



## POLITEKNIK MASAMY INTERNASIONAL

SK Menristekdikti RI Nomor: 731/KPT/I/2018

Jalan Ikan Paus No.10-15 Kertosari Banyuwangi - 68411  
Telp (0333) 3384593 – <http://polmain.info>

### PROGRAM STUDI D3 TEKNIK KOMPUTER

Form:  
B.Ak/eva/04/20

Nama Dosen : *Arif Fahmi, S.T.,M.T.*

Mata Kuliah : Pemrograman Berorientasi Object

Semester : 4 (Empat)

Kode Mata Kuliah : TKV4044

Th. Akdm : 2019/2020.

## BAB

### CLASS ABSTRAK

#### SUB BAB :

1. Class Abstrak
2. Perbedaan Class Abstrak dengan Interface
3. Class Abstrak dan Method Abstrak

## TUJUAN MATERI

1. Mahasiswa memiliki kompetensi dalam menerapkan konsep class abstrak
2. Mahasiswa mampu membuat program java dengan mengaplikasikan class abstrak

## REFERENSI

1. J.Eck, D. (2006). *Introduction to Programming Using Java*. Geneva.
2. An Object-Oriented Approach to Programming Logic and Design, Joyce FarrelUSA, 2013
3. An Introduction to Object Oriented Programming with Java, C. Thomas Wu, McGraw-Hill, New York, 2010.
4. <https://docs.oracle.com/javase/tutorial/java/IandI/polymorphism.html>

## **PEMROGRAMAN BERORIENTASI OBJECT**

### **BAGIAN III LANJUTAN**

#### Overview,

Saat mendengar kata “Kendaraan”, apa yang kamu pikirkan pertama kali? Motor, Mobil, Pesawat. Apapun itu, semuanya adalah kendaraan. Dan kita juga sudah tahu, kendaraan pasti bisa berjalan. Tapi gimana kalau saya bilang: “Kendaraan berjalan”.

Mungkin kamu akan bertanya. Bagaimana cara jalanya?, terbang atau jalan di aspal? saya juga tidak tahu. Inilah yang disebut konsep abstraksi. Kata “Kendaraan” sendiri masih bersifat abstrak.

Tapi kita bisa membayangkan konsep kendaraan itu seperti apa. Jika saya bilang: “Mobil berjalan”. Kamu pasti sudah tahu maksud saya. Mobil adalah benda konkrit atau nyata, sedangkan kendaraan masih abstrak.

Prinsip abstraksi ini juga ada dalam OOP dan kita sebenarnya sudah pernah menggunakannya. Namun tidak sadar aja, kalau itu adalah abstraksi. Nah di Java sendiri, ada yang namanya class abstrak untuk membuat abstraksi. Apa itu class abstrak? dan bagaimana cara penggunaannya?, mari kita bahas dan perinci.

#### 1. Class Abstrak

Class abstrak adalah class yang masih dalam bentuk abstrak. Karena bentuknya masih abstrak, dia tidak bisa dibuat langsung menjadi objek. Sebuah class agar dapat disebut class abstrak setidaknya memiliki satu atau lebih **method abstrak**. Method abstrak adalah method yang tidak memiliki implementasi atau tidak ada bentuk konkritnya. Secara praktisnya **Method abstrak itu adalah method yang tidak memiliki isi namun hanya memiliki nama**.

Berikut contoh penerapan class abstrak,

```
// ini abstrak method
void sayHello();

// ini bukan abstrak method karena
// punya implementasi di body method
void greeting() {
    System.out.println("Hello Java");
}
```

class yang menggunakan method ini, secara otomatis akan menjadi class abstrak.

#### Mengapa Harus Pakai Class Abstrak?

Kita tahu, class abstrak memang belum bisa digunakan secara langsung. Karena itu, agar class abstrak dapat digunakan, maka ia harus dibaut bentuk konkritnya. Cara membuat class abstrak menjadi konkrit adalah dengan membuat implementasi dari method-method yang masih abstrak. Ini bisa kita lakukan dengan pewarisan (inheritance). Class abstrak biasanya digunakan sebagai class induk dari class-class yang lain. Class anak akan membuat versi konkrit dari class abstrak.



Lalu pertanyaanya:

Mengapa class harus dibuat menjadi abstrak?

Memang kita bisa memakai class biasa. Tapi pada suatu kondisi tertentu, class induk tidak ingin kita buat sebagai objek. karena kode methodnya belum jelas mau diimplementasikan seperti apa. Maka class ini sebaiknya dijadikan abstrak.

Contoh studi kasus,

```
class Database {
    String getTableName() {
        return null;
    }
}

class MySQLDatabase extends Database {
    @Override
    String getTableName() {
        return this.sql("SELECT table_name FROM
database.tabel")
    }
}
```

Kedua class tersebut merupakan class biasa. Sekarang mari kita coba buat objek dengan class Database.

```
Database db = new Database();
```

Ini akan valid, tapi sebenarnya objek db tidak bisa digunakan. Saat kita memanggil method getTableName() si class Database akan bingung. Mengapa? Karena belum jelas, bagaimana metode untuk mengambil nama tabel di database. Dan database apa yang akan dipakai? Karena itu, class Database sebaiknya dibuat menjadi abstrak agar tidak bisa digunakan untuk membuat objek. lalu gimana cara membuat class menjadi abstrak?

### A. Cara Membuat Abstrak Class di Java

Cara membuat class abstrak adalah dengan memberikan kata kunci abstract kepada class dan method yang ingin dijadikan abstrak.

Kata kunci untuk menyatakan  
class ini adalah abstrak class

```
abstract class NamaClass {
    abstract void namaMethod();
    abstract int namaMethod(int x);
}
```

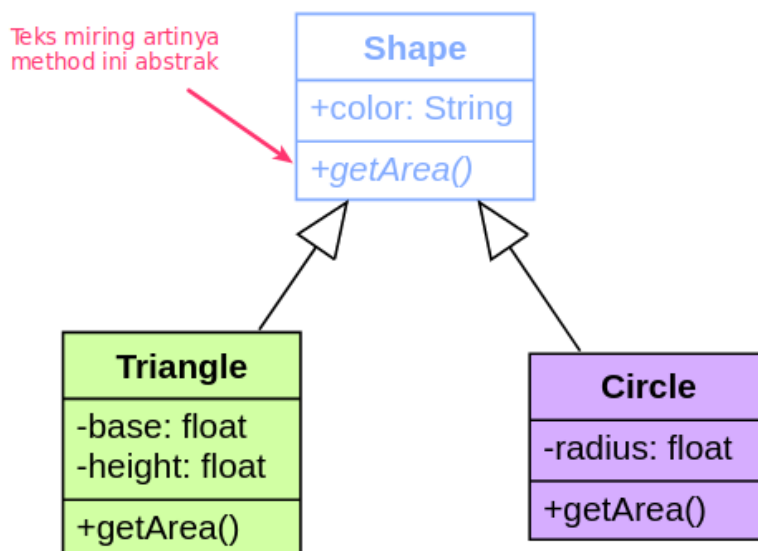
Kata kunci untuk menyatakan  
method ini adalah abstrak method

Contoh:

```
abstract class Database {
    abstract String getDatabaseName();
    abstract String getTableName();
}
```

### Contoh Program Class Abstrak

Baiklah, kita akan coba latihan dengan contoh yang sudah pernah kita bua, yakni bangun datar. Coba perhatikan class diagram berikut;



Class Shape adalah class abstrak, karena dia punya method abstrak `getArea()` (hitung luas). Jika kita membuat objek dengan class Shape dan memanggil method `getArea()`, maka si class Shape akan bingung..

Shape atau bentuk bangun datar yang mau dihitung luasnya seperti apa?

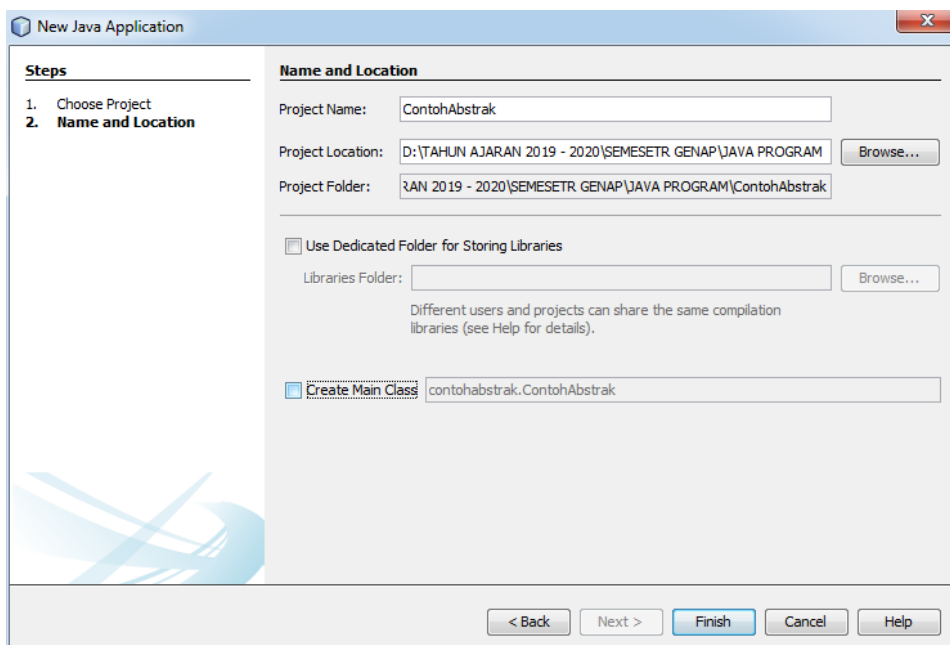
Rumusnya gimana?

Karena itu, class ini kita jadikan abstrak. Soalnya belum jelas cara ngitung luasnya.

Dan kita juga semua tahu, kalau Shape atau bangun datar pasti memiliki luas. Tapi cara ngitungnya berbeda-beda.

Nah sekarang mari kita coba dalam kode program.

A. Buatlah proyek baru di Netbeans dengan nama ContohAbstrak.





B. Setelah itu, pada <default package> buat class baru bernama Shape

C. Dan isi kodenya sebagai berikut:

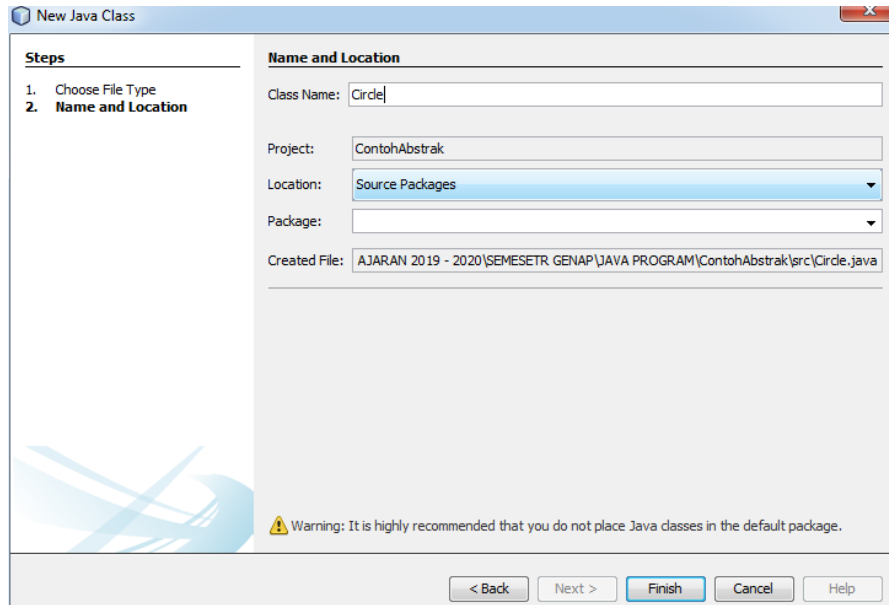
```
1  /**
2   *
3   * @author ARIF FAHMI
4   */
5  public abstract class Shape {
6
7      String color;
8
9      void setColor(String color){
10         this.color = color;
11     }
12
13     String getColor(){
14         return this.color;
15     }
16
17     abstract float getArea();
18 }
```

D. Berikutnya, buat class Triangle. Class ini merupakan turunan dari class Shape.

E. Dan isi kodenya sebagai berikut:

```
1  /**
2   *
3   * @author ARIF FAHMI
4   */
5  public class Triangle extends Shape {
6
7      private float base;
8      private float height;
9
10     public Triangle(int base, int height) {
11         this.base = base;
12         this.height = height;
13     }
14
15     @Override
16     float getArea() {
17         return 0.5f * base * height;
18     }
19 }
20
21 }
```

F. Setelah itu, buat satu lagi turunan dari class Shape, yakni Circle.



The image shows the 'New Java Class' dialog box in an IDE. The 'Steps' tab is active, showing '1. Choose File Type' and '2. Name and Location'. The 'Name and Location' tab is also visible, showing the following fields:

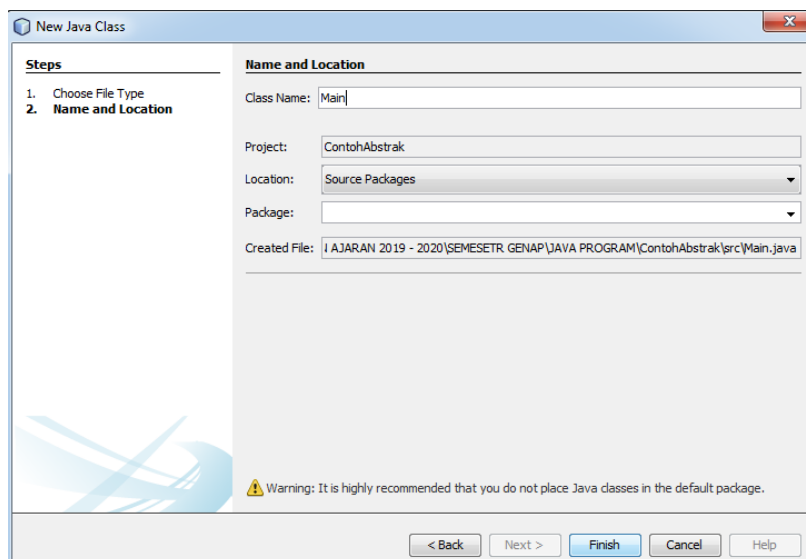
- Class Name: Circle
- Project: ContohAbstrak
- Location: Source Packages
- Package: (empty)
- Created File: AJARAN 2019 - 2020\SEMESTER GENAP\JAVA PROGRAM\ContohAbstrak\src\Circle.java

A warning message at the bottom states: 'Warning: It is highly recommended that you do not place Java classes in the default package.' The 'Finish' button is highlighted.

G. Dan isi kodenya sebagai berikut:

```
1  /**
2   *
3   * @author ARIF FAHMI
4   */
5  public class Circle extends Shape {
6
7      private float radius;
8
9      public Circle(float radius) {
10         this.radius = radius;
11     }
12
13     @Override
14     float getArea() {
15         return (float) (Math.PI * radius * radius);
16     }
17 }
18
19
20 }
```

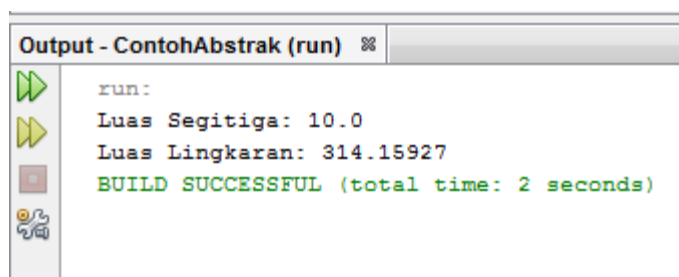
H. Terakhir, buat class Main untuk mencoba membuat objek dengan class yang sudah dibuat.



```
1
2  /**
3   *
4   * @author ARIF FAHMI
5   */
6  public class Main {
7      public static void main(String[] args) {
8          // membuat objek dari class Triangle
9          Shape segitiga = new Triangle(4, 5);
10
11         // membuat objek dari class Circle
12         Shape lingkaran = new Circle(10);
13
14
15         System.out.println("Luas Segitiga: " + segitiga.getArea());
16         System.out.println("Luas Lingkaran: " + lingkaran.getArea());
17     }
18 }
```

Coba perhatikan objek segitiga dan lingkaran! Kedua objek ini memiliki tipe yang sama yakni Shape, tapi mereka dibuat dari class yang berbeda.

I. Nah, sekarang coba jalankan programnya. Maka hasilnya:



## 2. Apa Bedanya Class Abstrak dengan Interface?

Class abstrak dan interface sama-sama digunakan untuk membuat abstraksi.

Keduanya hampir memiliki sifat yang sama. Tapi ada bedanya.

Apa itu?

1. di class abstrak kita bisa baut properti atau variabel sedangkan di interface kita cuma bisa buat konstanta saja
2. di class abstrak kita bisa implementasikan kode method seperti class biasa, sedangkan di interface harus menggunakan default
3. di class abstrak dapat memiliki member private dan protected sedangkan interface harus publik semua
4. Class abstrak diimpelentasikan dengan pewarisan (extends) sedangkan interaface menggunakan implement.

### 3. Class Abstrak & Method Abstrak

Dalam java terdapat abstraksi yang berupa :

- Abstract method
- Abstract class

Untuk lebih memahami konsep ini, mari dikupas satu persatu dari kedua point diatas dimulai dari point yang pertama, yaitu

#### A. Abstract method

Abstract method merupakan sebuah method yang dideklarasikan dengan menambahkan keyword abstract pada deklarasinya, dan tanpa ada implementasi dari method tersebut. Dalam arti, hanya pendeklarasian saja, tanpa tanda sepasang kurung kurawal. Tetapi diakhiri dengan tanda titik koma(;), seperti contoh sederhana berikut ini :

```
abstract void setName();  
abstract void setMakanan();
```

Kode program diatas artinya : Mendeklarasikan abstract method dengan nama **setName** dan **setMakanan**.

#### B. Abstract class

Abstract class merupakan sebuah class yang dideklarasikan dengan menambahkan keyword abstract pada deklarasinya. Dalam deklarasi abstract class ini dapat didefinisikan abstract methodnya, juga dapat tidak dideklarasikan abstract methodnya. Selain itu, untuk abstract class ini tidak dapat di **instansiasi**, akan tetapi dapat di **inherit** oleh subclassnya, seperti contoh sederhana berikut ini :

```
public abstract class Hewan {  
    ...  
}
```

Atau seperti berikut ini bila menginginkan terdapat abstract method di dalamnya :

```
public abstract class Hewan {  
    abstract void setName();  
    abstract void setMakanan();  
}
```

Perlu dipahami, jika sebuah class terdapat abstract method di dalamnya, maka class tersebut **harus** ditandai dengan abstract dalam deklarasikannya. Selain itu, apabila abstract class di inherit oleh class lain(subclass), maka semua abstract method yang terkandung dalam abstract class **harus** di implementasikan dalam subclass. Bila menginginkan abstract methodnya tidak diimplementasi dalam subclass, maka subclass tersebut **harus** ditandai dengan abstract dalam deklarasinya.





Contoh program.

A. Buat project baru pada IDE Netbeans “Penerpan\_Class\_Absttrak”

B. Buat package dengan nama “Class\_Absttrak”

C. Buat class dengan nama “Hewan” dan isi program sebagai berikut;

```
1
2 package Class_Absttrak;
3
4 /**
5  *
6  * @author ARIF FAHMI
7  */
8 public abstract class Hewan {
9     // Deklarasi method
10    abstract void setName();
11    abstract void setMakanan();
12 }
```

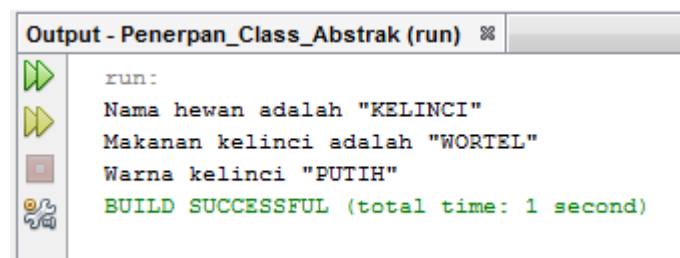
Kode program diatas artinya : Mendeklarasikan abstract class dengan nama Hewan yang di dalamnya terkandung abstract method setName dan setMakanan.

D. Kemudian mari kita buat subclass Kelinci yang inherit dari superclass diatas, seperti berikut ini.

```
1
2  package Class_Abstrak;
3
4  /**
5   *
6   * @author ARIF FAHMI
7   */
8  public class Kelinci extends Hewan {
9      public void setName() {
10         System.out.println("Nama hewan adalah \"KELINCI\");
11     }
12
13     public void setMakanan() {
14         System.out.println("Makanan kelinci adalah \"WORTEL\");
15     }
16
17     public void setWarna() {
18         System.out.println("Warna kelinci \"PUTIH\");
19     }
20
21     public static void main(String[] args) {
22         Kelinci k = new Kelinci();
23         k.setName();
24         k.setMakanan();
25         k.setWarna();
26     }
27 }
```

Kode program diatas artinya : Mendeklarasikan subclass Kelinci yang inherit dari superclass Hewan, dan mengimplementasi abstract method yang dimiliki oleh superclass tersebut.

E. Bila kode program diatas kita eksekusi keluaran yang dihasilkan seperti berikut ini



Berdasarkan keluaran yang dihasilkan diatas, maka semua method yang dimiliki superclass dapat dimiliki oleh subclass, karena kita inherit dan implementasikan semua method tersebut. Apabila kita menginginkan untuk tidak mengimplementasi abstract method diatas, maka kita harus menambahkan keyword abstract pada waktu pendeklarasian subclass.