**Laporan Praktikum Modul 3**

**Dasar-Dasar Pemrograman Dart (OOP)**

**Nama : Muhammad Anantha Mahardika Ridwan**

**Kelas : D4SIKC-48-01**

**NIM : 707012400122**



**1. Perangkat Keras dan Lunak**

**a. Perangkat Keras (Hardware)**

* Laptop/PC minimal Intel Core i3
* RAM minimal 4 GB (disarankan 8 GB atau lebih)
* Penyimpanan minimal 10 GB
* Layar resolusi minimal 1366 × 768 piksel
* Koneksi internet

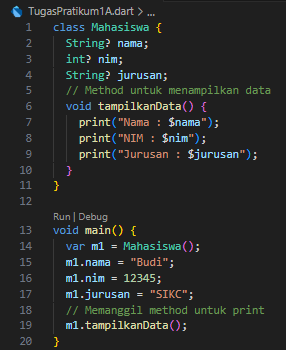
**b. Perangkat Lunak (Software)**

* Sistem Operasi: Windows 10/11, macOS, atau Linux
* Flutter SDK (termasuk Dart SDK)
* Visual Studio Code dengan ekstensi Flutter dan Dart
* Android Studio
* DartPad

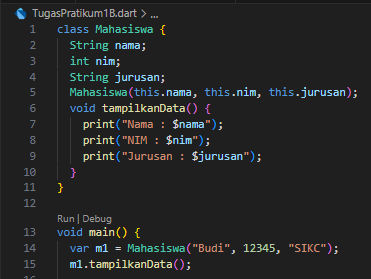
**2. Langkah – Langkah Praktikum**

* 1. **Contructor pada Class Mahasiswa**

1A. Kode yang gapake Contsructor



1B. Kode yang pakai constructor

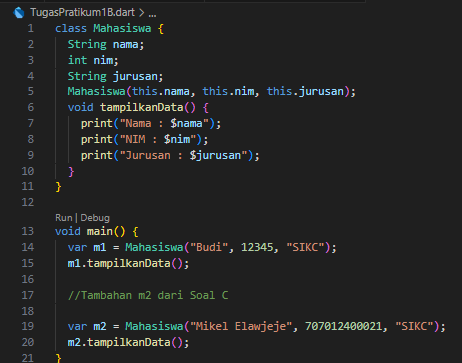


Pertanyaan :

* 1. Sudah saya jalankan keduanya dan Output dari kedua kode A dan B sama)
  2. Kelebihannya :
     + Membuat kode lebih enak diliat, struktur gitu
     + Datanya lebih aman karna wajib diisi saat objek dibuat, mengurangi kemungkinan nilai null
     + Proses inisisasi lebih Mudah dan lebih jelas

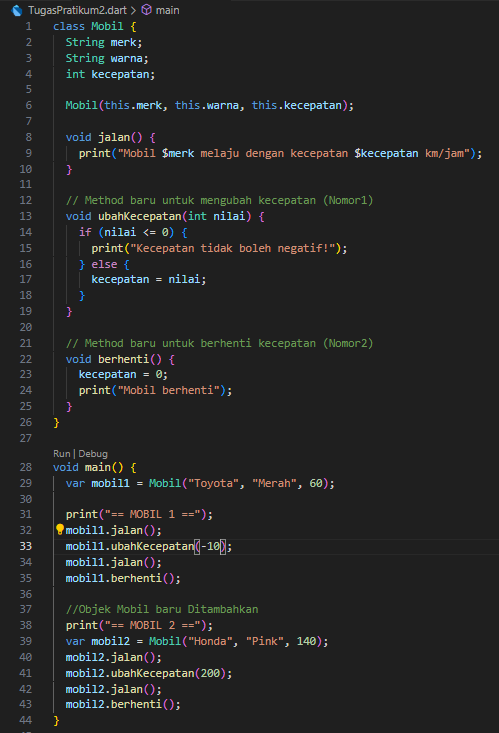
Kekurangan :

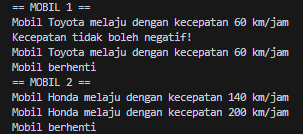
* + - Kalau jumlah atributnya banyak, parameter constructor bisa jadi Panjang
    - Tidak fleksibel kalua membuat objeknya tanpa dimulai dari awal
  1. Menambahkan Objek Kedua



Telah ditambahkan objek kedua bernama m2, dengan data mahasiswa yang berbeda yaitu Mahasiswa(”Mikel Elawjeje”, 707012400021, ”SIKC”) .

* 1. Yang Lebih efisien itu kode yang B karena pakai constructor Karena Datanya langsung diinisialisasi saat objke dibuat, Berfungsi untuk mencegah null Kode singkat dan efisien, jika dalam Sekala besar itu jadi aman digunakan Mengurangi risiko kelalaian orang saat mengisi data di objeknya
  2. **Implementasi Method pada class Mobil**





Pertanyaan :

1. Method ubahKecepatan(int nilai)

Method ini sudah ditambahkan yang ada di baris 13 sampai 19 Method ini dipakai untuk mengubah kecepatan mobil, Hanya menerima nilai ≥ 0, Kalau negatif munculnya pesan error seperti Output yang ada di mobil 1 pada gambar diatas

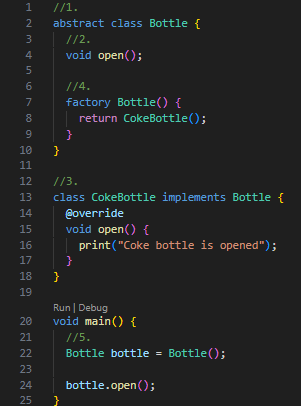
1. Method Berhenti()

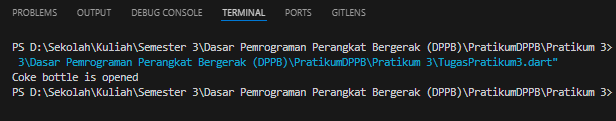
Method ini sudah ditambahkan yang ada di baris 22 sampai 26, Method ini dipakai untuk mengatur kecepatan mobil menjadi 0, dan menampilajn pesan ”Mobil Berhenti” seperti yang ditunjukkan di output pada gambar

1. Buat 2 Objek Mobiil

Mobil 2 Sudah ditambahkan yang ada pada baris 38 sampai 43 hal ini untuk menguji nilai kecepatan, terlihat output dan error jika negatif

* 1. **Abstract Class dan Factory Constructor**

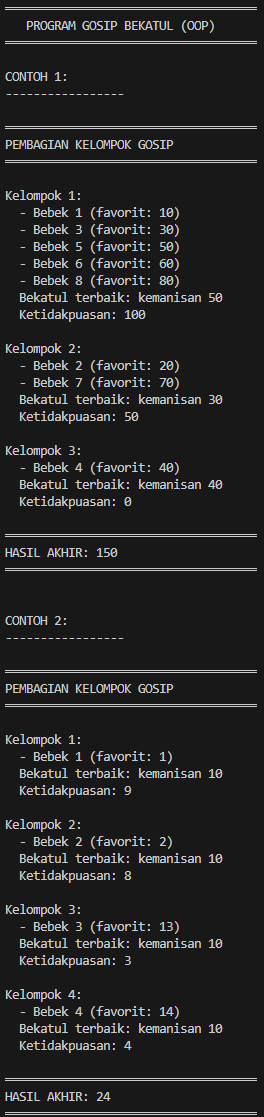




Pertanyaan :

1. abstract class Bottle digunakan sebagai antarmuka (interface) agar tidak bisa dibuat objek langsung.
2. Sudah ditambahkan seperti yang terlihat pada baris 4
3. Sudah ditambahkan seperti yang terlihat pada baris 13 sampai 19
4. Sudah ditambahkan seperti yang terlihat pada baris 7 sampai 10
5. Sudah ditambahkan seperti yang terlihat pada baris 22 sampai 25
   1. **Studi Kasus Gosip Bekatul**





Tujuannya membagikan bekatul dengan tingkat kemanisan terbaik, sehingga semua bebek yang saling bergosip mendapat jenis bekatul yang sama.

Analisis & Solusi :

1. **Pengelompokan Bebek:**
   * Class **Bebek** menyimpan data tiap bebek, termasuk nomor dan kemanisan favorit.
   * Class **KelompokGosip** mengelola kelompok bebek yang saling bergosip, membentuk jaringan.
   * Semua bebek yang terhubung, baik langsung maupun tidak, dianggap satu kelompok.
2. **Optimasi Bekatul:**
   * Setiap kelompok memilih bekatul yang menghasilkan total ketidakpuasan paling kecil.
   * Method di **KelompokGosip** menghitung selisih kemanisan tiap bebek dengan setiap bekatul yang dicoba, lalu menentukan yang terbaik.
3. **Hasil Akhir:**
   * Total ketidakpuasan terkecil dihitung dari semua kelompok setelah masing-masing memilih bekatul terbaik.

Penggunaan Class, Object, dan Method membuat masalah kompleks ini lebih terstruktur, memudahkan pemodelan entitas nyata (bebek) dan perilakunya, serta mempermudah penghitungan bekatul terbaik untuk setiap kelompok.

**3. Analisis dan Pembahasan**

1. **Tugas 1 - Constructor dan Inisialisasi Objek**
   * Constructor mempermudah proses pemberian nilai awal pada properti objek secara otomatis.
   * Menjaga data objek tetap konsisten sejak dibuat, lebih efisien dibanding mengisi properti manual.
2. **Tugas 2 - Property dan Method (Class Mobil)**
   * Method seperti **ubahKecepatan**() dan **berhenti**() menggabungkan data (properti) dan perilaku (method).
   * Pendekatan ini membuat struktur kode lebih rapi karena perilaku objek langsung melekat pada objek itu sendiri.
3. **Tugas 3 - Abstract Class dan Factory Constructor (Bottle & CokeBottle)**
   * Abstract class mendefinisikan kontrak yang harus dipatuhi kelas turunannya.
   * Factory constructor memungkinkan fleksibilitas pembuatan objek, sehingga instance **Bottle** bisa menghasilkan **CokeBottle**.
4. **Tugas 4 - Studi Kasus Gosip Bekatul**
   * OOP digunakan untuk memodelkan entitas nyata: **Bebek** dan **KelompokGosip**.
   * Setiap class memiliki method yang memisahkan logika pengelompokan dan optimasi, sehingga memudahkan dalam menentukan distribusi bekatul terbaik dan menghitung total ketidakpuasan.

**4. Kesimpulan**

Dengan pendekatan OOP, struktur kode menjadi lebih terorganisir dan mudah dipahami karena menyerupai konsep dunia nyata. Class dapat digunakan kembali sebagai cetakan untuk membuat objek baru tanpa perlu menulis ulang logika yang sama, sehingga lebih efisien. Selain itu, karena data dan fungsi berada dalam satu kesatuan, penambahan fitur baru menjadi lebih mudah dan langsung relevan dengan objek yang bersangkutan.

**5. Link Repositori GitHub**

<https://github.com/ThaMhrdk/PratikumDPPB>

**TERIMAKASIH**