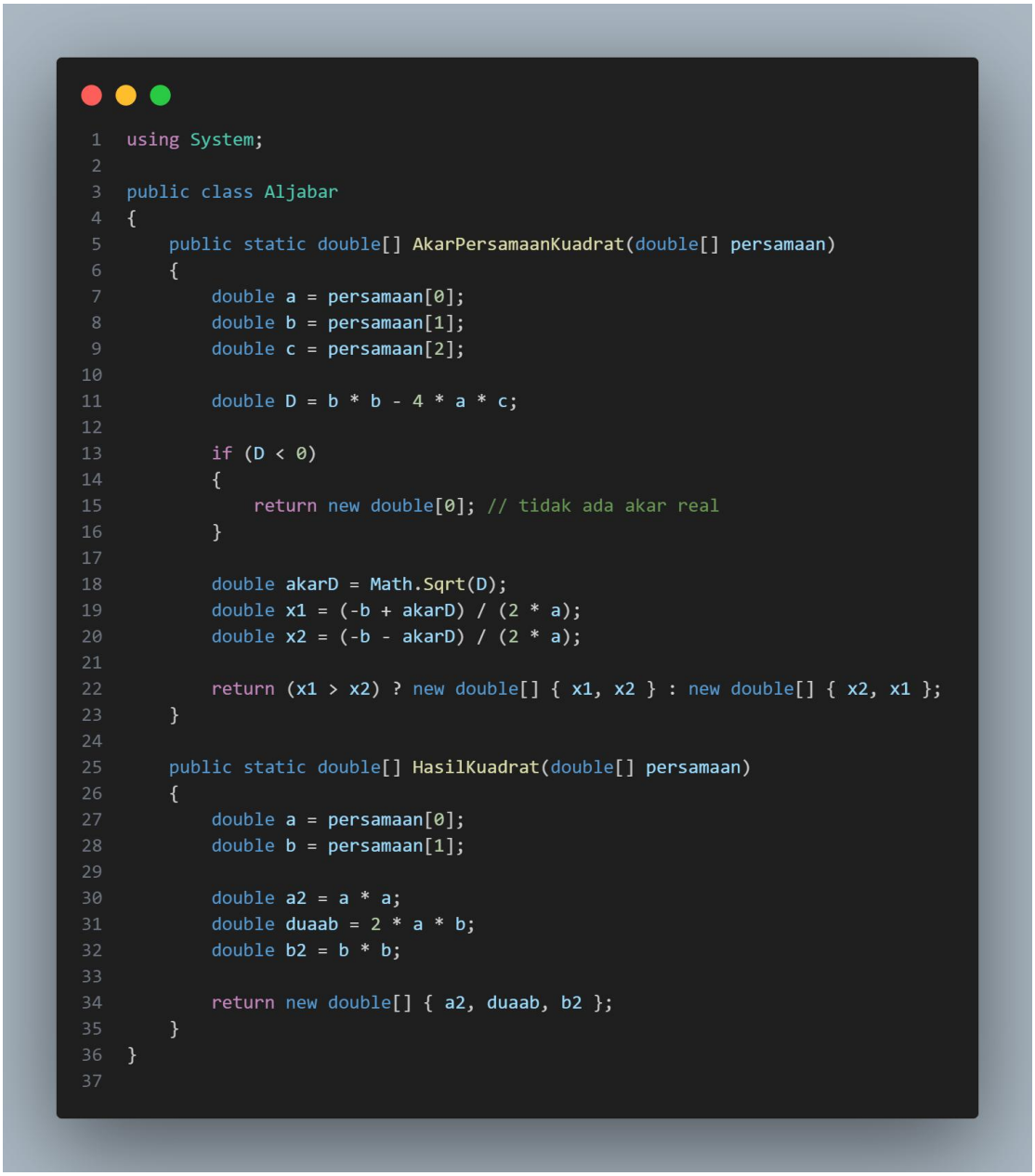
Interface IObserver

```
public interface IObserver
{
    void Update(string message);
}
```

Interface ISubject

```
public interface ISubject
{
    void Attach(IObserver observer);
    void Detach(IObserver observer);
    void Notify(string message);
}
```

ConcreteBuilder sebagai Subject



```
using System;

public class Aljabar
{
    public static double[] AkarPersamaanKuadrat(double[] persamaan)
    {
        double a = persamaan[0];
        double b = persamaan[1];
        double c = persamaan[2];

        double D = b * b - 4 * a * c;

        if (D < 0)
        {
            return new double[0]; // tidak ada akar real
        }

        double akarD = Math.Sqrt(D);
        double x1 = (-b + akarD) / (2 * a);
        double x2 = (-b - akarD) / (2 * a);

        return (x1 > x2) ? new double[] { x1, x2 } : new double[] { x2, x1 };
    }

    public static double[] HasilKuadrat(double[] persamaan)
    {
        double a = persamaan[0];
        double b = persamaan[1];

        double a2 = a * a;
        double duaab = 2 * a * b;
        double b2 = b * b;

        return new double[] { a2, duaab, b2 };
    }
}
```

C. Kelebihan dan Kekurangan Observer Pattern

✅ Kelebihan:

Low coupling: Subject dan Observer tidak bergantung langsung satu sama lain.

Mudah dikembangkan: Menambah observer baru tidak perlu ubah kode subject.

Real-time update: Cocok untuk sistem event-driven.

❌ Kekurangan:

Sulit dilacak: Jika observer banyak, alur update bisa membingungkan.

Potensi performa buruk: Banyak observer = Notify() bisa lambat.

Memory leak: Jika lupa Detach(), observer tetap tersimpan di memori.

Penjelasan Builder Design Pattern di C#

1. Interface IBuilder

Menyediakan metode: BuildPartA(), BuildPartB(), BuildPartC().

2. Class ConcreteBuilder

Implementasi nyata dari IBuilder

Memiliki _product (instance dari Product)

Setiap BuildPartX() menambahkan bagian ke _product

GetProduct() mengembalikan produk akhir

Reset() menginisialisasi ulang produk

3. Class Product

Menyimpan daftar bagian List<string> _parts

ListParts() menampilkan semua bagian

4. Class Director

Mengatur urutan pembangunan produk

Bisa membangun:

Produk minimum (BuildMinimalViableProduct())

Produk penuh (BuildFullFeaturedProduct())

5. Class Program (Main)

Membuat Director dan ConcreteBuilder

Menampilkan hasil dari:

Produk standar

Produk lengkap

Produk kustom