A 1. Static variabel adalah variabel yang nilainya tetap ada selama program berjalan dan bisa digunakan bersama oleh semua objek dalam satu kelas.

~ Fungsi Static Variabel

- Berbagi data antar objek → Semua objek bisa mengakses dan mengubah nilai yang sama.
- Menghemat memori → Hanya dibuat sekali, tidak perlu disimpan di setiap objek.
- Menjaga nilai tetap ada → Tidak hilang meskipun objek dihapus.

~ Kapan Menggunakan Static Variabel?

- Jika suatu nilai harus sama untuk semua objek (misalnya jumlah total objek yang dibuat).
- Jika ingin menyimpan data yang bisa diakses tanpa membuat objek (misalnya konstanta atau konfigurasi).

2. Main method harus dituliskan dengan static karena program dapat langsung menjalankannya tanpa perlu membuat objek terlebih dahulu. Jika main method tidak static, maka Java harus membuat objek dari kelas tersebut sebelum bisa menjalankannya, padahal saat program dimulai, objek belum ada. Dengan menjadikannya static, JVM bisa langsung mengeksekusi main method sebagai awal program tanpa harus membuat instance kelas. Selain itu, penggunaan static pada main method juga membantu menghemat memori karena tidak perlu membuat objek hanya untuk menjalankan program.

4. Jika dalam method hitungPenjumlahan() ditambahkan pemanggilan hitungPerkalian (a, b) , maka setiap kali hitungPenjumlahan() dijalankan, method hitungPerkalian() juga akan ikut dieksekusi. Karena hitungPerkalian() adalah method static, maka bisa langsung dipanggil tanpa perlu membuat objek. Akibatnya, selain menampilkan hasil penjumlahan, program juga akan menampilkan hasil perkalian, meskipun perkalian tidak diminta secara langsung.

5. Jika dalam method hitungPerkalian() ditambahkan hitungPenjumlahan(a, b), maka akan error. Ini karena hitungPerkalian() adalah method static*, sedangkan *hitungPenjumlahan()* adalah *non-static*. Method non-static tidak bisa langsung dipanggil dalam method static tanpa membuat objek dari kelasnya dulu.

B 2. Jika menghapus separator “/” pada baris 4-6 di file Vehicle.java dan pada baris 6 di file TestVehicle.java, maka kode yang sebelumnya dari komentar akan menjadi aktif.

~ Di Vehicle.java, ada konstruktor yang sebelumnya di-comment dengan tanda `//`. Jika tanda komentar dihapus, maka konstruktor tersebut akan menjadi kode yang aktif, tetapi karena konstruktor itu private, maka tidak bisa menggunakannya di luar kelas tersebut, yang akan menyebabkan error.

~ Di TestVehicle.java, ada baris kode yang mencoba membuat objek `Vehicle1` menggunakan konstruktor yang tidak ada. Jika komentar dihapus, program akan mencoba menjalankan kode ini, tetapi karena Vehicle1 tidak ada, ini akan menyebabkan error.

3. Jika mengubah variabel `load` menjadi konstanta dengan menambahkan kata kunci `final`, maka nilai dari variabel tersebut tidak bisa diubah setelah pertama kali diinisialisasi. Biasanya, variabel `load` akan bertambah setiap kali kita menambahkan beban menggunakan metode `addBox()`. Namun, jika `load` diubah menjadi `final`, maka akan error ketika mencoba mengubah nilainya, karena konstanta hanya bisa diberikan nilai satu kali dan tidak bisa diubah lagi setelah itu. Jadi, program akan gagal dijalankan karena mencoba mengubah nilai variabel yang sudah ditetapkan sebagai konstanta.

4. Jika menambahkan keyword `static` pada variabel `maxLoad`, maka `maxLoad` menjadi milik kelas, bukan objek. Artinya, semua objek Vehicle akan berbagi nilai `maxLoad` yang sama. Nilai `maxLoad` hanya bisa diubah sekali untuk semua objek, dan bisa diakses tanpa harus membuat objek, hanya dengan `Vehicle.maxLoad`.