**Pemrograman Berorientasi Objek**

**Abstract Class**

Abstraksi adalah pengabstrakan atau penyembunyian kerumitan dari suatu proses. Sebagai contoh, orang tidak perlu berpikir bahwa mobil adalahkumpulan dari puluhan atau ribuan bagian-bagian kecil (mesin, kabel, baut,roda, rem, dsb). Orang hanya perlu berpikir bahwa mobil adalah sebuah objek yang telah memiliki prilaku spesifik, yang dapat digunakan sebagai alat transportasi, sehingga dia/mereka tinggal menggunakannya atau mengendarainya tanpa harus mengetahui kerumitan proses yang terdapat di dalam mobil tersebut. Ini artinya, si pembuat mobil telah menyembunyuikan semua kerumitan-kerumitan proses yang terdapat didalam mobil dan pengguna tidak perlu mengetahui bagaimana sistem mesin , transmisi, danrem berkerja. Konsep seperti inilah yang dinamakan dengan abstraksi. Contoh abstraksi dalam model pemrograman berorientasi objek, kita sebagai programmer atau pengembang program tidak perlu tahu secara detail bagaimana cara membuat sebuah tombol, yang perlu kita ketahui adalah bagaimana cara menggunakan objek tombol yang sudah ada (baik data/propeti maupun method-nya). dalam kasus ini, si pembuat objek telah mengabstraksikan semua proses yang berkaitan dengan pembuatan tombol.

**Interface**

Interface adalah wadah dari kumpulan method yang bersifat abstrak atau tidak memiliki implementasi. Sedangkan method yang didefinisikan di dalam interface tersebut akan diimplementasikan oleh class yang mengimplementasikan interface tersebut. Interface merupakan bentuk perluasan dari kelas abstrak.

Selain method, interface juga dapat berisi sekumpulan variable, namun variable yang dideklarasikan di dalam interface harus bersifat final(nilainya tidak dapat diubah /konstan).

Sebagai contoh : Dalam kehidupan nyata dapat diketahui ada manusia yang bekerja sebagai tentara, penyanyi, pengacara, dan sebagainya, tentunya manusia-manusia tersebut selain harus memiliki method standard sebagai seorang manusia, juga harus memiliki method yang sesuai dengan pekerjaannya. Dengan demikian untuk membuat objek manusia yang bekerja sebagai penyanyi, harus dibuat kelas yang merupakan turunan kelas manusia yang meng-implementasikan interface penyanyi.

**Perbedaan Interface dan Abstract Class :**

- Interface hanya bisa mendefinisikan Method, tidak bisa mengimplementasikan, sedangkan Abstrat Class bisa mengimplementasikan Method

- Interface tidak boleh terdiri dari field atau property, Abstract Class bias

- Interface tidak boleh terdiri dari konstruktor dan destruktor, sedangkan Abstract class bisa:

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| Bisa berisi abstract dan non-abstract method. | Hanya boleh berisi abstract method. |
| Kita harus menuliskan sendiri modifiernya. | Kita tidak perlu susah2 menulis public abstract di depan nama method. Karena secara implisit, modifier untuk method di interface adalah **public** dan***abstract***. |
| Bisa mendeklarasikan **constant** dan **instance variable**. | Hanya bisa mendeklarasikan **constant**.  Secara implisit variable yang dideklarasikan di interface bersifat**public**, **static** dan **final**. |
| Method boleh bersifat **static**. | Method tidak boleh bersifat **static**. |
| Method boleh bersifat **final**. | Method tidak boleh bersifat **final**. |
| Suatu abstact class hanya bisa meng-*extend* satu abstract class lainnya. | Suatu interface bisa meng-*extend*  satu atau lebih interface lainnya. |
| Suatu abstract class hanya bisa meng-*extend* satu abstract class dan meng-**implement** beberapa interface. | Suatu interface hanya bisa meng-*extend* interface lainnya.  Dan tidak bisa meng-*implement* class atau interface lainnya. |

**Hubungan dari Interface ke Class**

Seperti yang telah kita lihat dalam bagian sebelumnya, class dapat mengimplementasikan sebuah interface selama kode implementasi untuk semua method yang didefinisikan dalam interface tersedia.

Hal  lain  yang  perlu  dicatat  tentang  hubungan  antara  interface  ke  class-class  yaitu,  class hanya dapat meng-EXTEND satu superclass, tetapi dapat mengimplementasikan banyak interface.

**Tipe Abstraksi**

1. Parsial = Pengaplikasian abstrak sebagian (Class)
2. Full = Pengaplikasikan abstrak sepenuhnya (Interface)

**Contoh Implementasi dalam Bahasa Java**

1. Parsial

abstract class Geometri {

public void printGeometry(){

System.out.println("Geometri");

}

// isi dari metode dibawah ini di definisikan oleh masing-masing subclass

abstract double HitungLuas(); }

class Lingkaran extends Geometri{

private double r;

public void setJari(souble r){

this.r=r;

}

// mendefinisikan metode yang abstrak yang berada di kelas induk

public double HitungLuas(){

return(r\*r\*3.14);

}

}

1. Full

interface geometry{

// semua metode didefiniskan secara abstrak

double HitungLuas();

double HitungKeliling();

}

class Lingkaran implements geometry{

private double r;

public void setJari(double r){

this.r=r;

}

// mendefinisikan metode yang abstrak yang berada di kelas induk

public double HitungLuas(){

return(r\*r\*3.14);

}

public double HitungKeliling(){

return(2\*r\*3.14);

}

}