TUGAS PENDAHULUAN
KONSTRUKSI PERANGKAT LUNAK

MODUL XIII
DESIGN PATTERN IMPLEMENTATION

Disusun Oleh:

Aorinka Anendya Chazanah / 2211104013

S1 SE-06-01

Dosen Pengampu:

Yudha Islami Sulistya, S.Kom., M.Cs

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

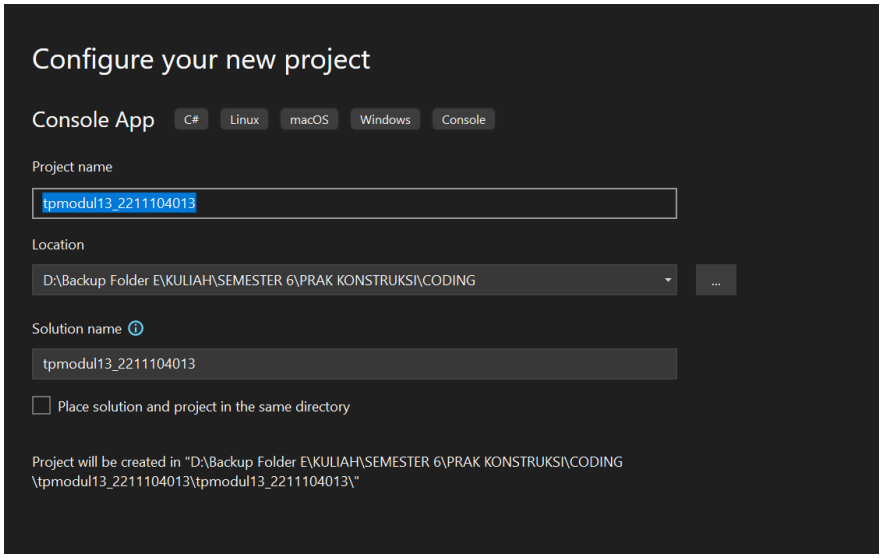
TUGAS PENDAHULUAN

1. MEMBUAT PROJECT GUI

Buka IDE misalnya dengan Visual Studio

A. Misalnya menggunakan Visual Studio, buatlah project baru dengan nama tpmodul113_NIM

B. Project yang dibuat bisa berupa console atau sejenisnya



2. MENJELASKAN SALAH SATU DESIGN PATTERN

Buka halaman web https://refactoring.guru/design-patterns/catalog kemudian baca design pattern dengan nama “Observer”, dan jawab pertanyaan berikut ini (dalam Bahasa Indonesia):

A. Berikan salah satu contoh kondisi dimana design pattern “Observer” dapat digunakan

Jawaban:

Salah satu contoh kondisi penggunaan Observer adalah pada sistem notifikasi aplikasi e-commerce. Misalnya, pengguna ingin mendapatkan pemberitahuan saat harga suatu produk turun. Dalam kasus ini:

1. Produk bertindak sebagai *publisher*, karena status harganya dapat berubah.
2. Pengguna yang ingin diberi tahu saat harga turun bertindak sebagai *subscriber*.

Dengan Observer pattern, pengguna bisa *subscribe* ke produk tertentu. Ketika harga produk berubah, sistem akan otomatis memberi tahu semua pengguna yang telah mendaftar (*subscribe*) ke produk tersebut. Ini memungkinkan pemberitahuan yang efisien dan hanya dikirim ke pihak yang relevan.

B. Berikan penjelasan singkat mengenai langkah-langkah dalam mengimplementasikan design pattern “Observer”

a. **Pisahkan logika bisnis**

Tentukan bagian program yang bertindak sebagai *publisher* (sumber perubahan) dan yang sebagai *subscriber* (yang akan merespons perubahan).

b. **Buat interface subscriber**

Interface ini minimal memiliki satu metode, seperti `update()`, yang akan dipanggil oleh *publisher* saat terjadi perubahan.

c. **Buat interface publisher**

Interface ini berisi metode untuk menambahkan, menghapus, dan memberi tahu *subscriber*, seperti `subscribe()`, `unsubscribe()`, dan `notify()`.

d. **Implementasikan daftar subscriber**

Biasanya dibuat dalam bentuk list atau map, bisa ditaruh di kelas abstrak atau helper class (komposisi), tergantung struktur program.

e. **Buat publisher konkret**

Saat terjadi perubahan status, publisher akan memanggil `notify()` untuk memberi tahu semua subscriber.

f. **Buat subscriber konkret**

Implementasikan metode `update()` untuk menentukan bagaimana objek merespons notifikasi dari publisher.

g. **Konfigurasikan di Client**

Daftarkan subscriber ke publisher yang sesuai saat inisialisasi atau saat runtime, sesuai kebutuhan aplikasi.

C. Berikan kelebihan dan kekurangan dari design pattern “Observer”

Jawaban:

Kelebihan Observer Pattern

1. **Mendukung Prinsip Open/Closed**: Kamu bisa menambahkan subscriber (pengamat) baru tanpa mengubah kode publisher (subjek), sehingga kode lebih mudah dikembangkan dan dipelihara.
2. **Relasi yang Longgar (Loose Coupling):** Publisher tidak tahu detail tentang subscriber. Mereka hanya terhubung melalui antarmuka, membuat sistem lebih fleksibel dan modular.
3. **Notifikasi Otomatis:** Subscriber akan otomatis diberi tahu jika ada perubahan pada publisher, tanpa perlu melakukan polling secara manual.
4. **Fleksibel di Waktu Runtime**: Subscriber bisa bergabung atau keluar

dari daftar kapan saja saat program berjalan.

Kekurangan Observer Pattern

1. **Urutan Notifikasi Tidak Terjamin:** Subscriber menerima notifikasi dalam urutan acak, sehingga sulit jika urutan eksekusi penting.
2. **Risiko Memori Bocor:** Jika subscriber tidak dihapus dengan benar dari daftar, bisa terjadi memory leak atau notifikasi yang tidak diinginkan.
3. **Sulit untuk Menelusuri Alur Program**: Dengan banyak subscriber, alur notifikasi bisa menjadi kompleks dan sulit ditelusuri saat debugging.
4. **Potensi Overhead:** Jika jumlah subscriber besar, setiap notifikasi bisa menyebabkan overhead performa karena semua subscriber harus dipanggil.

3. MEMANGGIL LIBRARY DI FUNGSI UTAMA

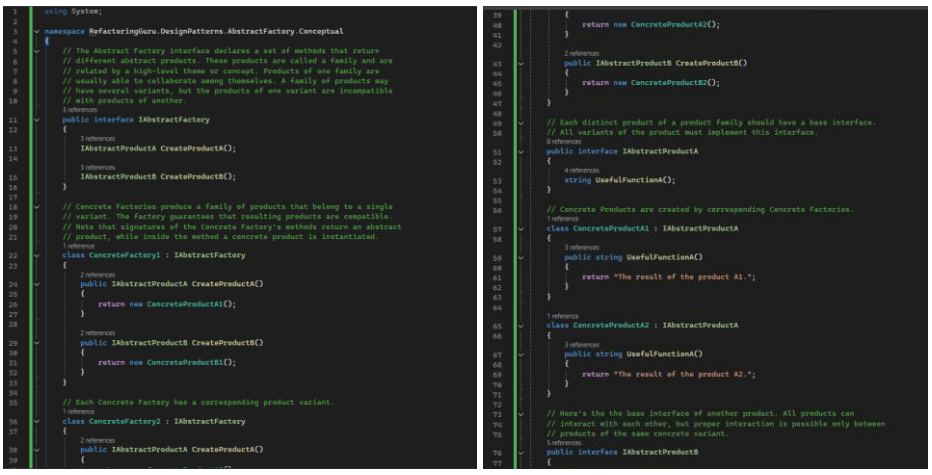
Buka halaman web berikut https://refactoring.guru/design-patterns/observer dan scroll ke bagian “Code Examples”, pilih kode yang akan dilihat misalnya C# dan ikuti langkah-langkah berikut:

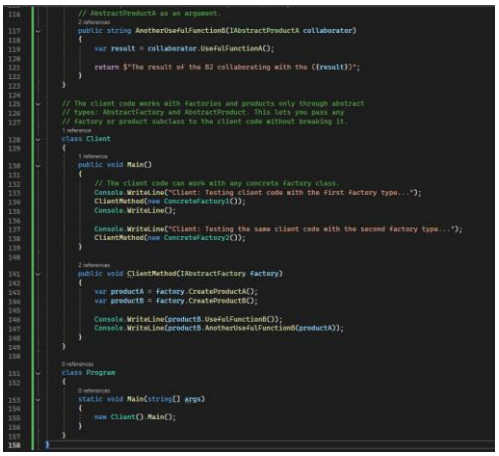
A. Pada project yang telah dibuat sebelumnya, tambahkan kode yang mirip atau sama dengan contoh kode yang diberikan di halaman web tersebut

B. Jawaban:

Pada contoh program kali ini memilih Abstract Factory

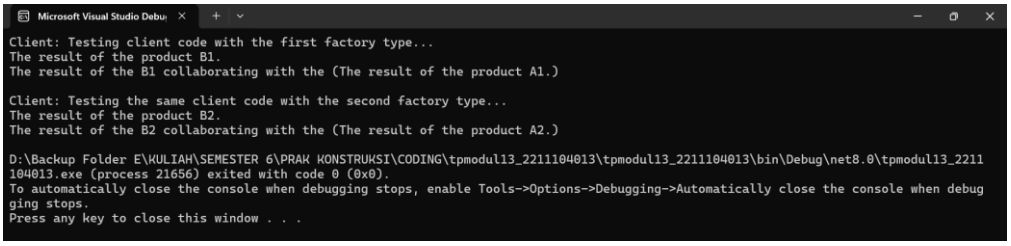
Kode program





C. Jalankan program tersebut dan pastikan tidak ada error pada saat project dijalankan

Output



D. Jelaskan tiap baris kode yang terdapat di bagian method utama atau “main”

1. **class Program**

   Mendeklarasikan sebuah kelas bernama Program yang merupakan entri utama dari aplikasi C#.

2. **static void Main(string[] args)**

   Ini adalah **method utama (entry point)** dari program. Ketika program dijalankan, eksekusi dimulai dari sini. string[] args adalah parameter opsional yang bisa menerima argumen dari command line.

3. **new Client().Main();**

   - Membuat objek baru dari class Client dengan new Client().
   - Langsung memanggil method Main() milik objek tersebut, yang berisi logika client untuk menguji dua concrete factory: ConcreteFactory1 dan ConcreteFactory2.
   - Dalam method Client.Main(), dua factory tersebut dipakai untuk menciptakan produk dan menjalankan metode kolaborasi antar produk, serta mencetak hasilnya ke konsol.