

Nama : Oase Bimasena Ilhamaziiz

NIM : 245150707111059

Matkul : Prak. Pemrograman Lanjut

A. Static Method

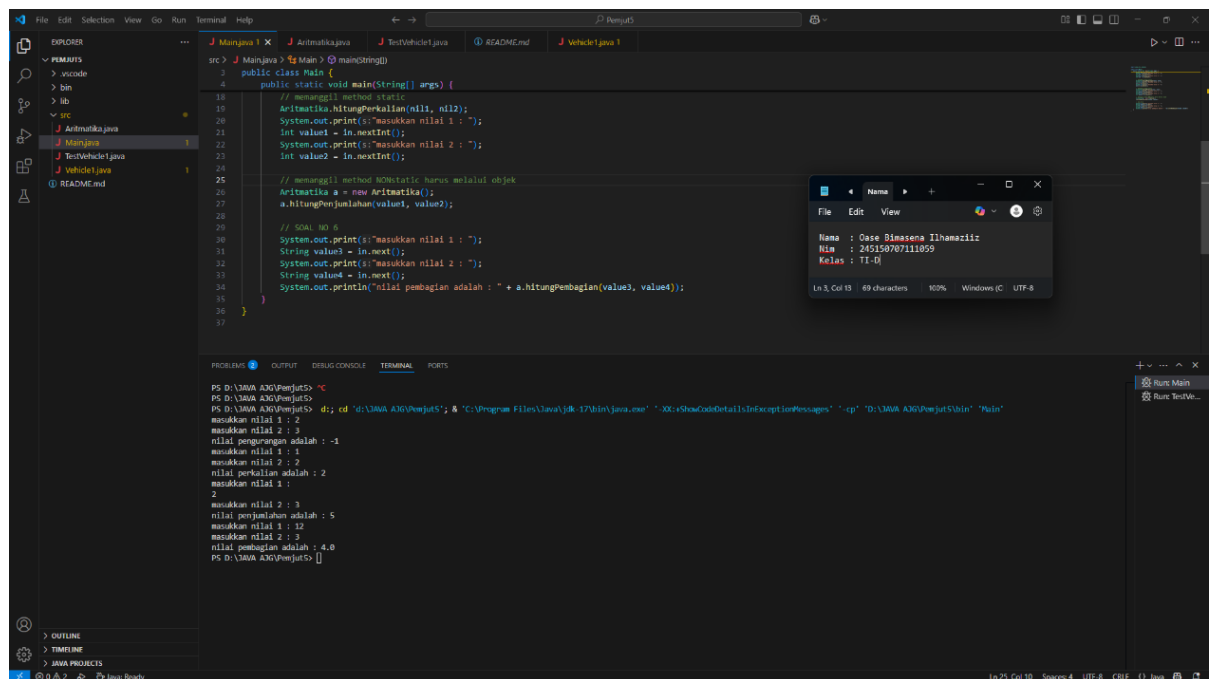
Pertanyaan

1. Apakah yang disebut dengan static variabel? Dan apa fungsi dari static variabel serta kapan kita dapat menggunakan static variabel?
2. Mengapa pada main method harus dituliskan static? Jelaskan jawaban anda beserta dengan alasan!
3. Lakukan percobaan diatas dan benahi jika menemukan kesalahan!
4. Jika pada tubuh method hitungPenjumlahan ditambahkan syntax hitungPerkalian(a,b)apa yang terjadi? Jelaskan?
5. Jika pada tubuh method hitungPerkalian ditambahkan syntax hitungPenjumlahan(a,b)apa yang terjadi? Jelaskan?
6. Tambahkan method non static dengan nilai balikan double untuk menghitung pembagiandengan parameter String nil dan String nil2, dan panggil method tersebut pada methodmain!

Jawab

1. Static variabel adalah variabel dalam Java yang dideklarasikan dengan kata kunci "static" dan menjadi milik kelas itu sendiri, bukan objek individual, sehingga nilainya dibagikan ke semua instance kelas tersebut. Fungsi utamanya adalah untuk menyimpan data yang sama untuk semua objek, menghemat memori karena hanya ada satu salinan, serta memungkinkan berbagi nilai antar objek tanpa perlu membuat variabel terpisah. Static variabel cocok digunakan untuk data yang bersifat global seperti konstanta ($\pi = 3.14$) atau informasi bersama (nama sekolah), tetapi tidak digunakan untuk data unik tiap objek (seperti nama siswa).
2. Method main harus dideklarasikan static karena method ini merupakan entry point program Java yang dipanggil oleh JVM (Java Virtual Machine) sebelum objek apa pun dibuat. Karena JVM memanggil method main tanpa membuat instance dari kelas yang mengandungnya, method tersebut harus bersifat static agar bisa diakses langsung melalui kelas tanpa perlu instansiasi objek terlebih dahulu. Jika main tidak static, JVM tidak akan bisa menemukan atau menjalankannya saat program dimulai, sehingga kode tidak akan bisa dijalankan. Dengan kata lain, sifat static memungkinkan eksekusi program dimulai tanpa ketergantungan pada pembuatan objek, yang sesuai dengan kebutuhan awal saat sebuah aplikasi Java pertama kali dijalankan.

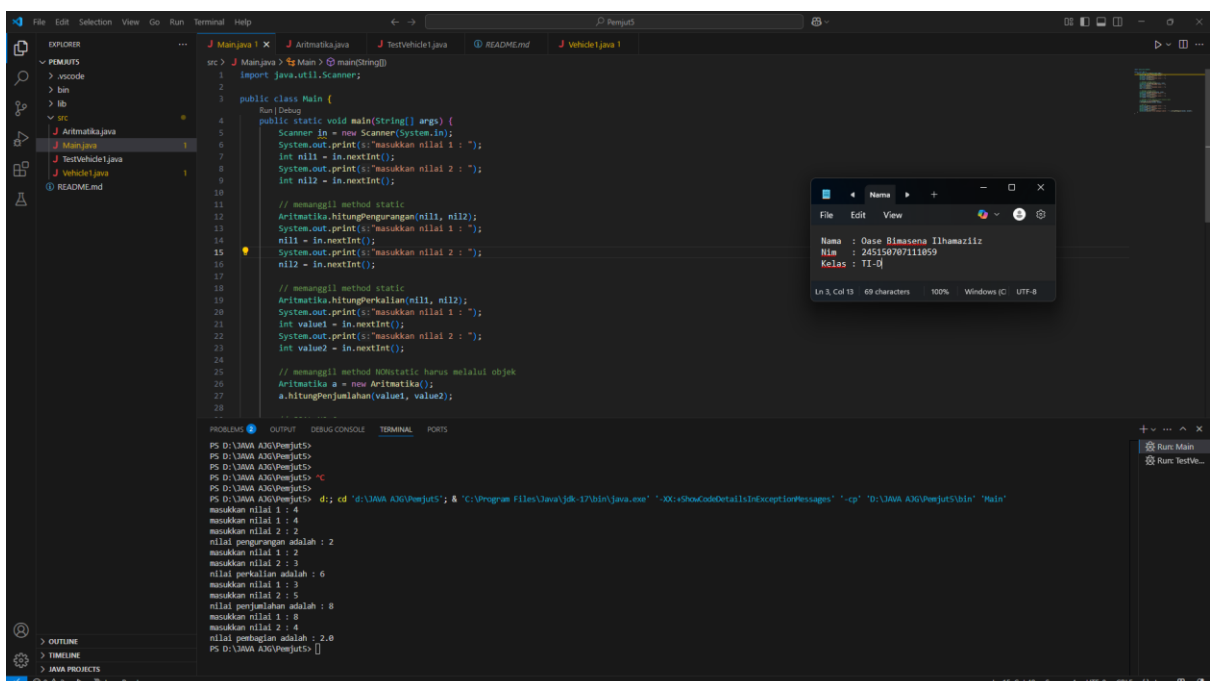
3. Program berjalan dengan baik tanpa adanya error yang terjadi.



```
src > J Main > @mainString()
3 public class Main {
4     public static void main(String[] args) {
5         // memanggil method static
6         Aritmatika.hitungPerkalian(nil1, nil2);
7         System.out.print("masukkan nilai 1 : ");
8         int value1 = in.nextInt();
9         System.out.print("masukkan nilai 2 : ");
10        int value2 = in.nextInt();
11
12        // memanggil method nonstatic harus melalui objek
13        Aritmatika a = new Aritmatika();
14        a.hitungPenjumlahan(value1, value2);
15
16        // SOAL NO 6
17        System.out.print("masukkan nilai 1 : ");
18        String value3 = in.next();
19        System.out.print("masukkan nilai 2 : ");
20        String value4 = in.next();
21        System.out.println("nilai pembagian adalah : " + a.hitungPembagian(value3, value4));
22    }
23 }
```

```
PS D:\JAVA A3G\Penjurut5> -C
PS D:\JAVA A3G\Penjurut5>
PS D:\JAVA A3G\Penjurut5> d; cd "d:\JAVA A3G\Penjurut5"; & "C:\Program Files\Java\jdk-17\bin\java.exe" "-XX:+ShowCodeDetailsInExceptionMessages" "-cp" "D:\JAVA A3G\Penjurut5\bin" "Main"
masukkan nilai 1 : 2
masukkan nilai 2 : 3
nilai pengurangan adalah : -1
masukkan nilai 1 : 1
masukkan nilai 2 : 2
nilai perkalian adalah : 2
masukkan nilai 1 : 2
masukkan nilai 2 : 3
nilai penjumlahan adalah : 5
masukkan nilai 1 : 12
masukkan nilai 2 : 3
nilai pembagian adalah : 4.0
PS D:\JAVA A3G\Penjurut5> []
```

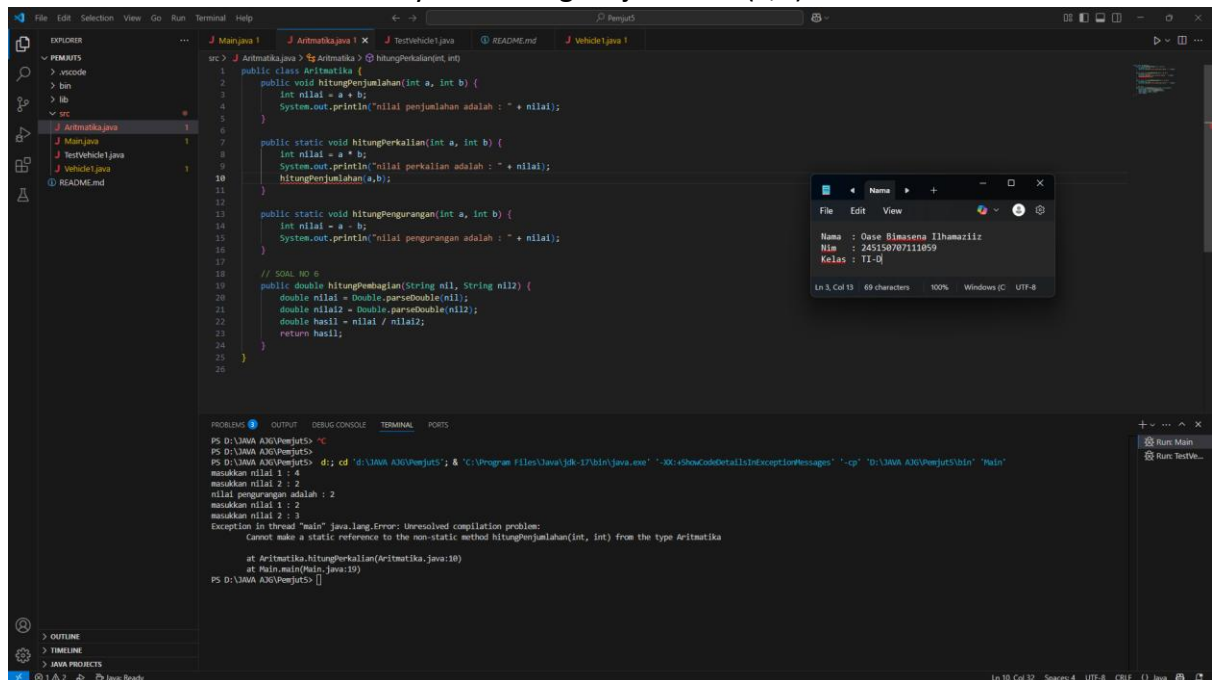
4. Jika kita menambahkan syntax *hitungPerkalian(a,b)* pada method hitungPejumlahan, maka tidak akan terjadi error karena modifier yang terdapat pada method hitungPerkalian adalah modifier static sehingga method ini bisa diakses dimana saja tanpa kita membuat objek.



```
src > J Main > @mainString()
1 import java.util.Scanner;
2
3 public class Main {
4     Run | Debug
5     public static void main(String[] args) {
6         Scanner in = new Scanner(System.in);
7         System.out.print("masukkan nilai 1 : ");
8         int nil1 = in.nextInt();
9         System.out.print("masukkan nilai 2 : ");
10        int nil2 = in.nextInt();
11
12        // memanggil method static
13        Aritmatika.hitungPengurangan(nil1, nil2);
14        System.out.print("masukkan nilai 1 : ");
15        nil1 = in.nextInt();
16        System.out.print("masukkan nilai 2 : ");
17        nil2 = in.nextInt();
18
19        // memanggil method static
20        Aritmatika.hitungPerkalian(nil1, nil2);
21        System.out.print("masukkan nilai 1 : ");
22        int value1 = in.nextInt();
23        System.out.print("masukkan nilai 2 : ");
24        int value2 = in.nextInt();
25
26        // memanggil method nonstatic harus melalui objek
27        Aritmatika a = new Aritmatika();
28        a.hitungPenjumlahan(value1, value2);
29    }
30 }
```

```
PS D:\JAVA A3G\Penjurut5>
PS D:\JAVA A3G\Penjurut5>
PS D:\JAVA A3G\Penjurut5> -C
PS D:\JAVA A3G\Penjurut5> d; cd "d:\JAVA A3G\Penjurut5"; & "C:\Program Files\Java\jdk-17\bin\java.exe" "-XX:+ShowCodeDetailsInExceptionMessages" "-cp" "D:\JAVA A3G\Penjurut5\bin" "Main"
masukkan nilai 1 : 4
masukkan nilai 2 : 2
nilai pengurangan adalah : 2
masukkan nilai 1 : 2
masukkan nilai 2 : 3
nilai perkalian adalah : 6
masukkan nilai 1 : 3
masukkan nilai 2 : 5
nilai penjumlahan adalah : 8
masukkan nilai 1 : 8
masukkan nilai 2 : 4
nilai pembagian adalah : 2.0
PS D:\JAVA A3G\Penjurut5> []
```

5. Tidak seperti no.4, jika kita menambahkan syntax hitungPenjumlahan(a,b) di method hitungPerkalian, maka akan muncul error yang disebabkan tidak adanya objek yang bisa mengakses method non-static tersebut, sehingga kita perlu membuat objeknya dulu baru bisa menambahkan syntax hitungPenjumlahan(a,b).



The screenshot shows the VS Code editor with a Java project named 'Penjurut'. The 'Aritmatika.java' file is open, showing the following code:

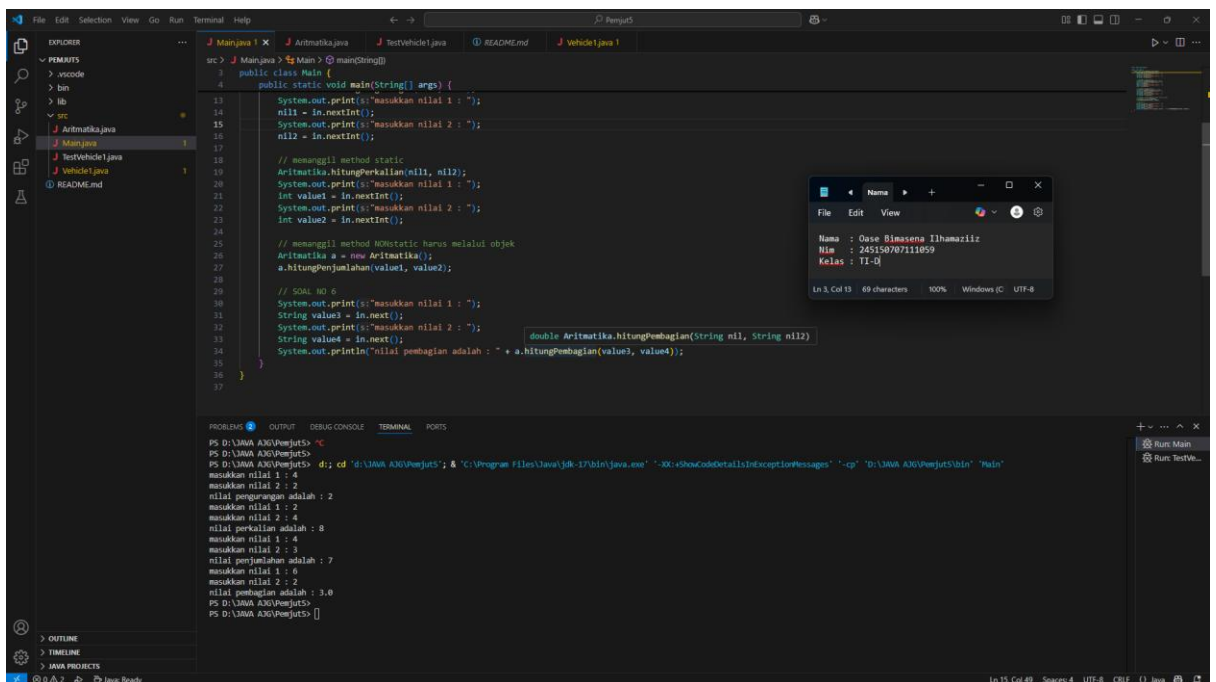
```
1 public class Aritmatika {
2     public void hitungPenjumlahan(int a, int b) {
3         int nilai = a + b;
4         System.out.println("nilai penjumlahan adalah : " + nilai);
5     }
6
7     public static void hitungPerkalian(int a, int b) {
8         int nilai = a * b;
9         System.out.println("nilai perkalian adalah : " + nilai);
10        hitungPenjumlahan(a,b);
11    }
12
13    public static void hitungPengurangan(int a, int b) {
14        int nilai = a - b;
15        System.out.println("nilai pengurangan adalah : " + nilai);
16    }
17
18    // SOAL NO 6
19    public double hitungPembagian(String nil1, String nil2) {
20        double nilai = Double.parseDouble(nil1);
21        double nilai2 = Double.parseDouble(nil2);
22        double hasil = nilai / nilai2;
23        return hasil;
24    }
25 }
```

The terminal output shows the following error:

```
PS D:\JAWA A30\Penjurut> cd .; cd "D:\JAWA A30\Penjurut"; & "C:\Program Files\Java\jdk-17\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp "D:\JAWA A30\Penjurut\bin" Main
masukkan nilai 1 : 4
masukkan nilai 2 : 2
nilai pengurangan adalah : 2
masukkan nilai 1 : 2
masukkan nilai 2 : 3
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
Cannot make a static reference to the non-static method hitungPenjumlahan(int, int) from the type Aritmatika
    at Aritmatika.hitungPerkalian(Aritmatika.java:10)
    at Main.main(Main.java:19)
PS D:\JAWA A30\Penjurut>
```

The error message indicates that the static method `hitungPerkalian` is trying to call the non-static method `hitungPenjumlahan` without creating an instance of the `Aritmatika` class.

6. Tambahan method non-static

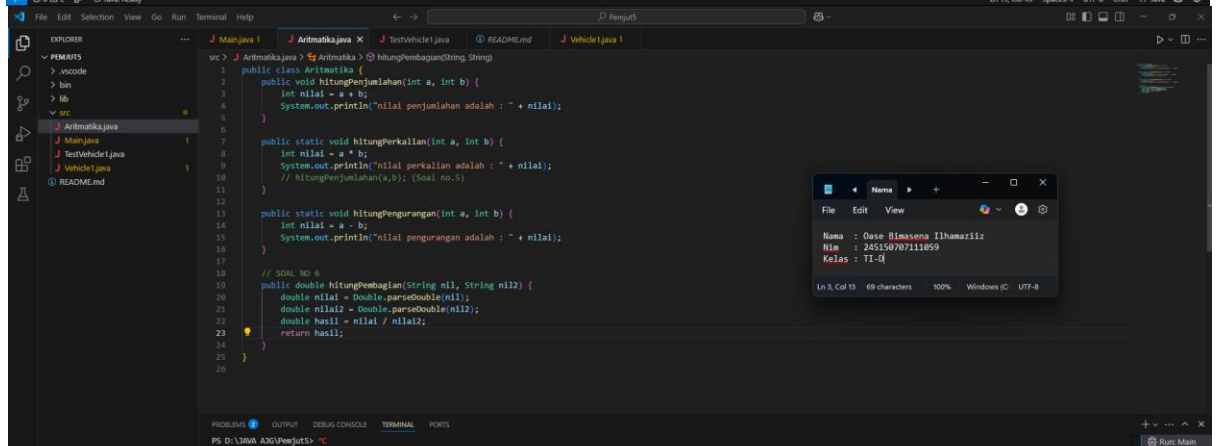


The screenshot shows the VS Code editor with the 'Aritmatika.java' file open. The code is as follows:

```
1 public class Aritmatika {
2     public void hitungPenjumlahan(int a, int b) {
3         int nilai = a + b;
4         System.out.println("nilai penjumlahan adalah : " + nilai);
5     }
6
7     public static void hitungPerkalian(int a, int b) {
8         int nilai = a * b;
9         System.out.println("nilai perkalian adalah : " + nilai);
10        hitungPenjumlahan(a,b); // Soal no.5
11    }
12
13    public static void hitungPengurangan(int a, int b) {
14        int nilai = a - b;
15        System.out.println("nilai pengurangan adalah : " + nilai);
16    }
17
18    // SOAL NO 6
19    public double hitungPembagian(String nil1, String nil2) {
20        double nilai = Double.parseDouble(nil1);
21        double nilai2 = Double.parseDouble(nil2);
22        double hasil = nilai / nilai2;
23        return hasil;
24    }
25 }
```

The terminal output shows the following output:

```
PS D:\JAWA A30\Penjurut> cd .; cd "D:\JAWA A30\Penjurut"; & "C:\Program Files\Java\jdk-17\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp "D:\JAWA A30\Penjurut\bin" Main
masukkan nilai 1 : 4
masukkan nilai 2 : 2
nilai pengurangan adalah : 2
masukkan nilai 1 : 2
masukkan nilai 2 : 4
nilai perkalian adalah : 8
masukkan nilai 1 : 4
masukkan nilai 2 : 3
nilai penjumlahan adalah : 7
masukkan nilai 1 : 6
masukkan nilai 2 : 2
nilai pembagian adalah : 3.0
PS D:\JAWA A30\Penjurut>
```



The screenshot shows the VS Code editor with the 'Aritmatika.java' file open. The code is as follows:

```
1 public class Aritmatika {
2     public void hitungPenjumlahan(int a, int b) {
3         int nilai = a + b;
4         System.out.println("nilai penjumlahan adalah : " + nilai);
5     }
6
7     public static void hitungPerkalian(int a, int b) {
8         int nilai = a * b;
9         System.out.println("nilai perkalian adalah : " + nilai);
10        hitungPenjumlahan(a,b); // Soal no.5
11    }
12
13    public static void hitungPengurangan(int a, int b) {
14        int nilai = a - b;
15        System.out.println("nilai pengurangan adalah : " + nilai);
16    }
17
18    // SOAL NO 6
19    public double hitungPembagian(String nil1, String nil2) {
20        double nilai = Double.parseDouble(nil1);
21        double nilai2 = Double.parseDouble(nil2);
22        double hasil = nilai / nilai2;
23        return hasil;
24    }
25 }
```

The terminal output shows the following output:

```
PS D:\JAWA A30\Penjurut> cd .; cd "D:\JAWA A30\Penjurut"; & "C:\Program Files\Java\jdk-17\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp "D:\JAWA A30\Penjurut\bin" Main
masukkan nilai 1 : 4
masukkan nilai 2 : 2
nilai pengurangan adalah : 2
masukkan nilai 1 : 2
masukkan nilai 2 : 4
nilai perkalian adalah : 8
masukkan nilai 1 : 4
masukkan nilai 2 : 3
nilai penjumlahan adalah : 7
masukkan nilai 1 : 6
masukkan nilai 2 : 2
nilai pembagian adalah : 3.0
PS D:\JAWA A30\Penjurut>
```

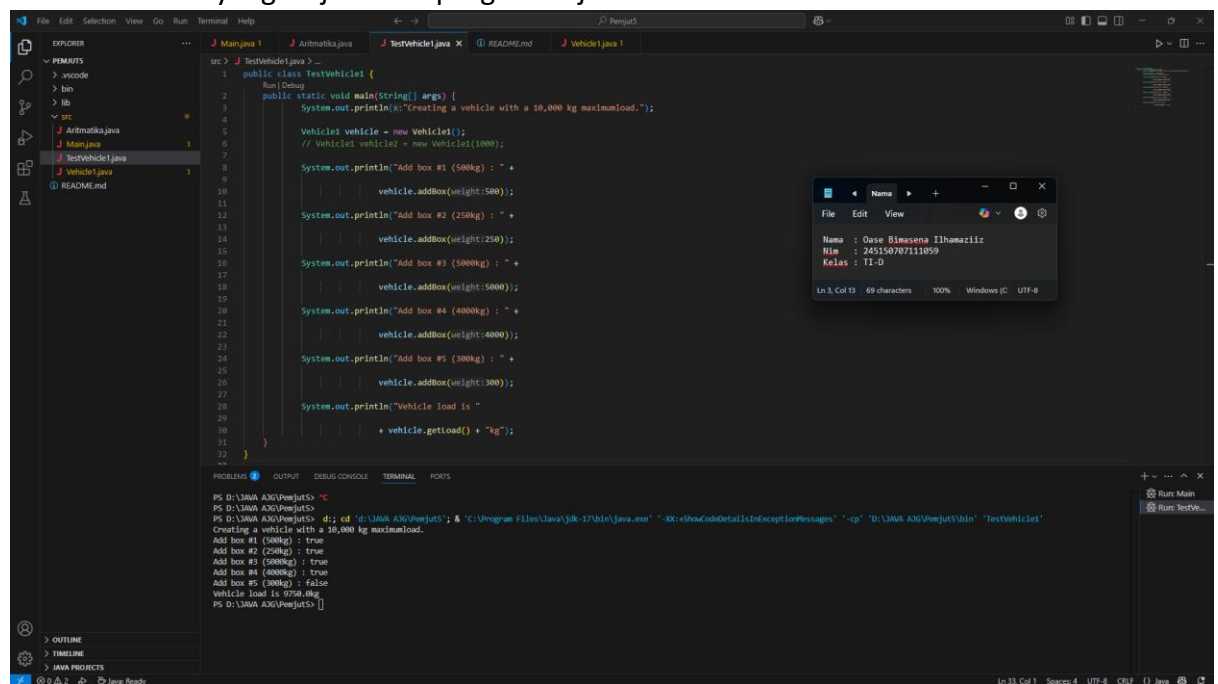
B. Konstanta Final

Pertanyaan

1. Benahi kode Vehicle1 dan TestVehicle1 dan perbaiki jika menemui kesalahan!
2. Hapus separator "/" pada file Vehicle1.java pada baris 4-6 serta pada file TestVehicle1.java pada baris 6, apa yang terjadi dan jelaskan!
3. Pada file Vehicle1.java variabel load ubah menjadi konstanta final, apa yang terjadi, jelaskan!
4. Tambahkan keyword "static" pada file Vehicle1.java variabel maxLoad, apa yang terjadi dan jelaskan!

Jawab

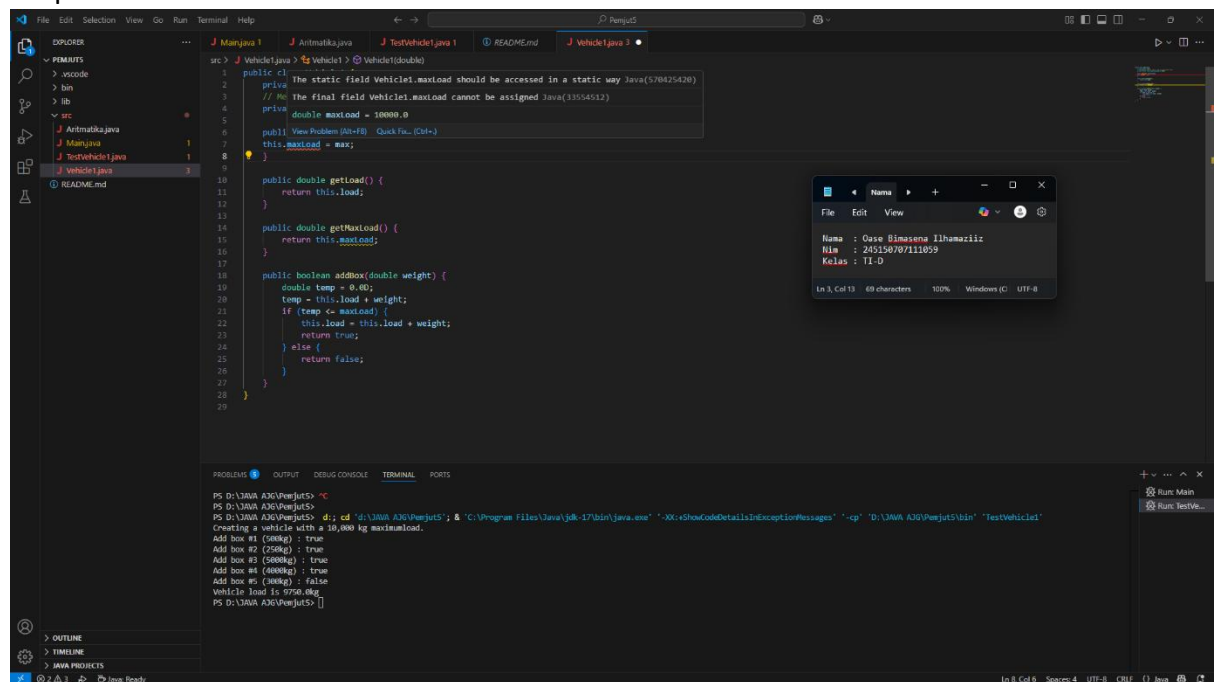
1. Tidak ada error yang terjadi saat program dijalankan.



```
src > TestVehicle1.java > ...
1 public class TestVehicle1 {
2     public static void main(String[] args) {
3         System.out.println("Creating a vehicle with a 10,000 kg maxLoad.");
4         Vehicle1 vehicle = new Vehicle1();
5         // Vehicle1 vehicle2 = new Vehicle1(1000);
6         System.out.println("Add box #1 (500kg) : " +
7             vehicle.addBox(500));
8         System.out.println("Add box #2 (250kg) : " +
9             vehicle.addBox(250));
10        System.out.println("Add box #3 (5000kg) : " +
11            vehicle.addBox(5000));
12        System.out.println("Add box #4 (4000kg) : " +
13            vehicle.addBox(4000));
14        System.out.println("Add box #5 (300kg) : " +
15            vehicle.addBox(300));
16        System.out.println("Vehicle load is "
17            + vehicle.getLoad() + "kg");
18    }
19 }
```

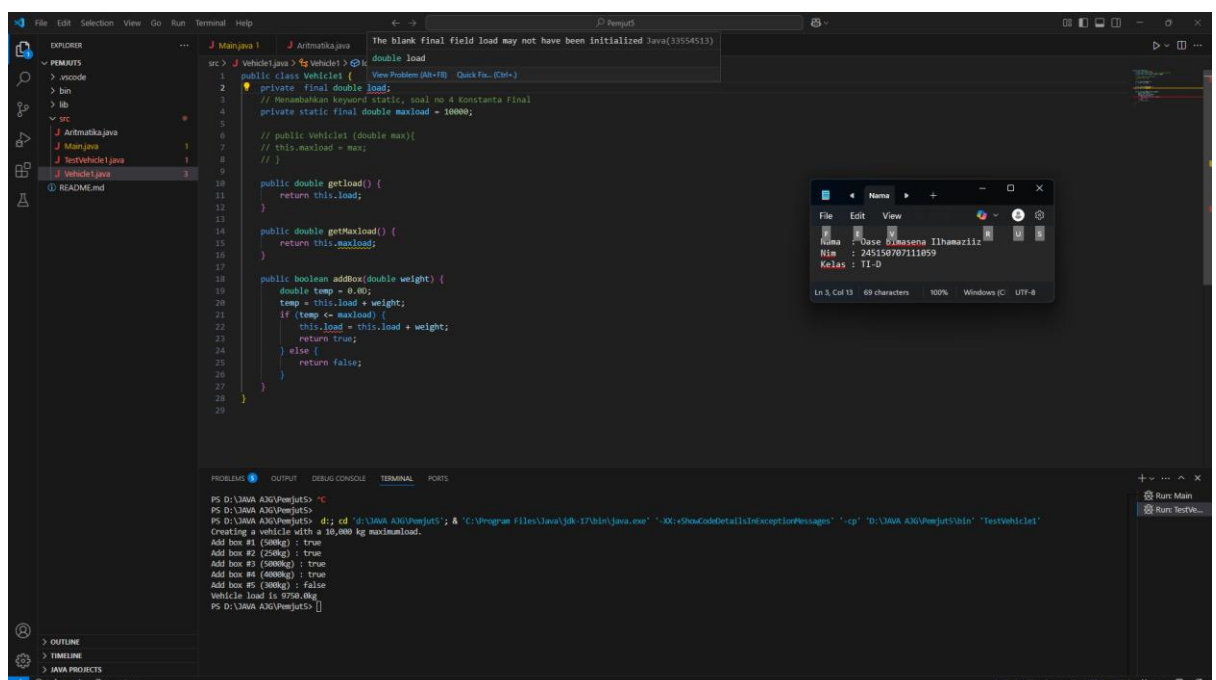
```
PS D:\JANA AIG\Uppjuts> .C
PS D:\JANA AIG\Uppjuts>
PS D:\JANA AIG\Uppjuts> cd "d:\JANA AIG\Uppjuts" & "C:\Program Files\Java\jdk-17\bin\java.exe" "-Xmx65536m" "-cp" "D:\JANA AIG\Uppjuts\bin" "TestVehicle1"
Creating a vehicle with a 10,000 kg maxLoad.
Add box #1 (500kg) : true
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add box #5 (300kg) : false
Vehicle load is 9750.0kg
PS D:\JANA AIG\Uppjuts> ]
```

2. Ada 2 kesalahan yang ditemukan. Kesalahan pertama ada di kelas Vehicle1, yaitu membuat suatu konstruktor berparameter double, yang nantinya parameter tersebut akan merubah nilai dari variabel maxLoad yang sudah di-finalkan nilainya 10000, sehingga tidak bisa dirubah sama sekali meskipun di assignment dan menimbulkan error. Kesalahan kedua terdapat di kelas TestVehicle1, yang dimana ketika konstruktor berparameter sudah dibuat di kelas Vehicle1, maka konstruktor default yang awalnya tidak perlu dibuat harus dibuat secara manual karena sudah ada konstruktor yang berparameter.



```
1 public class Vehicle1 {
2     private static final double maxLoad = 10000.0;
3     // The final field Vehicle1.maxLoad cannot be assigned Java(33554512)
4     private double maxLoad = 10000.0;
5     public Vehicle1(double max) {
6         this.maxLoad = max;
7     }
8     public double getLoad() {
9         return this.load;
10    }
11    public double getMaxLoad() {
12        return this.maxLoad;
13    }
14    public boolean addBox(double weight) {
15        double temp = 0.0;
16        temp = this.load + weight;
17        if (temp <= maxLoad) {
18            this.load = temp;
19            return true;
20        } else {
21            return false;
22        }
23    }
24 }
```

3. Jika mengubah variabel load pada file Vehicle1.java menjadi konstanta final, maka akan muncul error. Hal ini dikarenakan konstanta final harus sudah diinisialisasi nilainya terlebih dahulu ketika variabel tersebut dideklarasikan atau variabel tersebut diinisialisasi nilainya di dalam sebuah konstruktor dan nilainya tidak dapat dirubah kembali, sehingga tidak bisa diinisialisasi di method biasa.



```
1 public class Vehicle1 {
2     private final double load;
3     // The blank final field load may not have been initialized Java(33554513)
4     private static final double maxLoad = 10000.0;
5     public Vehicle1(double max) {
6         this.maxLoad = max;
7     }
8     public double getLoad() {
9         return this.load;
10    }
11    public double getMaxLoad() {
12        return this.maxLoad;
13    }
14    public boolean addBox(double weight) {
15        double temp = 0.0;
16        temp = this.load + weight;
17        if (temp <= maxLoad) {
18            this.load = temp;
19            return true;
20        } else {
21            return false;
22        }
23    }
24 }
```

4. Sesudah menambahkan keyword "static" pada variabel maxLoad di file Vehicle1.java, tidak ada perubahan signifikan terjadi. Hal ini dikarenakan modifier static hanya mempengaruhi apakah variabel tersebut bagian dari kelas atau dari sebuah objek, dan karena variabelnya sudah final, maka tidak ada yang bisa merubah nilai dari variabel tersebut dan menambahkan static hanya untuk mempermudah penggunaan variabel tersebut saja untuk kelas lain jika dibutuhkan.