Nama : Arditya Adjie Rosandi

NIM : 20230801274

Pemrogaman Berorientasi Objek

1. Operasi Bilangan 1

```java
package Pertemuan11.OperasiBilangan1;

public class OperasiBilangan {
    protected double a, b, c;

    protected void set_A(double a) { this.a = a; }
    protected void set_B(double b) { this.b = b; }
    protected void set_C(double c) { this.c = c; }
    protected double get_A() { return a; }
    protected double get_B() { return b; }
    protected double get_C() { return c; }

    protected void tampil() {
        System.out.println("HASIL OPERASI : " + c);
    }
}
```

```java
package Pertemuan11.OperasiBilangan1;

public class OperasiPembagian extends OperasiBilangan {

    @Override
    protected void tampil() {
        if (b != 0) {
            c = a / b;
            System.out.println("PEMBAGIAN : " + a + " / " + b + " = " + c);
        }
        else {
            System.out.println("TIDAK DAPAT MEMBAGI DENGAN NOL.");
        }
    }
}
```

```java
package Pertemuan11.OperasiBilangan1;

public class OperasiPengurangan extends OperasiBilangan {

    @Override
    protected void tampil() {
        c = a - b;
        System.out.println("PENGURANGAN : " + a + " - " + b + " = " + c );
    }
}
```

```java
package Pertemuan11.OperasiBilangan1;

public class OperasiPenjumlahan extends OperasiBilangan {

    @Override
    protected void tampil() {
        c = a + b;
        System.out.println("PENJUMLAHAN : " + a + " + " + b + " = " + c );
    }
}
```

```java
package Pertemuan11.OperasiBilangan1;

public class OperasiPerkalian extends OperasiBilangan {

    @Override
    protected void tampil() {
        c = a * b;
        System.out.println("PERKALIAN : " + a + " * " + b + " = " + c );
    }
}
```

```java
package Pertemuan11.OperasiBilangan1;

public class PrintOperasi {
    public void cetakSemua(OperasiBilangan[] OB, double a, double b) {
        for (OperasiBilangan operasi : OB) {
            operasi.set_A(a);
            operasi.set_B(b);
            operasi.tampil();
        }
    }
}
```

```java
package Pertemuan11.OperasiBilangan1;

public class Main {
    Run | Debug
    public static void main(String[] args) {

        System.out.print("\033[H\033[2J");
        System.out.println("OPERASI ARTIMATIKA JAVA\n");

        // Nilai A dan B
        double a = 10.5;
        double b = 0.5;

        // Array Operasi
        OperasiBilangan[] operasiArray = {
            new OperasiPenjumlahan(),
            new OperasiPengurangan(),
            new OperasiPerkalian(),
            new OperasiPembagian()
        };

        // Cetak semua operasi
        PrintOperasi cetak = new PrintOperasi();
        cetak.cetakSemua(operasiArray, a, b);
    }
}
```

1. Kelas OperasiBilangan (Superclass)

- Kelas ini adalah kelas dasar yang menyediakan atribut **a, b,** dan **c**, yang digunakan untuk menyimpan dua bilangan input (a dan b), serta hasil operasi yang disimpan dalam c.
- Ada juga beberapa **metode setter** dan **getter** untuk mengatur dan mengambil nilai dari atribut a, b, dan c.
- Metode **tampil()** digunakan untuk menampilkan hasil operasi, tetapi di kelas ini, hanya mencetak hasil yang disimpan dalam c tanpa melakukan perhitungan apapun. Metode ini akan dioverride pada subclass.

2. Kelas OperasiPembagian (Subclass)

- Kelas ini mewarisi dari OperasiBilangan dan mengimplementasikan metode **tampil()** untuk menghitung pembagian antara a dan b.
- **Periksa pembagiannya**: Sebelum melakukan pembagian, program memeriksa apakah b tidak sama dengan nol. Jika b == 0, maka akan mencetak pesan error ("TIDAK DAPAT MEMBAGI DENGAN NOL.") untuk mencegah kesalahan pembagian oleh nol.
- Jika b != 0, maka hasil pembagian disimpan dalam c dan ditampilkan.

3. Kelas OperasiPengurangan (Subclass)

- Kelas ini juga mewarisi dari OperasiBilangan dan mengimplementasikan metode **tampil()** untuk menghitung pengurangan antara a dan b.
- Hasil pengurangan disimpan dalam c dan dicetak dalam format "PENGURANGAN : a - b = c".

4. Kelas OperasiPenjumlahan (Subclass)

- Kelas ini juga mewarisi dari OperasiBilangan dan mengimplementasikan metode **tampil()** untuk menghitung penjumlahan antara a dan b.
- Hasil penjumlahan disimpan dalam c dan ditampilkan dalam format "PENJUMLAHAN : a + b = c".

5. Kelas OperasiPerkalian (Subclass)

- Kelas ini mewarisi dari OperasiBilangan dan mengimplementasikan metode **tampil()** untuk menghitung perkalian antara a dan b.
- Hasil perkalian disimpan dalam c dan dicetak dalam format "PERKALIAN : a * b = c".

6. Kelas Main

- Kelas Main adalah titik masuk program. Di sini, beberapa hal dilakukan:
    - Nilai untuk a dan b diinisialisasi (masing-masing dengan nilai 10.5 dan 0.5).
    - Array **operasiArray** didefinisikan, yang berisi objek dari kelas OperasiPenjumlahan, OperasiPengurangan, OperasiPerkalian, dan OperasiPembagian. Ini menciptakan polimorfisme, karena meskipun semua objek memiliki metode tampil(), implementasi masing-masing berbeda tergantung pada kelasnya.
    - **Instansi kelas PrintOperasi** digunakan untuk mencetak hasil operasi. Kelas ini (meskipun tidak diberikan dalam kode) tampaknya memiliki metode cetakSemua() yang akan melakukan iterasi terhadap array operasiArray dan menjalankan metode tampil() pada masing-masing objek dalam array, dengan nilai a dan b yang diberikan.

Penjelasan Polimorfisme Dinamis

- **Polimorfisme dinamis** tercermin dalam penggunaan array **OperasiBilangan[]** yang menyimpan objek dari kelas OperasiPenjumlahan, OperasiPengurangan, OperasiPerkalian, dan OperasiPembagian.
- Meskipun array tersebut hanya menyimpan objek dari superclass OperasiBilangan, ketika metode tampil() dipanggil, Java secara otomatis memanggil implementasi yang sesuai dari masing-masing subclass (berdasarkan objek yang sesungguhnya, bukan tipe variabel). Ini adalah contoh polimorfisme dinamis, di mana pemanggilan metode yang tepat ditentukan saat runtime berdasarkan tipe objek yang ada.

Output

OPERASI ARTIMATIKA JAVA

PEMBAGIAN : 10.5 / 0.5 = 21.0

PENGURANGAN : 10.5 - 0.5 = 10.0

PERKALIAN : 10.5 * 0.5 = 5.25

PENJUMLAHAN : 10.5 + 0.5 = 11.0

2. Operasi BIlangan 2

=> Abstract

```java
package Pertemuan11.OperasiBilangan2.Abstract;

public abstract class OperasiBilanganAbs {
    protected double a, b, c;

    public abstract void set_A(double a);
    public abstract void set_B(double b);
    public abstract void set_C();
    public abstract double get_A();
    public abstract double get_B();
    public abstract double get_C();
    public abstract void tampil();
}
```

=> Final

```java
package Pertemuan11.OperasiBilangan2.Final;

import Pertemuan11.OperasiBilangan2.Abstract.OperasiBilanganAbs;

public final class OperasiBilanganAbsCetak {

    public void cetakSemua(OperasiBilanganAbs[] OB, double a, double b) {
        for (OperasiBilanganAbs operasi : OB) {
            operasi.set_A(a);
            operasi.set_B(b);
            operasi.set_C();
            operasi.tampil();
        }
    }
}
```

=> Polimorfisme

```java
import Pertemuan11.OperasiBilangan2.Abstract.OperasiBilanganAbs;

public class OperasiPembagian extends OperasiBilanganAbs {

    @Override
    public void set_A(double a) {
        this.a = a;
    }

    @Override
    public void set_B(double b) {
        this.b = b;
    }

    @Override
    public void set_C() {
        if (b != 0) {
            this.c = this.a / this.b;
        } else {
            System.out.println("TIDAK DAPAT MEMBAGI DENGAN NOL.");
            this.c = 0;
        }
    }

    @Override
    public double get_A() {
        return this.a;
    }

    @Override
    public double get_B() {
        return this.b;
    }

    @Override
    public double get_C() {
        return this.c;
    }

    @Override
    public void tampil() {
        System.out.println("HASIL PEMBAGIAN : " + this.get_C());
    }
}
```

```java
import Pertemuan11.OperasiBilangan2.Abstract.OperasiBilanganAbs;

public class OperasiPengurangan extends OperasiBilanganAbs {

    @Override
    public void set_A(double a) {
        this.a = a;
    }

    @Override
    public void set_B(double b) {
        this.b = b;
    }

    @Override
    public void set_C() {
        this.c = this.a - this.b;
    }

    @Override
    public double get_A() {
        return this.a;
    }

    @Override
    public double get_B() {
        return this.b;
    }

    @Override
    public double get_C() {
        return this.c;
    }

    @Override
    public void tampil() {
        System.out.println("HASIL PENGURANGAN : " + this.get_C());
    }
}
```

```java
import Pertemuan11.OperasiBilangan2.Abstract.OperasiBilanganAbs;

public class OperasiPenjumlahan extends OperasiBilanganAbs {

    @Override
    public void set_A(double a) {
        this.a = a;
    }

    @Override
    public void set_B(double b) {
        this.b = b;
    }

    @Override
    public void set_C() {
        this.c = this.a + this.b;
    }

    @Override
    public double get_A() {
        return this.a;
    }

    @Override
    public double get_B() {
        return this.b;
    }

    @Override
    public double get_C() {
        return this.c;
    }

    @Override
    public void tampil() {
        System.out.println("HASIL PENJUMLAHAN : " + this.get_C());
    }
}
```

```java
import Pertemuan11.OperasiBilangan2.Abstract.OperasiBilanganAbs;

public class OperasiPerkalian extends OperasiBilanganAbs {

    @Override
    public void set_A(double a) {
        this.a = a;
    }

    @Override
    public void set_B(double b) {
        this.b = b;
    }

    @Override
    public void set_C() {
        this.c = this.a * this.b;
    }

    @Override
    public double get_A() {
        return this.a;
    }

    @Override
    public double get_B() {
        return this.b;
    }

    @Override
    public double get_C() {
        return this.c;
    }

    @Override
    public void tampil() {
        System.out.println("HASIL PERKALIAN : " + this.get_C());
    }
}
```

```java
public class Main {
    Run | Debug
    public static void main(String[] args) {

        System.out.print("\033[H\033[2J");
        System.out.println("OPERASI ARITMATIKA JAVA\n");
        double A = 6.5, B = 0.5;

        // Array untuk polimorfisme
        OperasiBilanganAbs[] operasiBilangan = {
            new OperasiPenjumlahan(),
            new OperasiPengurangan(),
            new OperasiPerkalian(),
            new OperasiPembagian()
        };

        // Final class untuk mencetak
        OperasiBilanganAbsCetak cetak = new OperasiBilanganAbsCetak();
        cetak.cetakSemua(operasiBilangan, A, B);
    }
}
```

1. Kelas OperasiBilanganAbs

- **Deskripsi:**
  OperasiBilanganAbs adalah kelas abstrak yang mendefinisikan struktur dasar untuk operasi matematika. Kelas ini memiliki atribut a, b, dan c, yang mewakili dua angka yang akan dihitung (a dan b), serta hasil perhitungan (c). Kelas ini juga mendeklarasikan beberapa metode abstrak yang harus diimplementasikan oleh kelas turunan:
    - set_A(double a) untuk menetapkan nilai a.
    - set_B(double b) untuk menetapkan nilai b.
    - set_C() untuk menetapkan hasil perhitungan.
    - get_A(), get_B(), dan get_C() untuk mendapatkan nilai a, b, dan hasilnya (c).
    - tampil() untuk menampilkan hasil perhitungan.

2. Kelas OperasiPenjumlahan

- **Deskripsi:**
  OperasiPenjumlahan adalah kelas turunan dari OperasiBilanganAbs yang mengimplementasikan metode set_A(), set_B(), dan set_C() untuk operasi penjumlahan.
    - set_A(double a) menetapkan nilai a.
    - set_B(double b) menetapkan nilai b.
    - set_C() menghitung hasil penjumlahan antara a dan b.
    - tampil() menampilkan hasil penjumlahan.

3. Kelas OperasiPengurangan

- **Deskripsi:**
  OperasiPengurangan adalah kelas turunan dari OperasiBilanganAbs yang mengimplementasikan metode set_A(), set_B(), dan set_C() untuk operasi pengurangan.
    - set_A(double a) menetapkan nilai a.
    - set_B(double b) menetapkan nilai b.
    - set_C() menghitung hasil pengurangan antara a dan b.
    - tampil() menampilkan hasil pengurangan.

4. Kelas OperasiPerkalian

- **Deskripsi:**
  OperasiPerkalian adalah kelas turunan dari OperasiBilanganAbs yang mengimplementasikan metode set_A(), set_B(), dan set_C() untuk operasi perkalian.
    - set_A(double a) menetapkan nilai a.
    - set_B(double b) menetapkan nilai b.
    - set_C() menghitung hasil perkalian antara a dan b.

- tampil() menampilkan hasil perkalian.

5. Kelas OperasiPembagian

- **Deskripsi:**
  OperasiPembagian adalah kelas turunan dari OperasiBilanganAbs yang mengimplementasikan metode set_A(), set_B(), dan set_C() untuk operasi pembagian.
  - set_A(double a) menetapkan nilai a.
  - set_B(double b) menetapkan nilai b.
  - set_C() memeriksa apakah b tidak sama dengan nol, kemudian menghitung hasil pembagian antara a dan b. Jika b adalah nol, program akan mencetak pesan kesalahan dan hasilnya diset ke 0.
  - tampil() menampilkan hasil pembagian.

6. Kelas OperasiBilanganAbsCetak

- **Deskripsi:**
  OperasiBilanganAbsCetak adalah kelas yang bertanggung jawab untuk mencetak hasil operasi.
  - cetakSemua(OperasiBilanganAbs[] OB, double a, double b) menerima array objek dari kelas yang mengimplementasikan OperasiBilanganAbs dan dua nilai angka a dan b. Metode ini akan memanggil set_A(), set_B(), set_C(), dan tampil() pada setiap objek untuk menghitung dan menampilkan hasil operasi.

7. Kelas Main

- **Deskripsi:**
  Di dalam kelas Main, program memulai dengan mendeklarasikan dua variabel angka A dan B dengan nilai 6.5 dan 0.5. Kemudian, array objek dari kelas OperasiPenjumlahan, OperasiPengurangan, OperasiPerkalian, dan OperasiPembagian dibuat untuk melakukan operasi matematika menggunakan polimorfisme.
  - Kelas OperasiBilanganAbsCetak digunakan untuk mencetak hasil operasi melalui metode cetakSemua().
  - Program kemudian memanggil metode ini untuk menghitung dan menampilkan hasil operasi penjumlahan, pengurangan, perkalian, dan pembagian.

Output:

- **HASIL PENJUMLAHAN:**
  Penjumlahan 6.5 + 0.5 menghasilkan 7.0.
- **HASIL PENGURANGAN:**
  Pengurangan 6.5 - 0.5 menghasilkan 6.0.
- **HASIL PERKALIAN:**
  Perkalian 6.5 * 0.5 menghasilkan 3.25.

- **HASIL PEMBAGIAN:**
  Pembagian 6.5 / 0.5 menghasilkan 13.0. (Jika pembagi B adalah nol, maka akan menampilkan pesan kesalahan.)