# JOBSHEET W08 ABSTRACT CLASS

Nama : M. Hasan Basri

Kelas : SIB2C

NIM : 2241760139

1. **KOMPETENSI**
   1. Memahami konsep dasar dan tujuan abstract class
   2. Mampu menerapkan abstract class dalam suatu kode program
   3. Mampu membuat subclass yang meng-extend abstract class dengan mengimplementasikan seluruh abstract method-nya

# PENDAHULUAN

Abstract class merupakan class yang **tidak dapat diinstansiasi namun dapat di-extend**. Umumnya abstract class digunakan sebagai **generalisasi** atau **guideline** dari subclass dan hanya bisa digunakan lebih lanjut setelah **di-extend** oleh **concrete class** (class pada umumnya)

Abstract class memiliki karakteristik sebagai berikut:

* 1. Selalu dideklarasikan dengan menggunakan keyword “**abstract class**”
  2. Dapat memiliki atribut dan methods (yang bukan abstract method) seperti concrete class
  3. Umumnya memiliki **abstract method**, yaitu method yang hanya dideklarasikan tetapi tidak memiliki implementasi (body)
     + Abstract method mendifinisikan apa saja yang bisa dilakukan oleh sebuah class namun tidak ada detail bagaimana cara melakukannya
     + Untuk membuat abstract method, hanya menuliskan deklarasi method tanpa body dan menggunakan keyword abstract

Untuk mendeklarasikan abstract class:

# public abstract class <NamaClass>

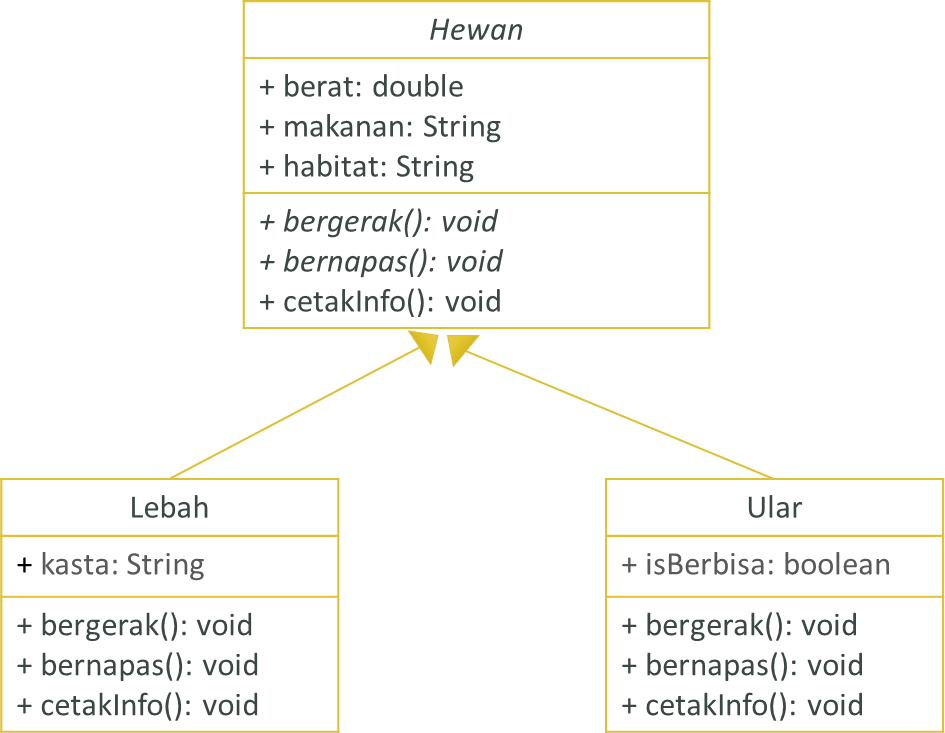
Untuk mendeklarasikan method abstract:

# public abstract <return\_type> <namaMethod>();

Contoh:



Secara umum, notasi class diagram untuk abstract class sama dengan concrete class, namun nama kelas dicetak miring atau ditambah anotasi <<abstract>> di atas nama kelas. Di samping itu abstract method harus dicetak miring juga seperti pada contoh berikut.



Cara menggunaan abstract class:

* + - Abstract class tidak dapat diinstansiasi (tidak dapat dibuat objectnya). Baris kode berikut akan memunculkan *compilation error* “Hewan is abstract; cannot be instantiated”

Hewan hewan1 = new Hewan();

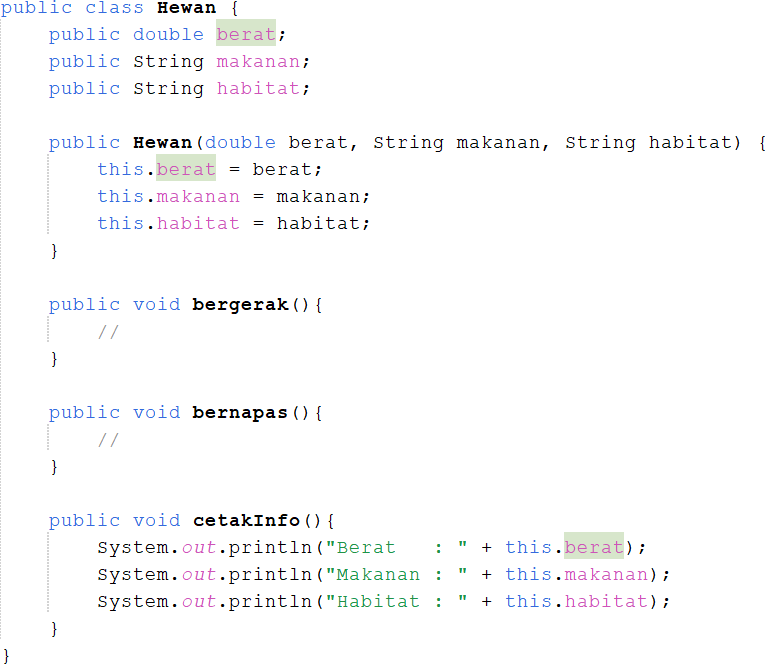
* + - Untuk menggunakan abstract class, dibuat concrete class yang meng-extend abstract class tersebut
      * concrete class menggunakan extends keyword
      * concrete class harus mengimplementasi semua abstract method
    - Class yang menge-extend abstract class tetapi tidak mengimplementasi seluruh abstract method nya maka harus dideklarasikan sebagai abstract class juga

Fungsi abstract class:

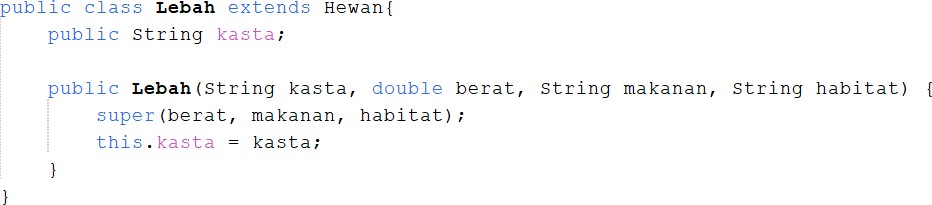
* + - Mencegah suatu class diinstansiasi atau dibuat objeknya
    - Sebagai generalisasi/superclass pada class hierarki
    - Sebagai guideline untuk subclass dengan cara memaksa subclass untuk mengimplementasikan abstract method

# PERCOBAAN

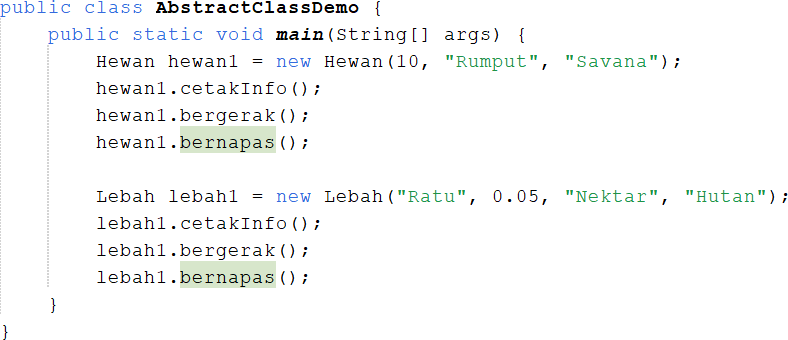
1. **PERCOBAAN 1**
   1. Buatlah project baru dengan nama Praktikum08 kemudian buat class baru dengan nama Hewan. Method bernapas dan bergerak tidak memiliki statement atau baris kode.



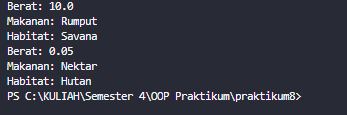
* 1. Buat class Lebah sebagai subclass dari class Hewan sebagai berikut



* 1. Buat class main dengan nama AbstractClassDemo lalu instansiasi objek dari class Hewan dan class Lebah. Run program kemudian amati hasilnya.



Output :

****

# PERTANYAAN

* 1. Bagaimana hasil pada langkah 3? Apakah objek hewan1 dan lebah1 berhasil diinstansiasi?

Jawab : Iya, object dari hewan 1 dan hewan 2 berhasil diinstansiasi.

* 1. Menurut Anda, mengapa tidak ada baris program pada method bergerak() dan bernapas() pada class Hewan()?

Jawab : Tidak terdapat baris program karena hanya digunakan untuk penanda bahwa hewan bisa bergerak dan bernafas, namun karena pada baris program pada method bergerak() dan bernapas() masih belum tau apa yang akan dilakukan dimethod tersebut(karena bisa saja proses /cara bernafas dan bergerak setiap hewan berbeda).

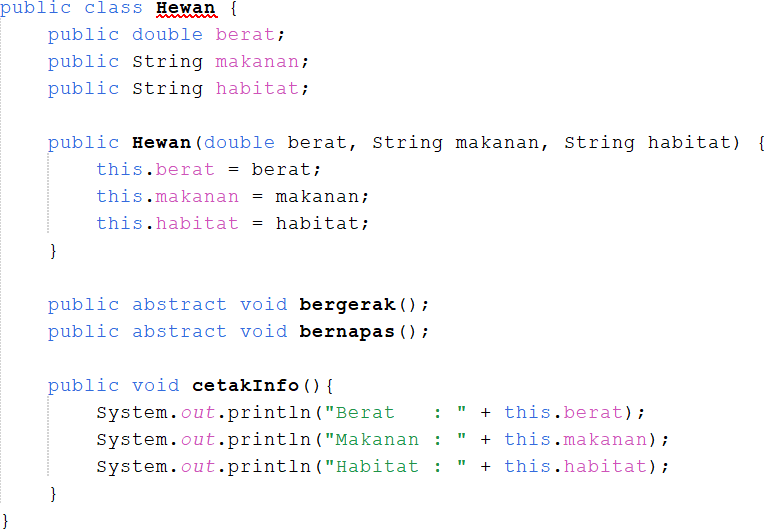
* 1. Class Lebah tidak memiliki method bergerak(), bernapas(), dan cetakInfo(), mengapa tidak terjadi error pada AbstractClassDemo?

Jawab : Tidak terjadi error pada AbstractClassDemo karena kelas Lebah mewarisi method cetakInfo(), bergerak(), dan bernapas() dari kelas Hewan.

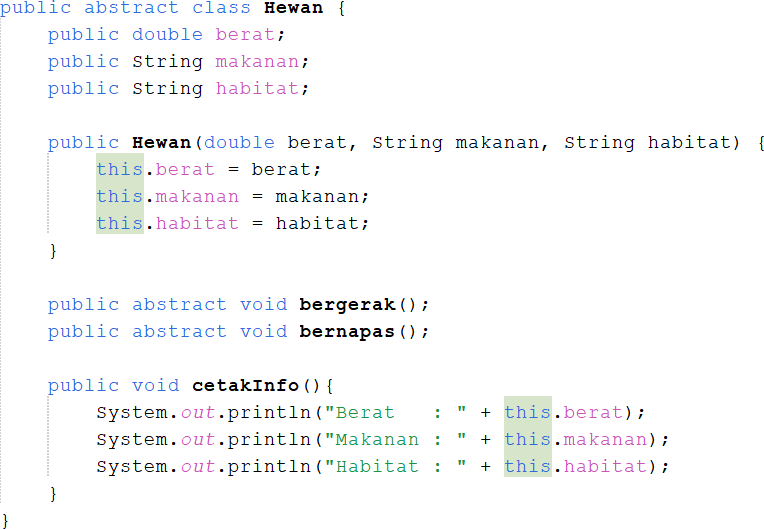
tidak terjadi error, namun tidak bisa memunculkan output karena tidak ada program dalam method tersebut

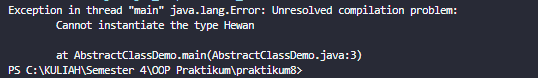
# PERCOBAAN 2

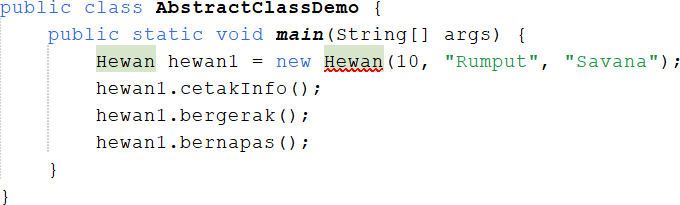
* 1. Ubah method bergerak dan bernapas menjadi abstract method.



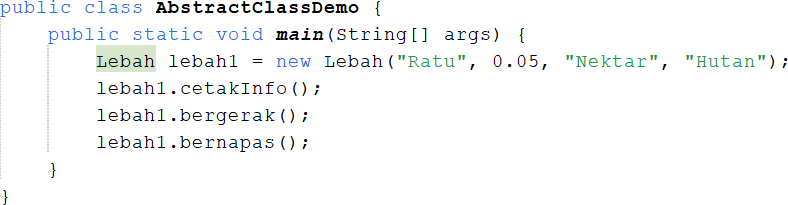
* 1. Akan muncul error sebagai berikut  
     
  2. Ubah class Hewan menjadi abstract Class. Jalankan program kemudian amati hasilnya.

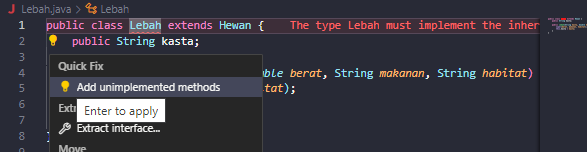


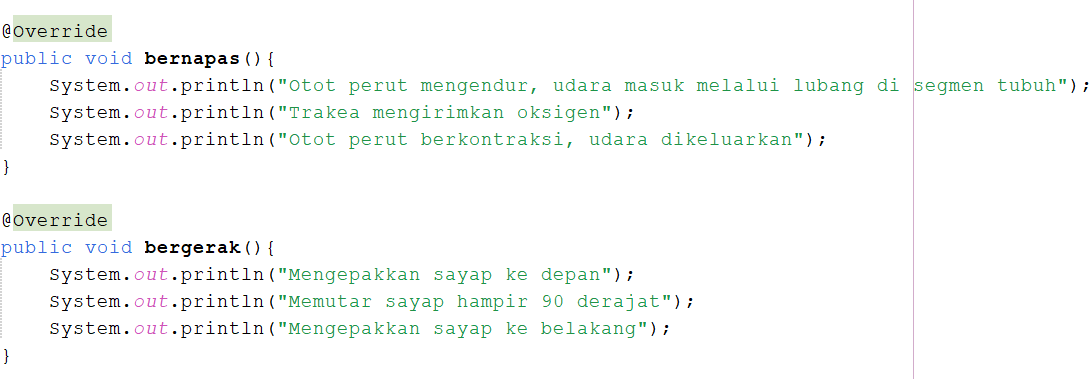
* 1. Ubah class demo sebagai berikut. Run program kemudian amati hasilnya  
     



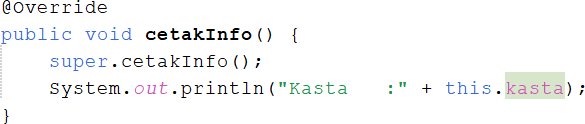
* 1. Ubah class demo sebagai berikut. Run program kemudian amati hasilnya

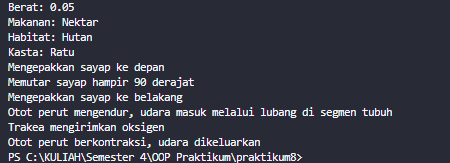


* 1. Klik icon lampu pada class Lebah, kemudian pilih option “Implement all abstract method”  
     
  2. Implementasi method bergerak dan bernapas pada class Lebah sebagai berikut. Run program kemudian amati hasilnya.



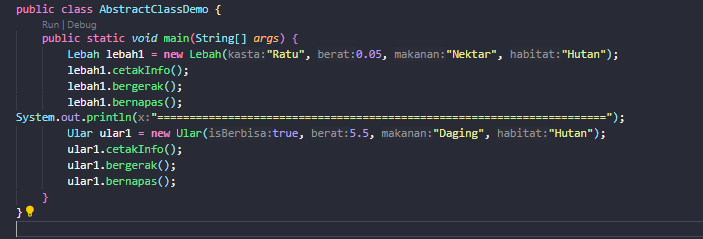
* 1. Tambahkan method cetakInfo() pada class Lebah. Run program kemudian amati hasilnya.



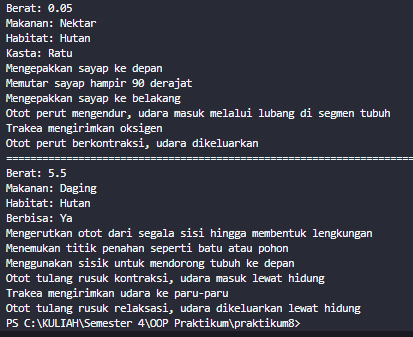


* 1. Buat class Ular kemudian sebagai berikut. Instansiasi objek bertipe Ular pada class AbstractClassDemo. Ekseksusi ketiga method untuk object tersebut.





Output :



# PERTANYAAN

* 1. Pada langkah 1, mengapa sebaiknya method bergerak() dan bernapas() dideklarasikan sebagai abstract method?  
     Jawab : Karena keduanya merupakan perilaku yang berbeda-beda tergantung pada jenis hewan yang diturunkan. Dengan mendeklarasikan method tersebut sebagai abstract, kelas turunan yang mewarisi kelas Hewan wajib mengimplementasikan method tersebut sesuai dengan perilaku spesifik hewan tersebut.
  2. Mengapa pada langkah 2 muncul error?

Jawab : Pada langkah 2, muncul error karena kelas Hewan tidak dideklarasikan sebagai abstract class, tetapi memiliki abstract method bernapas(). Hal ini bertentangan dengan aturan dalam Java dimana sebuah class yang memiliki abstract method harus dideklarasikan sebagai abstract class.

* 1. Apakah sebuah class yang memiliki abstract method harus dideklarasikan sebagai abstract class?  
     Jawab : Ya, sebuah class yang memiliki abstract method harus dideklarasikan sebagai abstract class.
  2. Sebaliknya, apakah abstract class harus memiliki abstract method?

Jawab : Tidak, sebuah abstract class tidak harus memiliki abstract method. Abstract class dapat memiliki implementasi lengkap dari semua methodnya dan bahkan tidak memiliki method abstract sama sekali.

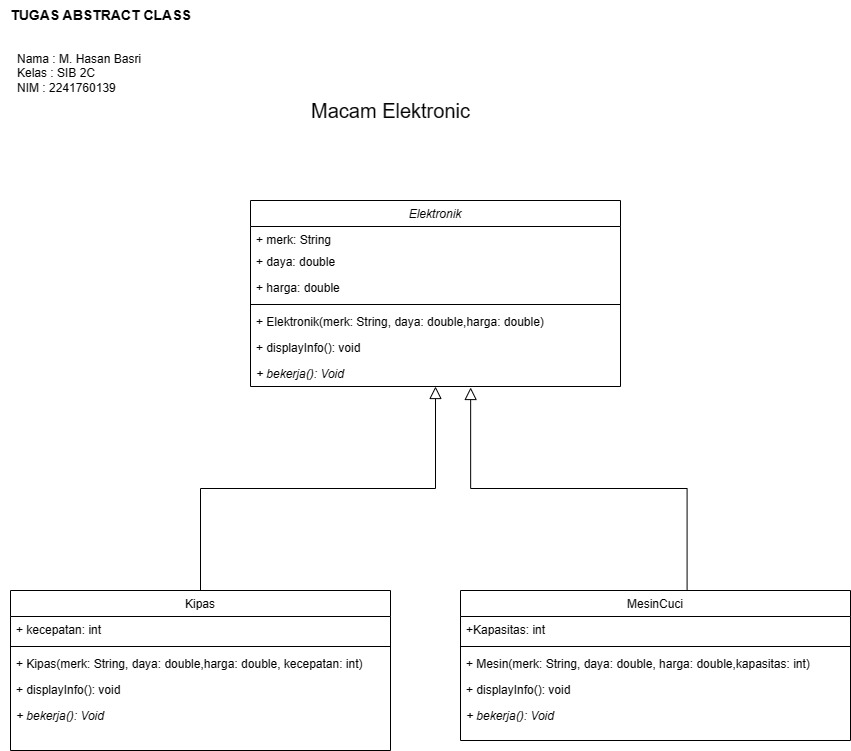
* 1. Mengapa muncul error pada langkah 4?  
     Jawab : Pada langkah 4, muncul error karena class Hewan dideklarasikan sebagai abstract class, dan objek tidak dapat diinstansiasi dari class abstract.
  2. Apakah abstract class dapat memiliki constructor?  
     Jawab : Ya, abstract class dapat memiliki constructor, tetapi constructor tersebut tidak dapat digunakan untuk membuat objek dari abstract class. Constructor abstract class biasanya digunakan untuk melakukan inisialisasi atau setup yang diperlukan oleh class turunannya.
  3. Apakah constructor abstract class dapat dipanggil?  
     Jawab : Tidak, constructor abstract class tidak dapat dipanggil secara langsung. Constructor abstract class hanya akan dipanggil secara tidak langsung melalui constructor dari class turunannya.
  4. Pada langkah 6-8, mengapa method bergerak() dan bernapas() **harus** di-override, namun method cetakInfo() **tidak harus** di-override?  
     Jawab : Karena method bergerak() dan bernapas() dideklarasikan sebagai abstract method di class Hewan, sehingga class turunan yang meng-extend Hewan wajib mengimplementasikan method tersebut.   
     Namun, method cetakInfo() memiliki implementasi lengkap di class Hewan (tidak di deklarasikan sebagai abstract method), sehingga tidak wajib di-override di class turunannya.
  5. Simpulkan kegunaan dari abstract method

Jawab : Abstract method digunakan untuk mendefinisikan perilaku yang harus diimplementasikan oleh class turunan. Dengan menggunakan abstract method, kita dapat menetapkan pola perilaku yang harus diikuti oleh class turunan, namun implementasi detilnya dibiarkan tergantung pada class turunan tersebut.

* 1. Simpulkan kegunaan dari abstract class  
     Jawab : Abstract class digunakan untuk menyediakan kerangka kerja untuk class turunannya. Abstract class dapat mendefinisikan method abstract yang harus diimplementasikan oleh class turunan, serta method non-abstract yang dapat digunakan secara langsung oleh class turunan. Abstract class juga dapat memiliki implementasi default untuk method-method tertentu yang dapat digunakan oleh semua class turunannya.

# 4. TUGAS

Implementasikan class diagram yang telah dirancang pada tugas PBO Teori ke dalam kode program. Selanjutnya buatlah instansiasi objek dari masing-masing subclass kemudian coba eksekusi method-method yang dimiliki.



Output setelah implementasi:

