**Jobsheet 7**

**Interface**

# A. Kompetensi

Setelah menyelesaikan lembar kerja ini mahasiswa diharapkan mampu:

1. Menjelaskan maksud dan tujuan penggunaan interface;
2. Menerapkan interface di dalam pembuatan program.

# B. Pendahuluan

*Interface* merupakan sekumpulan abstract method yang saling berkaitan

1. **Karakteristik:** 
   1. Umumnya terdiri dari abstract method
   2. Selalu dideklarasikan dengan menggunakan kata kunci interface.
   3. Diimplementasikan dengan menggunakan kata kunci implements
   4. Interface tidak dapat diinstansiasi, hanya dapat diinstasiasi melalui class yang mengimplement interface tersebut
2. **Kegunaan:**

Bertindak sebagai **kontrak/syarat** yang berisi **sekumpulan behavior/method** yang saling terkait untuk memenuhi suatu **kapabilitas**. Dengan kata lain, interface memberikan panduan mengenai method apa saja yang perlu diimplementasikan untuk memenuhi kapabilitas tertentu.

# 3. Notasi Class Diagram Interface

* Nama interface **tidak** dicetak miring
* Keterangan <<interface>> di atas nama interface
* Nama method boleh dicetak miring atau tidak
* Implements dilambangkan dengan garis panah putus-putus

# 4. Sintaks Interface

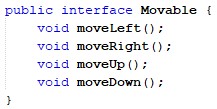
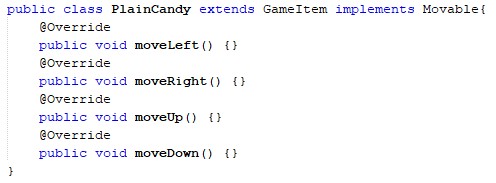
* Untuk mendeklarasikan suatu interface:

public interface <NamaInterface>

* Untuk mengimplementasikan interface:

public class <NamaClass> implements <NamaInterface>

* Nama interface sebaiknya dalam bentuk **adjective/kata sifat** jika merepresentasikan kapabilitas. Dapat juga menggunakan **kata benda**
* Contoh:



# 5. Implementasi Interface

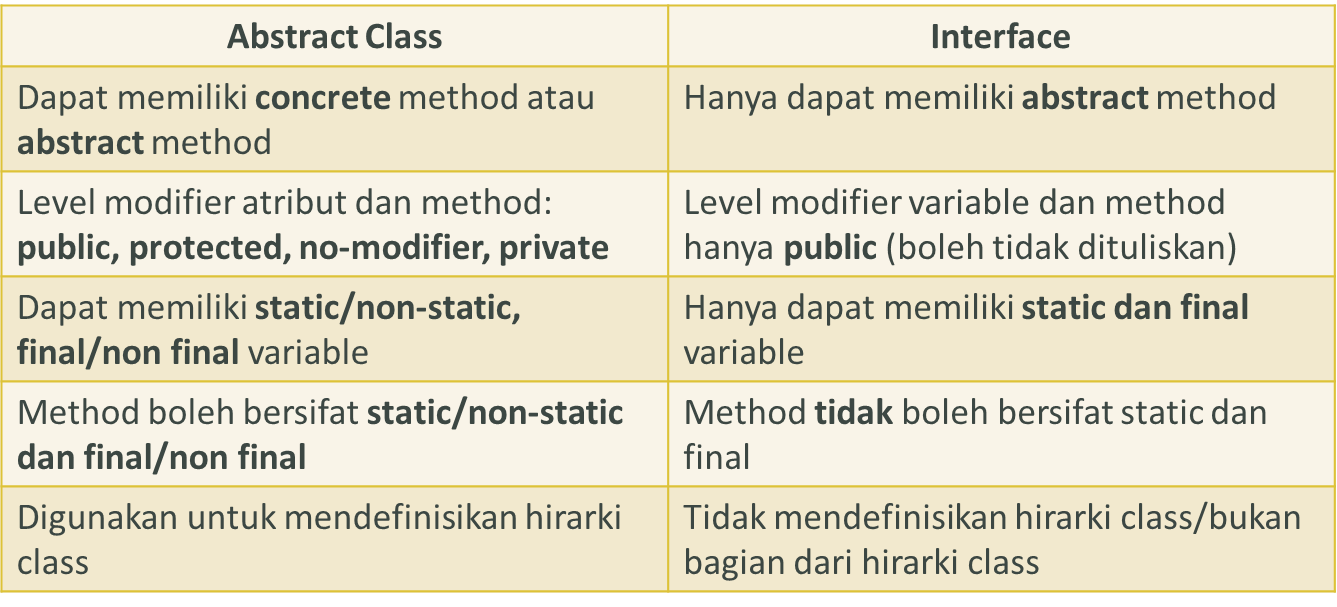
Bila sebuah class mengimplementasikan suatu interface:

* **Seluruh konstanta** dari interface akan dimiliki oleh class tersebut
* **Seluruh method** pada interface harus diimplementasikan
* Bila class yang meng-implement interface **tidak mengimplementasikan semua method**, maka class tersebut harus dideklarasikan sebagai **abstract class**

# 6. Multiple Interface

* Suatu class dapat meng-implement multiple interface
* Bila suatu class merupakan subclass dan meng-implement interface, maka **keyword extends mendahului implements** ▪ Contoh: public class PlainCandy extends GameItem implements Crushable, Movable

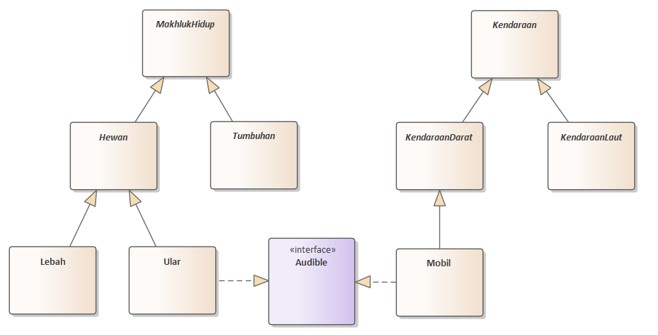
# 7. Perbedaan Abstract Class dan Interface



# 8. Interface tidak terikat pada hirarki

Suatu class di java hanya dapat meng-extend atau menjadi subclass secara langsung dari **satu** superclass saja. Akibatnya class tersebut akan terikat pada suatu hirarki tertentu. Misalnya class Lebah merupakan subclass Hewan sedangkan class Hewan sendiri merupakan subclass MakhlukHidup. Pembatasan 1 parent class secara langsung ini menyebabkan class Lebah terikat pada hirarki makhluk hidup dan tidak bisa terkait dengan hirarki lainnya.

Sementara itu interface tidak terikat pada suatu hirarki. Interface dibuat “secara lepas” tanpa bergantung pada hirarki. Misalkan terdapat interface Audible, interface tersebut dapat diimplementasikan di class apapun dari hirarki manapun. Misal class Ular bisa bersuara, class ini dapat mengimplementasikan interface Audible. Begitu juga dengan class Mobil dari hirarki kendaraan dapat pula mengimplementasikan interface Audible.



# 9. Penggunaan Abstract Class vs Interface

Abstract class dapat memiliki atribut (instance variable), yaitu suatu variable yang dimiliki oleh objek tertentu. Atribut dan method ini (jika access level modifier-nya sesuai) akan diwariskan terhadap subclass nya. Oleh karena itu, jika suatu class memiliki **common properties (dan method)** maka sebaiknya dibuat abstract class sebagai generalisasi. Misal

ada beberapa class PlainCandy, StripedCandy, RainbowChocoCandy, Wall dll yang merupakan jenis item dalam game dengan atribut yang sama, misalnya positionX, positionY, dan iconName, sebaiknya kita buat abstract class

GameItem sebagai generalisasi dari class-class tersebut.

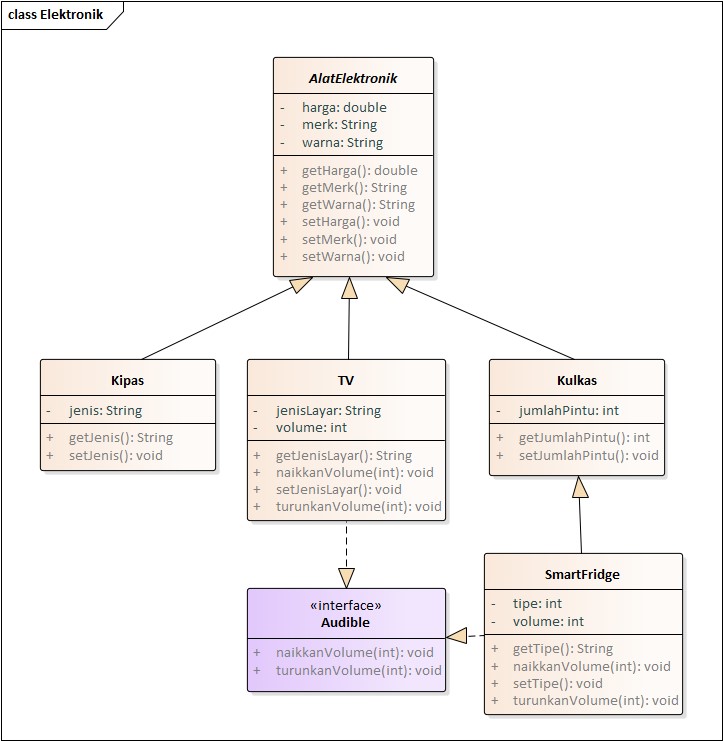
Sementara itu, jika beberapa class memiliki **common behavior** (perilaku atau kapabilitas yang sama) kita bisa menggunakan interface untuk memberikan panduan mengenai method apa saja yang perlu diimplementasikan untuk memenuhi kapabilitas tertentu. Misalnya jika suatu class memiliki kapabilitas untuk dapat berpindah atau

Movable, seharusnya dia memiliki method moveLeft(), moveRight(),

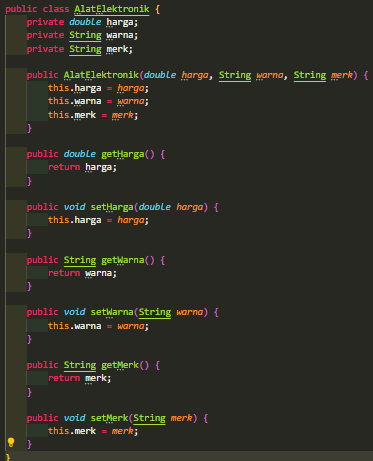
moveDown, moveUp. Sekumpulan method dalam interface ini akan menjadi panduan atau pedoman, bahwa jika selanjutnya ada pengembangan atau penambahan game item lain dan item tersebut dapat bergerak juga maka method-method tersebut harus diimplementasikan dalam class nya.

# C. PERCOBAAN

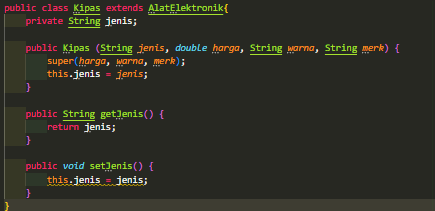
Implementasikan class diagram berikut ke dalam kode program.

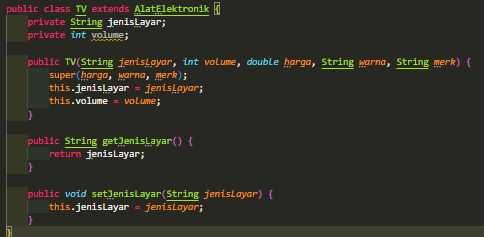


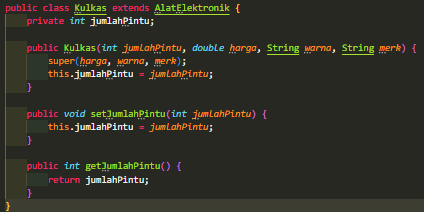
1. Buat project baru dengan nama InterfaceLatihan (boleh disesuaikan)
2. Pada sebuah package, buatlah abstract class AlatElektronik



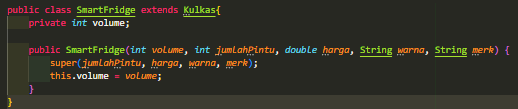
1. Selanjutnya buatlah subclass dari AlatElektronik, yaitu Kipas, TV, dan Kulkas sebagai berikut.



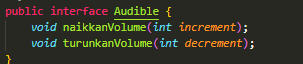




1. Buatlah class SmartFridge yang merupakan sublass dari class Kulkas



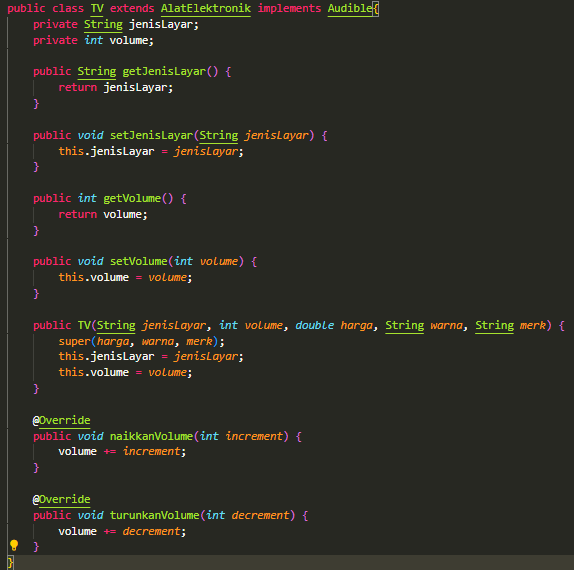
1. Beberapa dari alat elektronik dapat mengeluarkan suara. Kapabilitas ini kita buat ke dalam kode program dengan interface Audible dengan method naikkanVolume() dan turunkanVolume() sebagai berikut



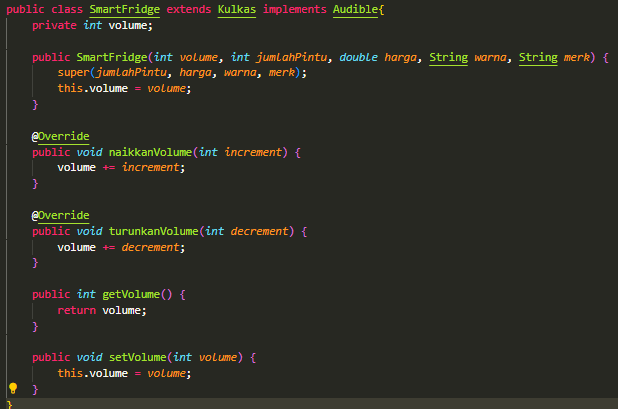
1. Ubah class TV untuk meng-impelement interface Audible



1. Implementasi abstract method pada interface Audible pada class TV



1. Lakukan hal yang sama pada class SmartFridge



# D. PERTANYAAN 2

1. Mengapa terjadi error pada langkah 5?

Karena kelas TV yang dibuat harus mengimplementasikan metode naikkanVolume(int) yang berasal dari interface Audible.

1. Mengapa Audible tidak dapat dibuat sebagai class?

Interface seperti Audible digunakan untuk mendefinisikan kontrak yang harus diikuti oleh kelas-kelas lain. Jika Audible diubah menjadi kelas, maka TV harus mewarisi kelas Audible, yang akan membatasi fleksibilitas.

1. Jika access level modifier interface Audible tidak dituliskan, apa access level modifier defaultnya?

**Default access level-nya adalah** public jika interface tersebut berada di dalam package yang sama, atau package-private (tanpa modifier) jika berada di luar package.

1. Access level modifier method-method dalam interface Audible tidak dituliskan, apa access level modifier sebenarnya?

**Dalam java, semua method dalam interface secara default adalah** public

1. Method naikkanVolume() dan turunkanVolume() memilki implementasi yang sama pada TV dan SmartFridge(), mengapa tidak langsung diimplementasikan pada interface Audible()?

Dikarenakan ada kode implementasi yang perlu di-share antar kelas tetapi membutuhkan akses ke data kelas tertentu

1. Method naikkanVolume() dan turunkanVolume() memilki implementasi yang sama pada TV dan SmartFridge(), mengapa tidak langsung diimplementasikan pada class AlatElektronik?

Dikarenakan tidak semua jenis AlatElektronik memerlukan volume

1. Apakah method naikkanVolume() dan turunkanVolume() pada class TV dan SmartFridge() dapat memiliki implementasi yang berbeda?

Ya, metode naikkanVolume() dan turunkanVolume() pada kelas TV dan SmartFridge() dapat memiliki implementasi yang berbeda.

1. Semua yang Audible seharusnya memiliki nilai volume, mengapa atribut volume tidak dideklarasikan dalam interface Audible()?

**Atribut** volume **tidak dideklarasikan dalam interface** Audible karena **interface** hanya mendefinisikan **perilaku** yang diharapkan, bukan data atau atribut.

1. Ubah implementasi method naikkanVolume() dan turunkanVolume() pada class TV sebagai berikut:



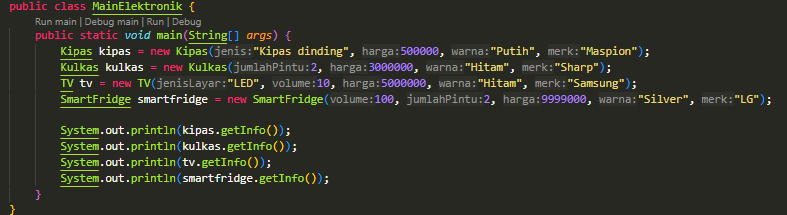
Compile dan run program. Apakah terjadi error? Mengapa?

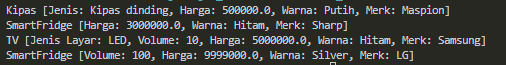
Kelas TV mencoba mengimplementasikan metode naikkanVolume() yang tidak sesuai dengan metode yang ada di **interface** Audible

1. Kembalikan method naikkanVolume() dan turunkanVolume() pada class TV seperti semula
2. Apa fungsi dari interface?

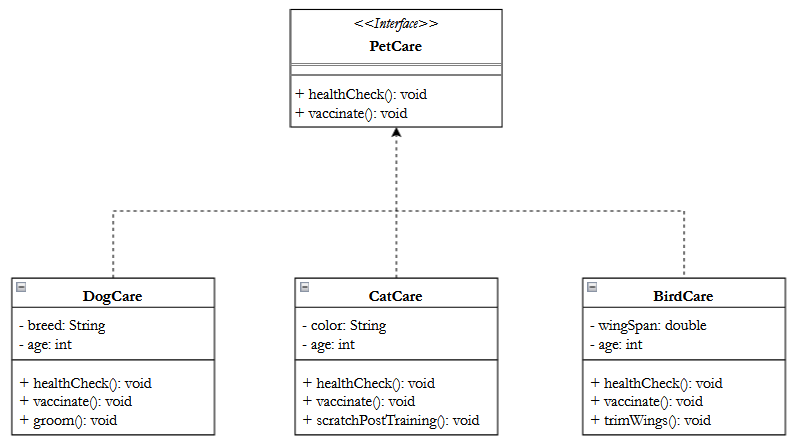
Untuk mendefinisikan **kontrak** atau **perilaku** yang harus diikuti oleh kelas yang mengimplementasikannya.

1. Buat method getInfo() untuk setiap class. Instansiasi objek dari setiap concrete class pada main class, kemudian tampilkan infonya.





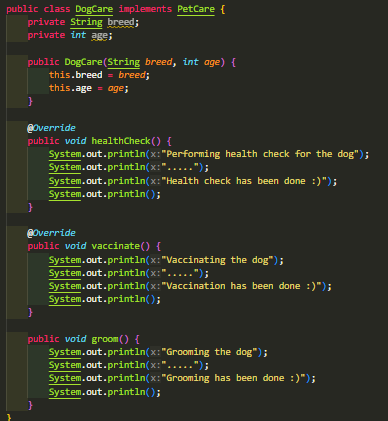
# E. TUGAS

Implementasikan class diagram yang dibuat pada tugas PBO ke dalam kode program.

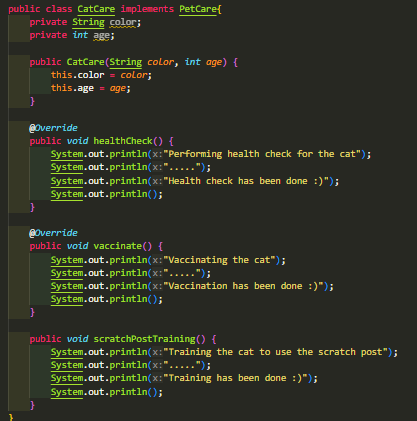
Interface PetCare



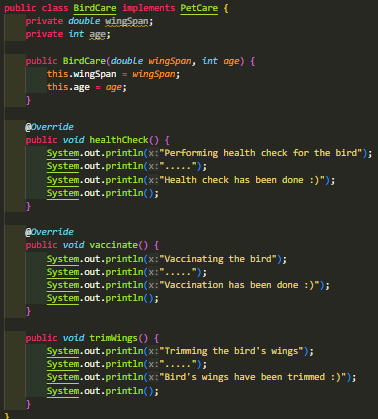
Class DogCare



Class CatCare



Class BirdCare



Class Main

