

	<div>LEMBAR JAWABAN</div> <div>Semester Ganjil/<del>Genap</del> Tahun Ajaran 2024/2025</div>																
<div>Maaf atas keterlambatan dalam pengumpulan tugas ini. Namun, saya dapat memastikan bahwa saya telah mengerjakan tugas ini dengan sungguh-sungguh dan penuh perhatian. Terima kasih atas pengertiannya.</div>																	
<table><tr><td>Nama Mahasiswa</td><td>:</td><td>Tirta Aditya</td></tr><tr><td>NIM</td><td>:</td><td>1203220057</td></tr><tr><td>Prodi</td><td>:</td><td>INFORMATIKA</td></tr><tr><td>Mata Kuliah</td><td>:</td><td>IMPLEMENTASI DAN PENGUJIAN</td></tr><tr><td>Tanggal Pengerjaan</td><td>:</td><td>22/10/2024</td></tr></table>		Nama Mahasiswa	:	Tirta Aditya	NIM	:	1203220057	Prodi	:	INFORMATIKA	Mata Kuliah	:	IMPLEMENTASI DAN PENGUJIAN	Tanggal Pengerjaan	:	22/10/2024	<div>Tanda Tangan Mahasiswa</div> <div></div> <div>TIRTA ADITYA</div>
Nama Mahasiswa	:	Tirta Aditya															
NIM	:	1203220057															
Prodi	:	INFORMATIKA															
Mata Kuliah	:	IMPLEMENTASI DAN PENGUJIAN															
Tanggal Pengerjaan	:	22/10/2024															

Penjelasan implementasi:

1. Hubungan dalam diagram:
- Interface **Transportasi** dengan method **bahanBakar** dan **kecepatan**
  - Kelas abstrak **AutoCarRpl** yang mengimplementasi interface **Transportasi**
  - Komposisi:
    - Satu **Setir**
    - Empat **Roda** (sesuai multiplisitas "4")
    - Satu **Mesin**
  - Pewarisan: **Mio** dan **Fuel** mewarisi **AutoCarRpl**
2. Fitur-fitur implementasi:
- Enkapsulasi menggunakan variabel private dan method public
  - Constructor untuk inisialisasi objek
  - Getter dan setter untuk akses properti
  - Method **start()** dan **drive()** sesuai diagram
  - Contoh penggunaan dalam method **main**

Link GitHub :

<https://github.com/Tirta7/IMPAL-ClassDiagram.git>

diagramClass.java C:\TIRTA\TUGAS TELKOM\SEMESTER 5\IMPLEM

```
1  // Interface Transportasi
2  interface Transportasi {
3      String bahanBakar();
4      int kecepatan();
5  }
6
7  // Class Setir
8  class Setir {
9      private String tipe;
10
11     public Setir(String tipe) {
12         this.tipe = tipe;
13     }
14
15     // Getters and setters
16     public String getTipe() {
17         return tipe;
18     }
19
20     public void setTipe(String tipe) {
21         this.tipe = tipe;
22     }
23 }
24
25 // Class Roda
26 class Roda {
27     private String jenis;
28
29     public Roda(String jenis) {
30         this.jenis = jenis;
31     }
32
33     // Getters and setters
34     public String getJenis() {
35         return jenis;
36     }
37
38     public void setJenis(String jenis) {
39         this.jenis = jenis;
40     }
41 }
42
43 // Class Mesin
44 class Mesin {
45     private String tipe;
46
47     public Mesin(String tipe) {
48         this.tipe = tipe;
49     }
50
51     // Getters and setters
52     public String getTipe() {
53         return tipe;
54     }
55
56     public void setTipe(String tipe) {
57         this.tipe = tipe;
58     }
59 }
60
61 // Class Mio
62 class Mio extends AutoCarRp1 {
63     public Mio() {
64         super("Bensin", 120);
65     }
66 }
67
```

```
67
68 // Class Fuel
69 class Fuel extends AutoCarRpl {
70     public Fuel() {
71         super("Solar", 100);
72     }
73 }
74
75 // Main Class AutoCarRpl
76 abstract class AutoCarRpl implements Transportasi {
77     private String bahanBakar;
78     private int kecepatan;
79     private Setir setir;
80     private List<Roda> roda;
81     private Mesin mesin;
82
83     public AutoCarRpl(String bahanBakar, int kecepatan) {
84         this.bahanBakar = bahanBakar;
85         this.kecepatan = kecepatan;
86         this.setir = new Setir("Standard");
87         this.roda = new ArrayList<>();
88         // Add 4 wheels as per diagram (4 multiplicity)
89         for (int i = 0; i < 4; i++) {
90             this.roda.add(new Roda("Standard"));
91         }
92         this.mesin = new Mesin("Standard");
93     }
94
95     @Override
96     public String bahanBakar() {
97         return this.bahanBakar;
98     }
99
100    @Override
101    public int kecepatan() {
102        return this.kecepatan;
103    }
104
105    // Methods from class diagram
106    public void start() {
107        System.out.println("Starting the car...");
108    }
109
110    public void drive() {
111        System.out.println("Driving the car...");
112    }
113
114    // Getters and setters
115    public Setir getSetir() {
116        return setir;
117    }
118
119    public List<Roda> getRoda() {
120        return roda;
121    }
122
123    public Mesin getMesin() {
124        return mesin;
125    }
126 }
127
```