LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

JOBSHEET 10: ABSTRACT CLASS

oleh :
Halim Teguh Saputro
2E
2141762122

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG
Jl. Soekarno Hatta No .9, Jatimulyo, Kec. Lowokwaru, Kota Malang,
JawaTimur 65141

KOMPETENSI

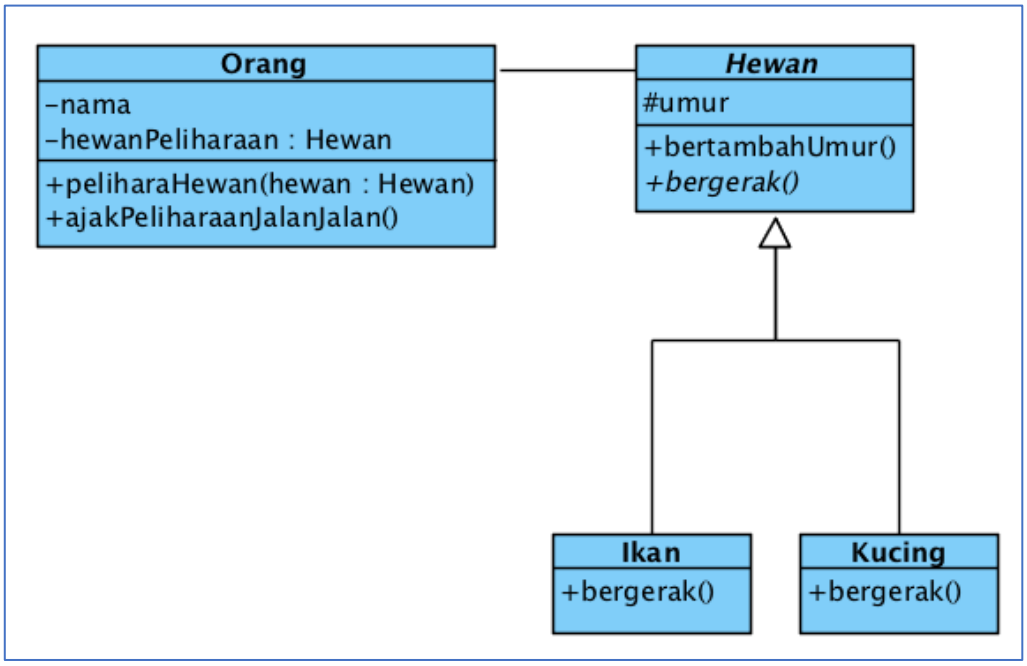
Setelah melakukan percobaan pada modul ini, mahasiswa memahami konsep:

1. Menjelaskan maksud dan tujuan penggunaan Abstract Class
2. Menerapkan Abstract Class di dalam pembuatan program

PRAKTIKUM 1. ABSTRACT CLASS

Di dunia ini terdapat banyak jenis hewan. Semua hewan memiliki beberapa karakteristik yang sama, seperti contohnya semua hewan memiliki umur, hewan apapun itu, umurnya akan bertambah sama jumlahnya setiap tahun.

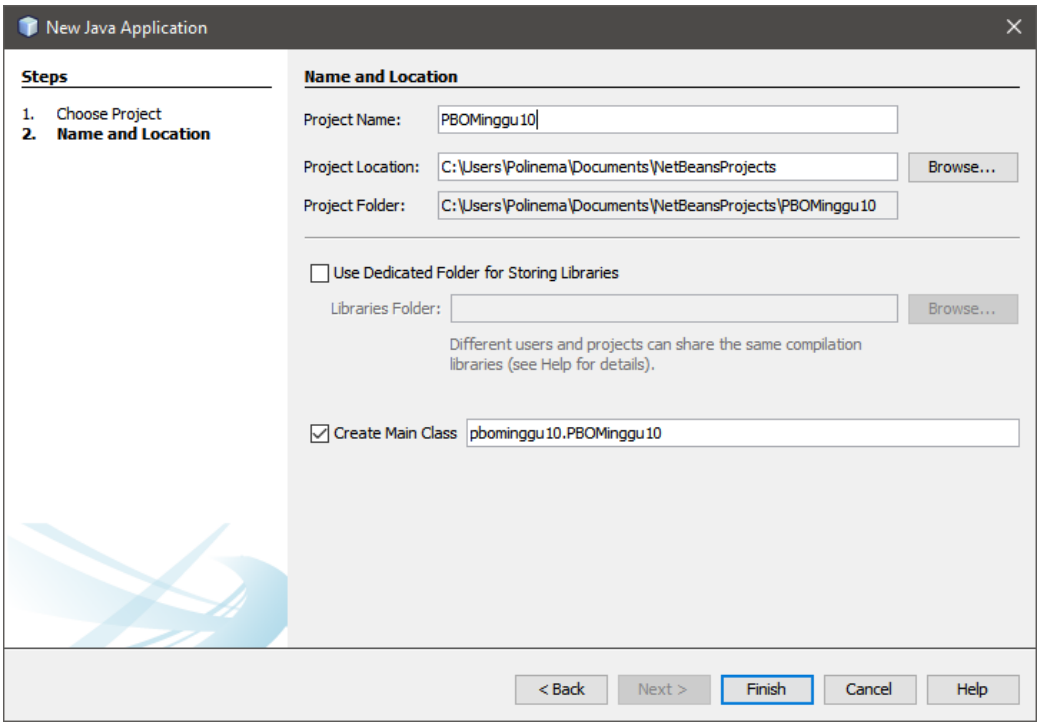
Selain karakteristik yang sama, masing-masing hewan juga memiliki karakteristik yang berbeda satu dengan yang lainnya. Contohnya dalam hal **bergerak**. Cara kucing bergerak berbeda dengan cara ikan bergerak. Kucing bergerak dengan cara melangkahkan kaki-kakinya sedangkan ikan bergerak dengan cara menggerakkan siripnya.



Setiap orang yang memelihara hewan dapat mengajak hewan peliharaannya berjalan (membuat agar hewan peliharaannya bergerak). Namun orang yang memelihara hewan yang berbeda, akan berbeda pula cara hewan peliharaannya dalam bergerak.

Pada percobaan pertama ini kita akan membuat sebuah program yang menggambarkan skenario di atas dengan memanfaatkan abstract class.

1. Buatlah sebuah project baru di NetBeans dengan nama **PBOMinggu10**



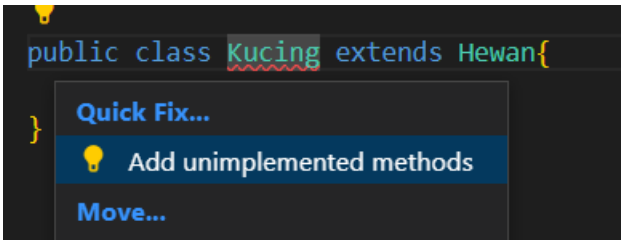
2. Pada package pbominggu10, tambahkan package baru dengan cara klik kenan nama package > New > Java Package…
3. Beri nama package tersebut dengan nama abstractclass. Semua class yang dibuat pada percobaan 1 ini diletakkan pada package yang sama, yaitu package abstractclass ini.
4. Pada package baru tersebut tambahkan class baru.
5. Beri nama class baru tersebut, yaitu class **Hewan**
6. Pada class Hewan tersebut. Ketikkan kode berikut ini.

   Class Hewan tersebut adalah abstract berisi property dan method biasa, ditambah sebuah method abstract Bernama bergerak(). Method tersebut didepannya terdapat kata kunci abstract dan tidak memiliki badan fungsi method ini nantinya akan di override oleh class mana saja yang menjadi class turunan dari class Hewan tersebut.

```
package abstractclass;
public abstract class Hewan {
    private int umur;
```

```
    protected Hewan() {
        this.umur = 0;
    }

    public void bertambarUmur() {
        this.umur += 1;
    }

    public abstract void bergerak();
}
```

7. Dengan cara yang sama, buatlah class dengan nama Kucing yang meng-extend class Hewan.di dalam class Kucing tersebut, setelah anda menuliskan kode seperti dibawah, maka akan muncul ikon lampu peringatan, klik lampu tersebut dan kemudian pilih implement all abstract methods



8. Maka akan secara otomatis dibuatkan fungsi yang meng-override fungsi abstract bergerak() yang ada pada class hewan.

```
package abstractclass;

public class Kucing extends Hewan{

    @Override
    public void bergerak() {
        // TODO Auto-generated method stub
    }
}
```

9. Ubahlah badan fungsi tersebut dengan mengganti kode didalamnya menjadi seperti berikut

```
    @Override
    public void bergerak() {
        // TODO Auto-generated method stub
        System.out.println("Berjalan dengan KAKI, \"Slashh.. Slasshh..\"");
```

```
    }
```

10. Dengan cara yang sama seperti Ketika membuat class Kucing, buatlah class Hewan baru Bernama Ikan dan buatlah kodenya seperti pada gambar dibawah.

```
package abstractclass;

public class Ikan extends Hewan {

    @Override
    public void bergerak() {
        // TODO Auto-generated method stub
        System.out.println("Berenang dengan SIRIP, \"Slutt.. Sluuutt..\"");
    }

}
```

11. Selanjutna, buatlah class biasa baru yang Bernama class Orang. Class ini adalah class yang menjadi penggunaan dari class abstract Hewan yang sudah dibuat sebelumnya. Ketikkan pada class orang tersebut baris-baris kode seperti dibawah ini.

```
package abstractclass;

public class Orang {
    private String nama;
    private Hewan hewanPeliharaan;

    public Orang(String nama) {
        this.nama = nama;
    }

    public void peliharaHewan(Hewan hewanPeliharaan) {
        this.hewanPeliharaan = hewanPeliharaan;
    }

    public void ajakPeliharaanJalanJalan() {
        System.out.println("Namaku " + this.nama);
        System.out.print("Hewan peliharaanku berjalan dengan cara: ");
        this.hewanPeliharaan.bergerak();
        System.out.println("----------------------------------------");
    }
}
```

12. Terakhir, buatlah sebuah Main Class baru di dalam package yang sama. Bari nama class baru tersebut dengan nama class Program. Ketikkan didalamnya seperti kode dibawah ini.

```
package abstractclass;

public class Program {
    public static void main(String[] args) {
        Kucing kucingKampung = new Kucing();
        Ikan lumbaLumba = new Ikan();

        Orang halim = new Orang("Halim");
        Orang teguh = new Orang("Teguh");

        halim.peliharaHewan(kucingKampung);
        teguh.peliharaHewan(lumbaLumba);

        halim.ajakPeliharaanJalanJalan();
        teguh.ajakPeliharaanJalanJalan();
    }
}
```

13. Jalankan class tersebut dengan cara klik kanan pada class Program kemudian pilih Run File (Shift + F6)
14. Perhatikan dan amati hasilnya!

```
erbasis Objek\Praktikum\10. Abstract Class\Jobsheet10\bin" abstractclass.Program
Namaku Halim
Hewan peliharaanku berjalan dengan cara: Berjalan dengan KAKI, "Slashh.. Slasshh.."
----------------------------------------
Namaku Teguh
Hewan peliharaanku berjalan dengan cara: Berenang dengan SIRIP, "Slutt.. Sluuutt.."
----------------------------------------
```

Keterangan: output yang muncul dari program praktikum ini yaitu, program menampilkan nama Orang (Halim dan Teguh) kemudina menampilkan cara hewan peliharaannya berbegerak.

15. Pertanyaan diskusi!

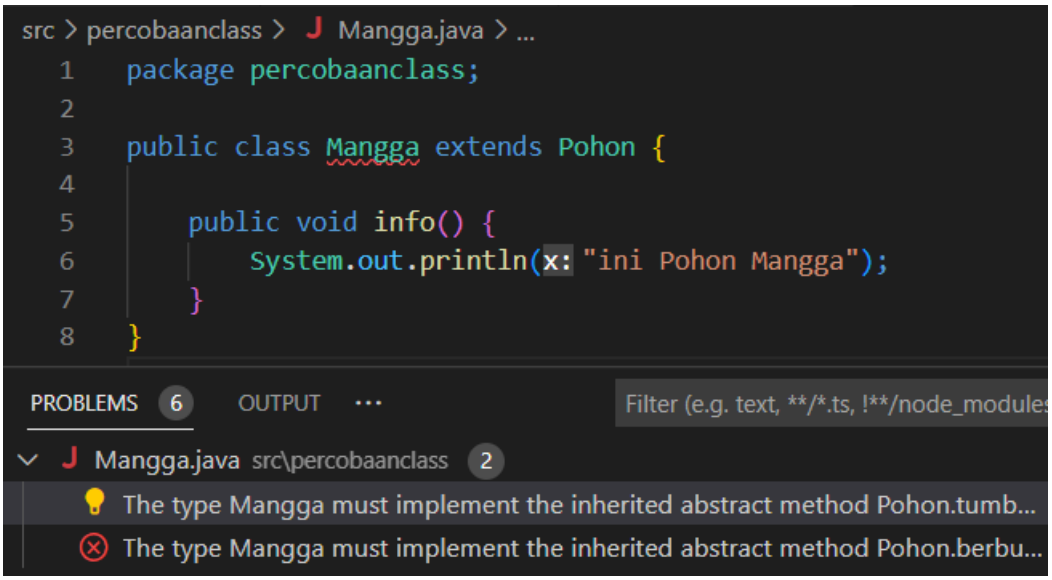
Bolehkah apabila sebuah class yang meng-extend suatu abstract class tidak mengimplementasikan method abstract yang ada di class induknya? Buktikan!

Jawab:

a. Pohon (Parent)

```java
package percobaanclass;

public abstract class Pohon {
    private int tinggi;

    protected Pohon() {
        this.tinggi = 0;
    }

    public abstract void tumbuh();

    public abstract void berbuah();
}
```

b. Mangga (child)



Berdasarkan dari percobaan tersebut, dapat diambil kesimpulan bahwa class yang meng-extend suatu abstract class harus mengimplementasikan method abstract yang ada di class parentnya. Jadi, dalam menyelesaikan masalah tesebut dapat menambahkan code seperti dibawah ini (menyesuaikan dengan method yang dimiliki class parent)

```java
package percobaanclass;

public class Mangga extends Pohon {

    public void info() {
```

```
        System.out.println("ini Pohon Mangga");
    }

    @Override
    public void tumbuh() {
        // TODO Auto-generated method stub
        System.out.println("Pohon Mangga sedang bertumbuh");
    }

    @Override
    public void berbuah() {
        // TODO Auto-generated method stub
        System.out.println("Pohon Mangga sudah berbuah");
    }
}
```

PERTANYAAN

1. Berikan penjelasan terkait jalannya program diatas

   Jawab: program merupakan implementasi konsep Hewan Peliharaan yang bisa bergerak dan bertambah umur. Namun, beberapa Hewan memiliki cara bergerak yang berbeda-beda. Sehingga kita dapat memanfaatkan konsep atau fitur overriding. Jadi, pada class Hewan yang memiliki method abstrak akan dipanggil pada masing masing class objek (Kucing dan Ikan). Class Kucing dan Class Ikan akan memiliki method yang harus dimiliki dan method tersebut akan digunakan untuk menampilkan cara hewan tersebut bergerak. Setelah itu, Class Orang memanggil Class Kucing dan Class Ikan agar bisa digunakan, setelah itu di deklarasikan pada Class Program (main program) agar bisa di run.

2. Tunjukkan hasil kompilasi program dan berikan penjelasan singkat jika method bergerak() diubah menjadi method abstract!

   Jawab:

```
erbasis Objek\Praktikum\10. Abstract Class\Jobsheet10\bin" abstractclass.Program
Namaku Halim
Hewan peliharaanku berjalan dengan cara: Berjalan dengan KAKI, "Slashh.. Slasshh.."
---------------------------------------------
Namaku Teguh
Hewan peliharaanku berjalan dengan cara: Berenang dengan SIRIP, "Slutt.. Sluuutt.."
---------------------------------------------
```

Keterangan: pada class abstrak Hewan memiliki sebuah mehod abstract bergerak yang nanti rulenya akan diisi pada class childnya yaitu Class Kucing dan Hewan. Jadi, pada class induk method bergerak tidak bisa diberi sebuah rule.

3. Tunjukkan hasil kompilasi program dan berikan penjelasan singkat jika tidak dilakukan overriding terhadap method bergerak()

   Jawab:

```
ogram'
Namaku Halim
Hewan peliharaanku berjalan dengan cara: Exception in thread "main" java.lang.Error: Unresolve
d compilation problem:
        The type Kucing must implement the inherited abstract method Hewan.bergerak()

        at abstractclass.Kucing.bergerak(Kucing.java:3)
        at abstractclass.Orang.ajakPeliharaanJalanJalan(Orang.java:18)
        at abstractclass.Program.main(Program.java:14)
PS C:\Users\Halim\Downloads\POLINEMA\Semester 3\5. Pemrograman Berbasis Objek\Praktikum\10. Ab
```

Keterangan: jika tidak dilakukan overriding terhadap method bergerak dengan menghilangkan method tersebut maka hasilnya saat di compile akan seperti gambar di atas, karena class child dari class abstrak harus mewarisi atau memiliki method abstrack parentnya.

Namun, jika yang dimaksud yaitu hanya menghapus kata override pada penulisan code tidak terjadi error

```
3   public class Kucing extends Hewan {
4
5   //@Override
6       public void bergerak() {
7           // TODO Auto-generated method stub
8           System.out.println(x: "Berjalan dengan KAKI, \"Slashh.. Slasshh..\"
9       }
10
11  }
```

4. Tunjukkan hasil kompilasi program dan berikan penjelasn singkat jika abstract method bergerak() yang dideklarasikan dalam Class Ikan

   Jawab:

```
Namaku Teguh
Hewan peliharaanku berjalan dengan cara: Berenang dengan SIRIP, "Slutt.. Sluuutt.."
----------------------------------------
```

Keterangan: mirip dengan class Kucing, pada class Ikan juga harus memiliki method abstract parentnya yaitu method bergerak(). Namun, terdapat perbedaan pada rule output yang harus ditampilkan. Ikan bergerang dengan menggunakan sirip.