

LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL III : Prinsip Perancangan Class



Disusun Oleh :

Ahmad Saiful Huda
(19102158)

Dosen Pengampu :

Merlinda Wibowo, S.T., M. Phil.

PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2021

MODUL III

Prinsip Perancangan Class

I. TUJUAN

Mahasiswa diharapkan mampu memahami tentang perancangan kelas yang baik serta menerjemahkannya dalam bahasa pemrograman.

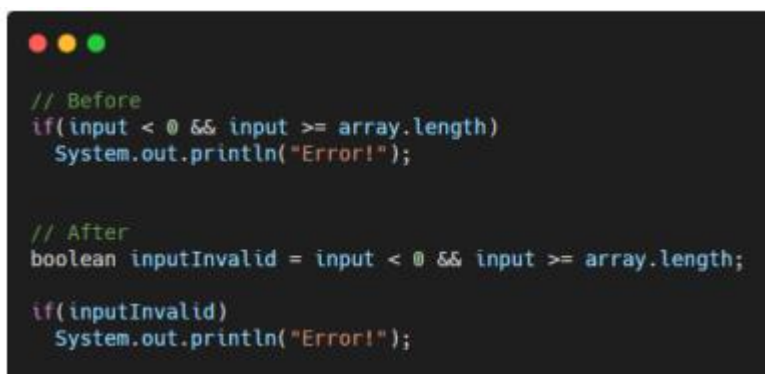
II. DASAR TEORI

Empat prinsip utama dalam Pemrograman Berorientasi Objek:

1. Abstraction

Abstract class adalah class-class yang memiliki informasi abstrak dan metode-metode dari sekumpulan data. Abstract Class tidak bisa diubah dan berlaku juga sebagai kerangka dalam penciptaan subclass-subclassnya (berperan seperti Superclass yang dibahas di konsep Inheritance). Sebuah Abstract Class memiliki informasi dan metode yang dapat diturunkan ke subclassnya, dan seluruh subclass akan mengikuti apa saja metode yang akan diturunkan oleh Abstract Class.

Abstraksi berarti menggunakan hal-hal sederhana untuk mewakili hal-hal rumit. Perhatikan contoh sederhana berikut:



```
// Before
if(input < 0 && input >= array.length)
    System.out.println("Error!");

// After
boolean inputInvalid = input < 0 && input >= array.length;

if(inputInvalid)
    System.out.println("Error!");
```

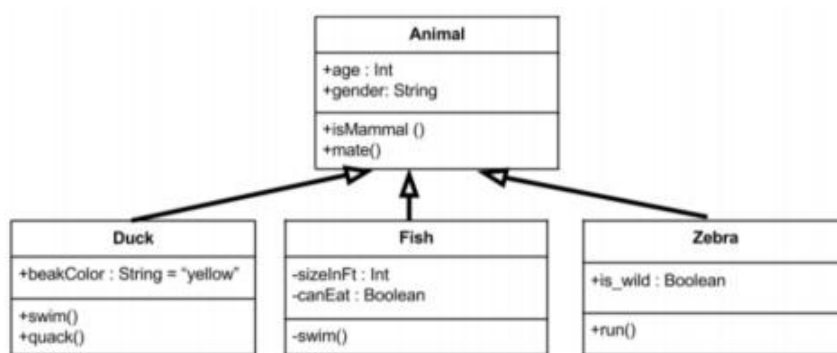
Pada contoh di atas, pada if pertama, orang lain yang belum paham tentang kode kita mungkin akan bertanya-tanya kondisi apa yang sedang dicek di if tersebut. Sedangkan pada if kedua, jika kita menunjukkan kode kita pada orang awam pun, orang itu akan paham bahwa kode tersebut sedang mengecek apakah input yang diterima program valid atau invalid.

2. Encapsulation

Enkapsulasi berarti menyembunyikan data-data kompleks di balik modifier Private dan menyediakan sedikit method Public sebagai “jalan” untuk mengakses data-data Private tersebut. Encapsulation atau pengkapsulan adalah konsep tentang pengikatan data atau metode yang berbeda yang disatukan atau “dikapsulkan” menjadi satu unit data. Encapsulation dapat mempermudah dalam pembacaan code karena informasi yang disajikan tidak perlu dibaca secara rinci dan sudah merupakan satu kesatuan.

3. Inheritance

Inheritance (arti: Pewarisan) merupakan konsep menurunkan atribut dan method milik sebuah class ke class lain. Perhatikan contoh class diagram berikut:



Pada contoh di atas, class **Animal** disebut Parent class atau Superclass, sedangkan class **Duck**, **Fish**, dan **Zebra** disebut Child class atau Subclass. Ketiga Child class tersebut akan memiliki SEMUA atribut dan method milik Parent class. Dalam merancang class, jika kita menemukan ada beberapa class dengan atribut dan method yang konteksnya sama, lebih baik menggunakan konsep Pewarisan saja. Salah satu keuntungannya adalah jika kita sewaktu-waktu ingin mengubah kode kita, kita hanya perlu mengubah kode kita satu kali saja, sedangkan jika kita menuliskan class berbeda tanpa ada hubungan Inheritance, jika kita ingin melakukan perubahan kita harus merubah isi dari semua class tersebut satu persatu.

4. Polymorphism

Polimorfisme merupakan konsep pembuatan method berbeda dengan nama yang sama. Polimorfisme dibagi menjadi dua, yaitu:

- a) Overloading Overloading adalah ketika di sebuah class ada 2 atau lebih method dengan nama sama namun masing-masing memiliki parameter berbeda. Walau namanya sama, program tetap mengerti harus menjalankan method yang mana berdasarkan parameter yang dimasukan saat pemanggilan method.

Contoh sederhana method overloading:

```
class Hero {  
    // Akan dieksekusi jika tipe data parameter adalah Armor  
    public void equip(Armor armor) {}  
  
    // Akan dieksekusi jika tipe data parameter adalah Weapon  
    public void equip(Weapon weapon) {}  
}
```

- b) Overriding

Overriding adalah ketika sebuah Child class memiliki method dengan nama yang sama dengan Parent class-nya, namun dengan isi yang berbeda (parameter boleh sama). Nantinya program akan memprioritaskan method milik Child class ini jika ternyata ketahuan Child class memiliki method yang di-Override dari Parent-nya. Contoh sederhana method overriding:

```
class Burung { // Parent class  
    /*  
        Jika method ini dipanggil dari object Burung, method  
        yang ini yang akan dieksekusi ketika terbang() dipanggil  
    */  
    public void terbang() {  
        System.out.println("Aku terbang!");  
    }  
}  
  
class Penguin extends Burung { // Child class  
    /*  
        Jika method ini dipanggil dari object Penguin, method  
        yang ini yang akan dieksekusi ketika terbang() dipanggil  
    */  
    public void terbang() {  
        System.out.println("Aku gak bisa terbang :(");  
    }  
}
```

(Inheritance dan Polymorphism akan dibahas lebih lanjut di modul selanjutnya)

Beberapa prinsip-prinsip perancangan kelas yang perlu diperhatikan, antara lain:

1. Constructor. Method ini digunakan untuk inialisasi atau mempersiapkan data untuk objek.
2. Visibilitas Bagi Atribut dan Method. Diperlukan kontrol akses untuk mengatur siapa saja yang dapat mengakses atau mengubah nilai dari atribut atau method dalam kelas tersebut sehingga penyalahgunaan atribut atau method dapat dihindari. Kontrol akses yang sering digunakan dalam perancangan suatu kelas sudah ada pada modul minggu lalu.
3. Fungsi Accessor dan Mutator. Fungsi accessor merupakan fungsi untuk mendapatkan properti dari suatu objek dan mengembalikan nilai atau value dari suatu atribut (get). Sedangkan fungsi mutator mengubah properti dari suatu objek dan mengubah nilai atau value dari sebuah atribut (set).
4. Method dan Operator Overloading. Method-method tersebut harus dapat dibedakan antara satu dengan yang lain dalam jumlah dan atau tipe datanya.
5. Melewatkan Argumen/Parameter ke Method. Cara melewati argumen/parameter ini dapat dilakukan dengan melewati secara nilai (pass by value) dan melewati secara referensi (pass by reference).



```
// VALUE
int x = 7;
int y = x;

System.out.println(x); // Output: 7
System.out.println(y); // Output: 7

y = 9;

System.out.println(x); // Output: 7
System.out.println(y); // Output: 9

// REFERENCE
int[] x = {2, 3, 5, 7};
int[] y = x;

for(int e : x)
    System.out.print(e + " "); // Output: 2, 3, 5, 7
System.out.println();

for(int e : y)
    System.out.print(e + " "); // Output: 2, 3, 5, 7
System.out.println();

y[0] = 1;

for(int e : x)
    System.out.print(e + " "); // Output: 1, 3, 5, 7
System.out.println();

for(int e : y)
    System.out.print(e + " "); // Output: 1, 3, 5, 7
System.out.println();
```

Pass by value diterapkan pada argumen bertipe data primitif dan prosesnya hanya menyalin isi memori (yang telah dialokasikan untuk suatu variabel), dan kemudian menyampaikan salinan tersebut kepada method yang bersangkutan (isi memory merupakan data sesungguhnya yang akan dioperasikan), karena yang disampaikan hanya salinan dari isi memori, maka perubahan yang terjadi pada variabel akibat proses didalam method tidak mempengaruhi nilai variabel asalnya di dalam memori. Sedangkan, pass by reference diterapkan pada argumen bertipe data array atau objek dan isi memori pada variabel array atau objek merupakan penunjuk alamat memori yang mengandung data sesungguhnya yang akan dioperasikan.

6. Responsibility Driven Design. Semua fungsi dan method yang ada harus mencerminkan perilaku lengkap yang dimiliki kelas tersebut. Semua fungsi/method tersebut bertanggung jawab terhadap maintenance atribut yang dimiliki kelas.

III. GUIDED

Buatlah package baru dengan format com.huda.praktikumpbo.guided3, lalu ikuti langkah langkah di bawah:

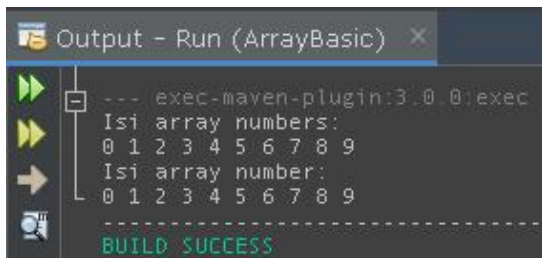
DASAR ARRAY Untuk membuat array kosong di Java, pertama kita tuliskan dulu tipe data yang nantinya akan ditampung array yang kita buat. Setelah itu, jangan lupa berikan kurung siku terbuka dan tertutup []. Setelah itu, tulis nama yang akan kita gunakan untuk mereferensikan array tersebut. Terakhir, kita harus menuliskan tipe data array tersebut lagi dan kali ini di dalam kurung siku kita tuliskan jumlah elemen (isi) maksimum yang dapat ditampung oleh array tersebut. Buatlah sebuah package baru di dalam package guided dan beri nama array. Buat class baru di dalam package array tersebut dan beri nama ArrayBasic.

Source Code:

```
6 package com.huda.praktikumpbo.guided3;
7
8 /**
9  *
10  * @author m0n
11  */
12 public class ArrayBasic {
13     public static void main(String[] args) {
14         //tipeData[] namaArray = new tipeData[pjgArray]
15         int[] numbers = new int [10];
16
17         //isi Array 0 sampai 9
18         for (int i=0; i<10; i++)
19             numbers[i] = i;
20
21         //Output seluruh isi array (Cara 1)
22         System.out.println("Isi array numbers: ");
23         for (int j=0; j<numbers.length; j++)
24             System.out.print(numbers[j] + " ");
25         System.out.println();
26
27         //Output seluruh isi array (Cara 2)
28         System.out.println("Isi array number:");
29         for (int number : numbers)
30             System.out.print(number + " ");
31         System.out.println();
32     }
33 }
34
35
36
```

Pada program di atas, array kita beri nama numbers. dan array numbers ini dapat menampung sejumlah 10 bilangan bulat. Untuk mencetak isi array, kita bisa memanfaatkan for loop. Ada 2 variasi for loop,

Hasil running:



ARRAY DENGAN NILAI YANG SUDAH DITENTUKAN DARI AWAL Untuk membuat sebuah array yang sudah langsung memiliki nilai sejak ia dibuat, pertama tuliskan tipe datanya terlebih dahulu diikuti dengan 2 kurung siku []. Setelah itu jangan lupa tulis nama array-nya. Lalu, tulis = {}. Nah, untuk memasukan sesuatu ke array ini, anda tinggal menyetikan saja apa yang anda masukan ke array tersebut ke dalam {} tadi. Gunakan tanda koma (,) sebagai pemisah antara elemen/nilai index satu dengan index lainnya.

Di package yang sama, buatlah class baru lagi dan beri nama ArrayNotSoBasic,

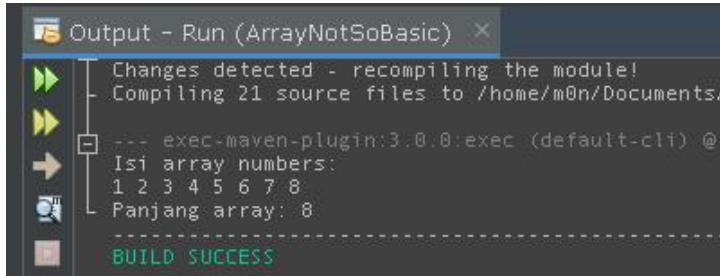
Source Code:

```
6 package com.huda.praktikumpbo.guided3;
7
8 /**
9  *
10  * @author mon
11  */
12 public class ArrayNotSoBasic {
13     public static void main(String[] args) {
14         int[] numbers = {1, 2, 3, 4, 5, 6, 7, 8};
15
16         System.out.println("Isi array numbers: ");
17         for (int number : numbers) {
18             System.out.print(number + " ");
19             System.out.print("\n");
20         }
21         System.out.println("");
22         System.out.println("Panjang array: " + numbers.length);
23     }
24 }
25 }
```

Pada program di atas, kita membuat array dengan nama numbers yang menampung bilangan bulat 1, 2, 3, 4, 5, 6, 7, 8. Jika anda membuat array dengan cara seperti ini, anda tidak perlu menuliskan berapa panjang array-nya, karena panjang array akan menyesuaikan, tergantung anda memasukan berapa nilai atau object ke dalam array tersebut saat anda membuatnya. Untuk mengecek panjang array, anda bisa

menuliskan nama array tersebut, lalu menambahkan `.length` setelahnya. Pada contoh di atas, karena nama array-nya adalah `numbers`, jadi kita menuliskan `numbers.length`

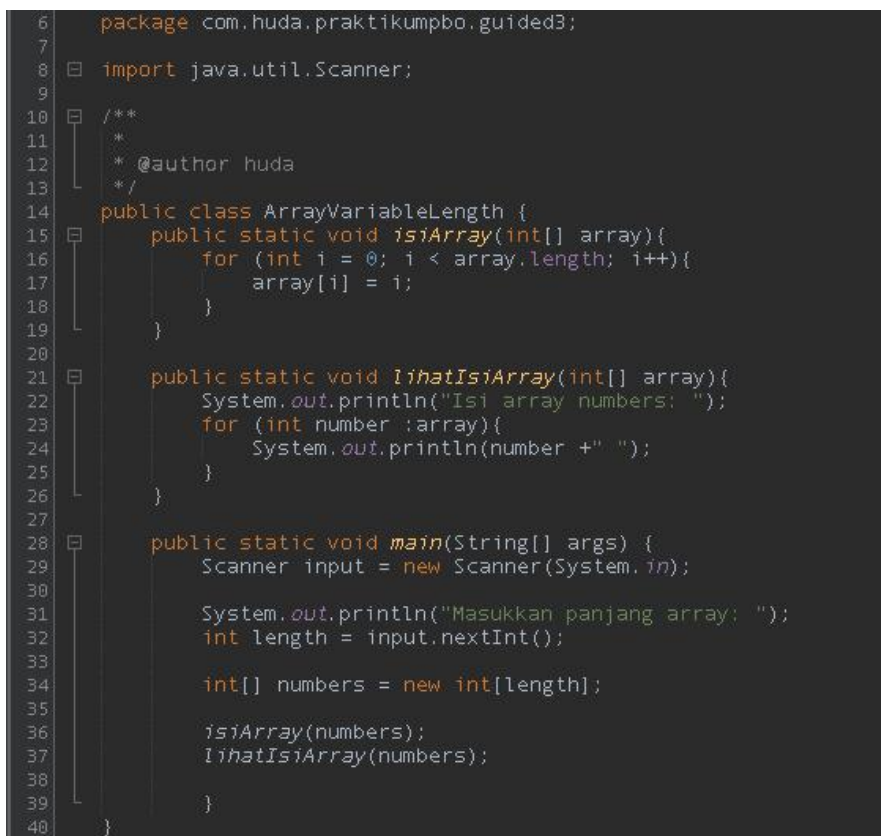
Hasil Running :



```
Output - Run (ArrayNotSoBasic) x
Changes detected - recompiling the module!
Compiling 21 source files to /home/m0n/Documents/...
--- exec-maven-plugin:3.0.0:exec (default-cli) @
Isi array numbers:
1 2 3 4 5 6 7 8
Panjang array: 8
-----
BUILD SUCCESS
```

ARRAY KOSONG DENGAN PANJANG BERVARIASI Selain 2 cara di atas, ada cara lain lagi yang bisa digunakan untuk membuat array. Pada cara ini, panjang array akan ditentukan oleh variable, yang mana berarti kita bisa membuat program yang memiliki array dengan panjang yang berbeda tergantung inputan user! Masih di package yang sama, buatlah class baru dan beri nama `ArrayVariableLength`.

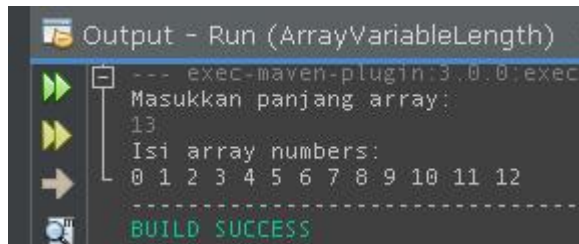
Source Code:



```
6 package com.huda.praktikumpbo.guided3;
7
8 import java.util.Scanner;
9
10 /**
11  *
12  * @author huda
13  */
14 public class ArrayVariableLength {
15     public static void isiArray(int[] array){
16         for (int i = 0; i < array.length; i++){
17             array[i] = i;
18         }
19     }
20
21     public static void lihatIsiArray(int[] array){
22         System.out.println("Isi array numbers: ");
23         for (int number : array){
24             System.out.println(number + " ");
25         }
26     }
27
28     public static void main(String[] args) {
29         Scanner input = new Scanner(System.in);
30
31         System.out.println("Masukkan panjang array: ");
32         int length = input.nextInt();
33
34         int[] numbers = new int[length];
35
36         isiArray(numbers);
37         lihatIsiArray(numbers);
38     }
39 }
40
```

Prinsipnya sama seperti metode pembuatan array pertama, hanya saja kita memasukan sebuah variable ke dalam [], bukan angka.

Hasil running:



```
Output - Run (ArrayVariableLength)
--- exec-maven-plugin:3.0.0:exec
Masukkan panjang array:
13
Isi array numbers:
0 1 2 3 4 5 6 7 8 9 10 11 12
-----
BUILD SUCCESS
```

IV. UNGUIDED

Seperti biasa, buat package baru di dalam package pertemuan3 dan beri nama unguided. Anda berada di jalan yang benar jika package anda terlihat seperti ini: com.nama.praktikumpbo.pertemuan3.unguided

Buatlah sebuah program sederhana dengan tema bebas yang melibatkan pembuatan class dan object! Syarat dan ketentuan:

- Minimal 2 file, satu untuk class object dan satu untuk class main.
- Sertakan source code dan screenshot output saat running program di laporan praktikum.
- Jangan lupa beri penjelasan tentang alur kerja program di laporan.
- Nilai plus jika program memiliki menu.
- Nilai plus plus jika source code rapi, enak dilihat, dan mudah dipahami saat pertama kali dibaca.

Hasil pengerjaan:

Class Objek

Class dengan nama ConvertSuhu pada package com.huda.praktikumpbo.pertemuan3, Terdapat method yang mendeklarasikan serta melakukan operasi aritmatika sesuai rumus untuk konversi suhu.

```
package com.huda.praktikumpbo.pertemuan3;

/**
 *
 * @author huda
 */
public class ConvertSuhu {
    //typeData
    private double celcius, fahrenheit, kelvin, reamur, suhuAwal;

    //Constructor
    public ConvertSuhu(double suhuAwal){
        this.suhuAwal = suhuAwal;
    }

    //Definisi method
    public double getSuhuAwal() {
        return this.suhuAwal;
    }
}
```

```

//method
void Celcius(double suhuAwal){
    reamur = ((4*suhuAwal)/5);
    fahrenheit = ((9*suhuAwal)/5)+32;
    kelvin = suhuAwal+273.15;
    System.out.println("Suhu dalam Reamur\t : "+ reamur + "°R");
    System.out.println("Suhu dalam Fahrenheit\t : "+fahrenheit+ "°F");
    System.out.println("Suhu dalam Kelvin\t : "+ kelvin + "°K");
    System.out.println("");
}

void Fahrenheit(double suhuAwal){
    celcius = ((suhuAwal-32)*5)/9;
    reamur = ((suhuAwal-32)*4)/9;
    kelvin = (((suhuAwal-32)*5)/9)+273.15;
    System.out.println("Suhu dalam Celcius\t : "+celcius + "°C");
    System.out.println("Suhu dalam Reamur\t : "+ reamur + "°R");
    System.out.println("Suhu dalam Kelvin\t : "+ kelvin + "°K");
    System.out.println("");
}

void Reamur(double suhuAwal){
    celcius = (5*suhuAwal)/4;
    fahrenheit = ((9*suhuAwal)/4) + 32;
    kelvin = ((5*suhuAwal)/4)+273.15;
    System.out.println("Suhu dalam Celcius\t : "+celcius + "°C");
    System.out.println("Suhu dalam Fahrenheit\t : "+fahrenheit + "°F");
    System.out.println("Suhu dalam Kelvin\t : "+ kelvin + "°K");
    System.out.println("");
}

void Kelvin(double suhuAwal){
    celcius = suhuAwal - 273.15;
    reamur = ((suhuAwal - 273.15) * 4)/5;
    fahrenheit = (((suhuAwal - 273.15) * 9)/5) + 32;
    System.out.println("Suhu dalam Celcius\t : "+celcius+ "°C");
    System.out.println("Suhu dalam Fahrenheit\t : "+fahrenheit + "°F");
    System.out.println("Suhu dalam Reamur\t : "+ reamur + "°R");
    System.out.println("");
}
}

```

Class Main

Pada program MainClass menggunakan perulangan do/while untuk menampilkan menu, program ini juga menggunakan switch case sebagai menu.

```
package com.huda.praktikumpbo.pertemuan3;

import java.util.Scanner;

/**
 *
 * @author huda
 */
public class MainClass {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        //typeData
        float suhuAwal;

        boolean keepLooping = true;
        do {
            System.out.println("-----");
            System.out.println("|\\tMenu Utama\\t| ");
            System.out.println("-----");
            System.out.println("| 1) Celcius\\t\\t|");
            System.out.println("| 2) Fahrenheit\\t\\t|");
            System.out.println("| 3) Reamur\\t\\t|");
            System.out.println("| 4) Kelvin\\t\\t|");
            System.out.println("| 5) Exit\\t\\t|");
            System.out.println("-----");
            System.out.println("Masukan suhu yang ingin anda konversi: ");

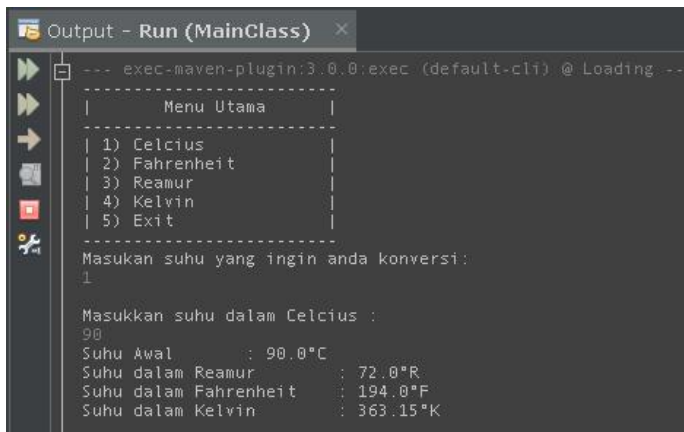
            int menu = input.nextInt();
            System.out.println();
        } while (keepLooping);
    }
}
```

```

switch (menu) {
    case 1 -> {
        //konversi dari suhu Celcius
        System.out.println("Masukkan suhu dalam Celcius : ");
        suhuAwal = input.nextFloat();
        ConvertSuhu c = new ConvertSuhu(suhuAwal);
        System.out.println("Suhu Awal\t: " + c.getSuhuAwal()+"°C");
        c.Celcius(suhuAwal);
    }
    case 2 -> {
        //konversi dari suhu Fahrenheit
        System.out.println("Masukkan suhu dalam Fahrenheit: ");
        suhuAwal = input.nextFloat();
        ConvertSuhu f = new ConvertSuhu(suhuAwal);
        System.out.println("Suhu Awal\t: " + f.getSuhuAwal() + "°F");
        f.Fahrenheit(suhuAwal);
    }
    case 3 -> {
        //konversi dari suhu Reamur
        System.out.println("Masukkan suhu dalam Reamur: ");
        suhuAwal = input.nextFloat();
        ConvertSuhu r = new ConvertSuhu(suhuAwal);
        System.out.println("Suhu Awal\t: " + r.getSuhuAwal() + "°R");
        r.Reamur(suhuAwal);
    }
    case 4 -> {
        //konversi dari suhu Kelvin
        System.out.println("Masukkan suhu dalam Kelvin: ");
        suhuAwal = input.nextFloat();
        ConvertSuhu k = new ConvertSuhu(suhuAwal);
        System.out.println("Suhu Awal\t: " + k.getSuhuAwal() + "°K");
        k.Kelvin(suhuAwal);
    }
    case 5 -> {
        //exit program
        System.out.println("Telah keluar dari program");
        keepLooping = false;
    }
    default -> {
        // jika user menginput tidak sesuai pada menu
        System.out.println("Menu yang anda masukan salah!");
        System.out.println();
    }
}
} while (keepLooping);
}
}

```

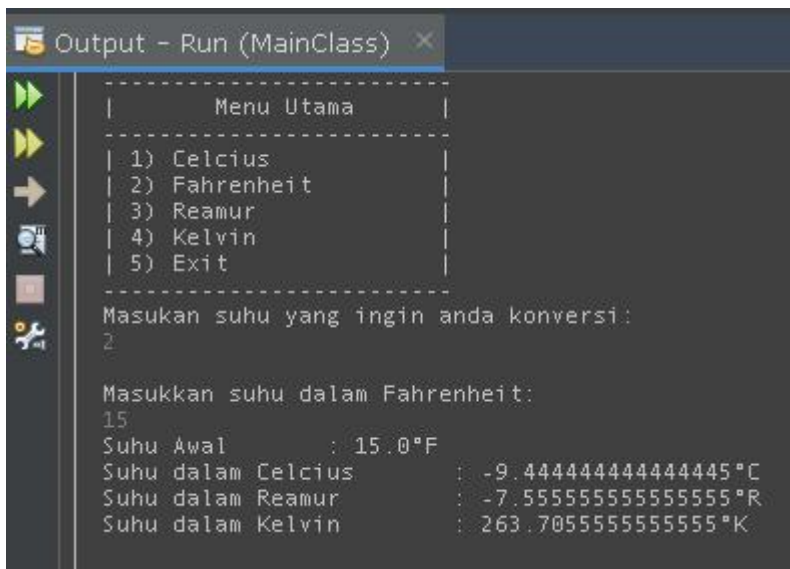
Hasil running program:



```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Loading ---
|-----|
| Menu Utama |
|-----|
| 1) Celcius |
| 2) Fahrenheit |
| 3) Reamur |
| 4) Kelvin |
| 5) Exit |
|-----|
Masukan suhu yang ingin anda konversi:
1

Masukkan suhu dalam Celcius :
90
Suhu Awal      : 90.0°C
Suhu dalam Reamur    : 72.0°R
Suhu dalam Fahrenheit : 194.0°F
Suhu dalam Kelvin    : 363.15°K
```

Jika user memilih atau memasukan menu 1, program akan menghitung suhu dari Celcius.



```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Loading ---
|-----|
| Menu Utama |
|-----|
| 1) Celcius |
| 2) Fahrenheit |
| 3) Reamur |
| 4) Kelvin |
| 5) Exit |
|-----|
Masukan suhu yang ingin anda konversi:
2

Masukkan suhu dalam Fahrenheit:
15
Suhu Awal      : 15.0°F
Suhu dalam Celcius    : -9.444444444444445°C
Suhu dalam Reamur     : -7.555555555555555°R
Suhu dalam Kelvin     : 263.7055555555555°K
```

Jika user memilih atau memasukan menu 2, program akan menghitung suhu dari Fahrenheit.


```
Output - Run (MainClass) X
|-----|
|      Menu Utama      |
|-----|
| 1) Celcius           |
| 2) Fahrenheit        |
| 3) Reamur            |
| 4) Kelvin            |
| 5) Exit              |
|-----|
Masukan suhu yang ingin anda konversi:
3

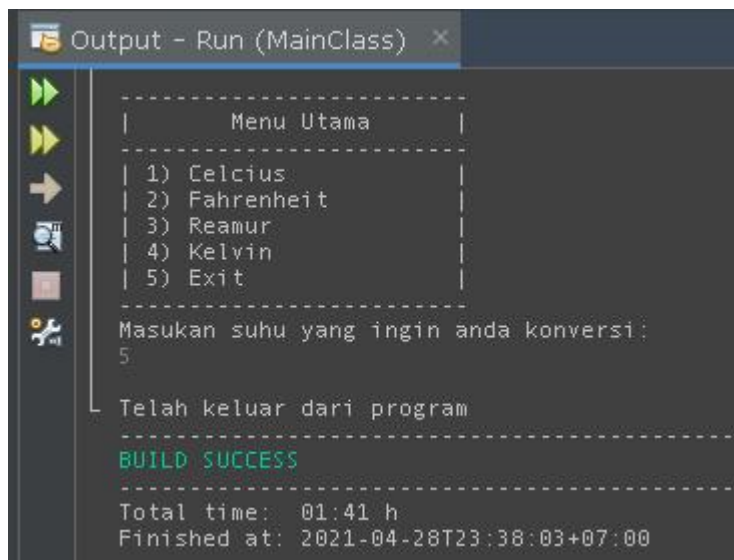
Masukkan suhu dalam Reamur:
28
Suhu Awal      : 28.0°R
Suhu dalam Celcius : 35.0°C
Suhu dalam Fahrenheit : 95.0°F
Suhu dalam Kelvin : 308.15°K
```

Jika user memilih atau memasukkan menu 3, program akan menghitung suhu dari Reamur.

```
Output - Run (MainClass) X
|-----|
|      Menu Utama      |
|-----|
| 1) Celcius           |
| 2) Fahrenheit        |
| 3) Reamur            |
| 4) Kelvin            |
| 5) Exit              |
|-----|
Masukan suhu yang ingin anda konversi:
4

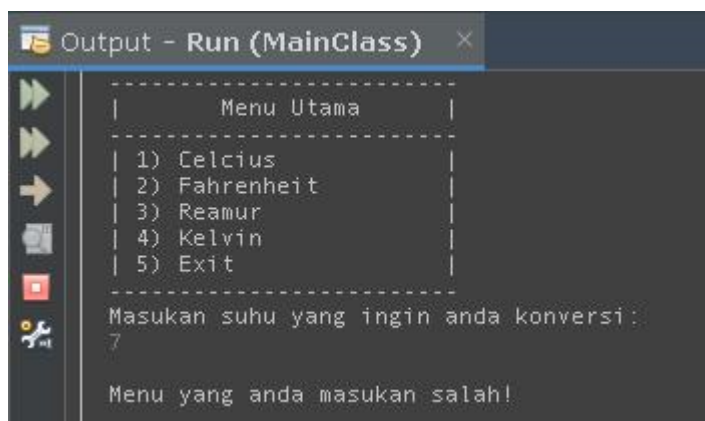
Masukkan suhu dalam Kelvin:
38
Suhu Awal      : 38.0°K
Suhu dalam Celcius : -235.14999999999998°C
Suhu dalam Fahrenheit : -391.27°F
Suhu dalam Reamur : -188.11999999999998°R
```

Jika user memilih atau memasukkan menu 4, program akan menghitung suhu dari Kelvin.



```
Output - Run (MainClass) x
|-----|
|      Menu Utama      |
|-----|
| 1) Celcius           |
| 2) Fahrenheit        |
| 3) Reamur            |
| 4) Kelvin            |
| 5) Exit              |
|-----|
Masukan suhu yang ingin anda konversi:
5
Telah keluar dari program
BUILD SUCCESS
Total time: 01:41 h
Finished at: 2021-04-28T23:38:03+07:00
```

Jika user memilih atau memasukan menu 5, program akan terhenti dan akan keluar dari program.



```
Output - Run (MainClass) x
|-----|
|      Menu Utama      |
|-----|
| 1) Celcius           |
| 2) Fahrenheit        |
| 3) Reamur            |
| 4) Kelvin            |
| 5) Exit              |
|-----|
Masukan suhu yang ingin anda konversi:
7
Menu yang anda masukan salah!
```

Jika user memilih atau memasukan tidak sesuai pada menu, program akan menampilkan pesan bahwa manu yang dimasukan salah dan akan kembali pada menu.

V. KESIMPULAN

Dalam mengimplementasikan sebuah program dibutuhkan perancangan kelas yang baik sehingga kelak dihasilkan kode program yang berkualitas baik dan mudah untuk dikembangkan. Hal ini perlu diperhatikan mengingat bahwa perubahan program karena adanya perubahan kebutuhan sangat sering terjadi sehingga dibutuhkan pengetahuan tentang prinsip perancangan yang baik untuk menghasilkan program yang baik tersebut.