



LEMBAR JAWABAN
QUIZ 2 PEMROGRAMAN BERORIENTASI OBYEK 2022/2023

Nama	:	Margaretha Violina Putri Purnomo
NIM	:	2141762056
KELAS	:	SIB 2F

<https://github.com/MargarethaViolinaPutri/PBO->

Jawaban :

1.

Jadi terdapat class abstract Dinosaurus yang berisi method void makan() dan berjalan(). Kemudian class abstract ini merupakan parent dari 3 sub class yang mengextends yaitu class TyrannosaurusRex, Triceratops, dan Oviraptor. Pada masing-masing subclass mewarisi method dari parentnya yaitu method makan dan berjalan. Kemudian isikan outputan menggunakan System.out.println sesuai yang diinginkan dimasing-masing subclass. Selain makan() dan berjalan(), khusus class Oviraptor ada tambahan method bertelur() yaitu method override dari class interface IBertelur karena class Oviraptor mengimplements class interface IBertelur.

Kemudian untuk class Pemburu juga merupakan class parent yang memiliki subclass AnakPemburu. Di class pemburu memiliki parameter class TyrannosaurusRex dengan nama objek trex. Di class ini kita bisa memanggil method berjalan dan makan dari class TyrannosaurusRex yang sebelumnya bisa ditambahkan isi dengan System.out.println.

Selanjutnya untuk class AnakPemburu yang extends dari class Pemburu dan implements dari class interface IMengambilTelur berypa method mengambilTelur dengan parameter class Dinosaurus dengan nama objek dino. Di class ini kita bisa memanggil method berjalan dan makan dari class anak dino detailnya bisa di class main yang sebelumnya bisa ditambahkan isi dengan System.out.println.

Class Main sendiri seperti pada umumnya, buat objek kemudian kode *pm.berburu(ty)* adalah memanggil method berburu dikelas Pemburu dengan parameter ty artinya pemburu hanya bisa memburu tyrex saja tidak bisa ke dinosaurus lain seperti tc dan ov, kemudian *apm.mengambilTelur* memanggil method mengambilTelur di class AnakPemburu dengan parameter tc,. Nah tadi di method *apm.mengambilTelur* kita tadi menuliskan dino.berjalan dan dino.makan nah disini kita bisa mengisikan parameter di main bukan hanya tyrek/ty tapi juga jenis dinosaurus lain.

2.

Source Code (Class Interface IBertelur)

```
13 public interface IBertelur {  
    public abstract void bertelur();  
}
```

Source Code (Class Interface IMengambilTelur)

```

12 public interface IMengambilTelur {
13     public abstract void mengambilTelur(Dinosaurus dino);
14 }

```

Source Code (Class Abstract Dinosaurus)

```

15 public abstract class Dinosaurus {
16     public void makan() {
17         System.out.println("Dinosaurus Sedang Mencari Makan");
18     }
19     public void berjalan() {
20         System.out.println("Dinosaurus Sedang Berjalan Di Hutan");
21     }
22 }

```

Source Code (Class TyrannosaurusRex)

```

10 */
11 public class TyrannosaurusRex extends Dinosaurus{
12     public void makan() {
13         System.out.println("TyrannosaurusRex Pemakan Daging");
14     }
15     public void berjalan() {
16         System.out.println("TyrannosaurusRex Berjalan Di Hutan");
17     }
18 }

```

Source Code (Class Triceratops)

```

11 public class Triceratops extends Dinosaurus{
12     public void makan() {
13         System.out.println("Triceratops Pemakan Tumbuhan");
14     }
15     public void berjalan() {
16         System.out.println("Triceratops Berjalan Di Hutan");
17     }
18 }

```

Source Code (Class Oviraptor)

```

10 */
11 public class Oviraptor extends Dinosaurus implements IBertelur{
12     public void makan() {
13         System.out.println("Oviraptor Pemakan Telur");
14     }
15     public void berjalan() {
16         System.out.println("Oviraptor Berjalan Di Hutan");
17     }
18
19     @Override
20     public void bertelur() {
21         System.out.println("Berkembang Biak Dengan Bertelur");
22     }
23 }

```

Source Code (Class Pemburu)

```

10 */
11 public class Pemburu {
12     public void berburu(TyrannosaurusRex trex){
13         System.out.println("Pemburu Sedang Mengawasi Telur TyrannosaurusRex");
14         System.out.println();
15         System.out.println("Ciri Pemilik Telur Yang Akan Diambil: ");
16         trex.berjalan();
17         trex.makan();
18     }
19 }

```

Source Code (Class AnakPemburu)

```

public class AnakPemburu extends Pemburu implements IMengambilTelur{

    @Override
    public void mengambilTelur(Dinosaurus dino) {
        System.out.println( x: "Anak Pemburu Memantau Dinosaurus!!");
        System.out.println();
        dino.berjalan();
        dino.makan();
        System.out.println( x: "Anak Pemburu Mengambil Telur Triceratops");

    }
}

```

Source Code (Main)

```

public class MainDino {
    public static void main(String[] args){
        Pemburu pm = new Pemburu();
        AnakPemburu apm = new AnakPemburu();

        TyrannosaurusRex ty = new TyrannosaurusRex();
        Triceratops tc = new Triceratops();
        Oviraptor ov = new Oviraptor();

        pm.berburu( t rex: ty);
        System.out.println( x: "=====");
        apm.mengambilTelur( dino: tc);

        System.out.println();
        System.out.println( x: "Pemburu dan Anak Pemburu Melihat Dinosaurus: ");
        System.out.println();
        System.out.println( x: "1. TyrannosaurusRex");
        ty.makan();
        System.out.println( x: "2. Triceratops");
        tc.makan();
        System.out.println( x: "3. Oviraptor");
        ov.makan();
        ov.bertelur();
    }
}

```

Output Program :

```

] --- exec-maven-plugin:3.0.0:exec (default-cli) @ :
Pemburu Sedang Mengawasi Telur TyrannosaurusRex

Ciri Pemilik Telur Yang Akan Diambil:
TyrannosaurusRex Berjalan Di Hutan
TyrannosaurusRex Pemakan Daging
=====
Anak Pemburu Memantau Dinosaurus!!

Triceratops Berjalan Di Hutan
Triceratops Pemakan Tumbuhan
Anak Pemburu Mengambil Telur Triceratops

Pemburu dan Anak Pemburu Melihat Dinosaurus:

1. TyrannosaurusRex
TyrannosaurusRex Pemakan Daging
2. Triceratops
Triceratops Pemakan Tumbuhan
3. Oviraptor
Oviraptor Pemakan Telur
Berkembang Biak Dengan Bertelur
-----
BUILD SUCCESS
-----

```

Modifikasi Lain :

1.

Jadi terdapat class abstract Dinosaurus yang berisi method abstract void makan() dan berjalan(). Kemudian class abstract ini merupakan parent darui 3 sub class yang

mengextends yaitu class *TyrannosaurusRex*, *Triceratops*, dan *Oviraptor*. Pada masing-masing subclass override method dari parentnya yaitu method makan dan berjalan. Pada ketiga subclass ini deklarasikan berat dan buat konstruktor berparameternya. Selain *makan()* dan *berjalan()*, tambahkan method *jenis()*, *berkembangBiak()*, *nama()*, *displayData()* dengan isian sesuai yang dimau. Khusus class *Oviraptor* ada tambahan method *bertelur()* yaitu method override dari class interface *IBertelur* karena class *Oviraptor* mengimplements class interface *IBertelur* untuk sisanya sama dengan class *TyrannosaurusRex* dan *Triceratops*.

Kemudian untuk class *Pemburu* juga merupakan class parent yang memiliki subclass *AnakPemburu*. Di class *pemburu* memiliki parameter class *TyrannosaurusRex* dengan nama objek *trex*. Di class ini kita bisa memanggil method dari class *TyrannosaurusRex*, contohnya saya akan memanggil method *nama()* yang sebelumnya itu bisa diisi isian lain dengan `System.out.println`.

Selanjutnya untuk class *AnakPemburu* yang extends dari class *Pemburu* dan implements dari class interface *IMengambilTelur* berypa method *mengambilTelur* dengan parameter class *Dinosaur* dengan nama objek *dino*. Di class ini kita bisa memanggil method dari class anak *dino*, contohnya kita panggil *dino.makan()* dalam method *mengambilTelur()* detailnya bisa di class main yang dimana *Anak Pemburu* bisa mengakses semua subclass *dino*. Dan sebelumnya bisa diid isian dengan `System.out.println`.

Class Main sendiri seperti pada umumnya, buat objek kemudian kode *pm.berburu(ty)* adalah memanggil method *berburu* dikelas *Pemburu* dengan parameter *ty* artinya pemburu hanya bisa memburu *tyrex* saja tidak bisa ke subclass *dinosaur* lain seperti *tc* dan *ov*. Kemudian *apm.mengambilTelur* memanggil method *mengambilTelur* di class *AnakPemburu* dengan parameter *ty/tc/ov*, Nah tadi di method *mengambilTelur* kita tadi menuliskan *dino.makan*, artinya *dino* ini kita bisa mengakses method *makan* disetiap subclass class *dinosaur*. Untuk yang akan ditampilkan tergantung main dan parameter yang diambil

2.

Source Code (Class Interface IBertelur)

```
10  /
11  public interface IBertelur {
12      public abstract void bertelur();
13  }
```

Source Code (Class Interface IMengambilTelur)

```
10  /
11  public interface IMengambilTelur {
12      public abstract void mengambilTelur(Dinosaur dino);
13  }
```

Source Code (Class Abstract Dinosaur)

```
11  public abstract class Dinosaur {
12      public abstract void makan();
13      public abstract void berjalan();
14  }
```

Source Code (Class TyrannosaurusRex)

```

public class TyrannosaurusRex extends Dinosaurus{
    private String berat;

    public TyrannosaurusRex (String berat) {
        this.berat = berat;
    }

    @Override
    public void makan() {
        System.out.println(x: "TyrannosaurusRex Makanan Daging");
    }

    @Override
    public void berjalan() {
        System.out.println(x: "Berjalan Dengan 2 Kaki Belakang");
    }

    public void jenis() {
        System.out.println(x: "Jenis Karnivora");
    }

    public void berkembangBiak(){
        System.out.println(x: "Berkembang Biak dengan Bertelur");
    }

    public void nama(){
        System.out.println(x: "TyrannosaurusRex");
    }

    public void displayData() {
        System.out.println(x: "-----Dinosaurus -----");
        this.nama();
        this.jenis();
        this.berkembangBiak();
        this.makan();
        System.out.println("Berat"+ this.berat);
        this.jalan();
    }
}

```

Source Code (Class Triceratops)

```

public class Triceratops extends Dinosaurus{
    private String berat;

    public Triceratops (String berat) {
        this.berat = berat;
    }

    @Override
    public void makan() {
        System.out.println(x: "Triceratops Makanan Tumbuhan");
    }

    @Override
    public void berjalan() {
        System.out.println(x: "Berjalan Dengan 4 Kaki");
    }

    public void jenis() {
        System.out.println(x: "Jenis Herbivora");
    }

    public void berkembangBiak(){
        System.out.println(x: "Berkembang Biak dengan Bertelur");
    }

    public void nama(){
        System.out.println(x: "Triceratops");
    }

    public void displayData() {
        System.out.println(x: "-----Dinosaurus -----");
        this.nama();
        this.jenis();
        this.berkembangBiak();
        this.makan();
        System.out.println("Berat"+ this.berat);
        this.jalan();
    }
}

```

Source Code (Class Oviraptor)

```

public class Oviraptor extends Dinosaurus implements IBertelur{
    private String berat;

    public Oviraptor (String berat) {
        this.berat = berat;
    }

    @Override
    public void makan() {
        System.out.println( ": "Oviraptor Makan Telur Dinosaurus");
    }
    @Override
    public void berjalan() {
        System.out.println( ": "Berjalan Dengan 2 Kaki Belakang");
    }
    @Override
    public void bertelur() {
        System.out.println( ": "Berkembang Biak Dengan Bertelur");
    }
    public void jenis() {
        System.out.println( ": "Karnivora");
    }
    public void nama() {
        System.out.println( ": "Oviraptor");
    }
    public void displayData() {
        System.out.println( ": "-----Dinosaurus -----");
        this.nama();
        this.jenis();
        this.berkembangBiak();
        this.makan();
        System.out.println("Berat"+ this.berat);
        this.berjalan();
    }
}

```

Source Code (Class Pemburu)

```

1 public class Pemburu {
2     public void berburu(TyrannosaurusRex trex){
3         System.out.println( ": "Pemburu Sedang Mengawasi Telur Dinosaurus");
4         System.out.println( ": "Pemburu Harus Berhati-hati Dengan");
5         trex.nama();
6     }
7 }

```

Source Code (Class AnakPemburu)

```

11 public class AnakPemburu extends Pemburu implements IMengambilTelur{
12
13     @Override
14     public void mengambilTelur(Dinosaurus dino) {
15         System.out.println( ": "Anak Pemburu Memantau Telur Dinosaurus!!");
16         dino.makan();
17     }
18
19 }

```

Source Code (Main)

```

public class MainDino {
    public static void main(String[] args){
        Pemburu pm = new Pemburu();
        AnakPemburu apm = new AnakPemburu();

        TyrannosaurusRex ty = new TyrannosaurusRex(berat: "6,4 ton");
        ty.displayData();

        Triceratops tc = new Triceratops(berat: "12 ton");
        tc.displayData();

        Oviraptor ov = new Oviraptor(berat: "36 kg");
        ov.displayData();

        System.out.println();
        System.out.println(x: "----Pemburu Dan Anak Pemburu----");
        pm.berburu(trex: ty);
        System.out.println(x: "=====");
        apm.mengambilTelur(dino: ty);
        System.out.println(x: "=====");
        apm.mengambilTelur(dino: tc);
        System.out.println(x: "=====");
        apm.mengambilTelur(dino: ov);
    }
}

```

Output Program :

```

-----Dinosaur -----
TyrannosaurusRex
Jenis Karnivora
Berkembang Biak dengan Bertelur
TyrannosaurusRex Makanan Daging
Berat6,4 ton
Berjalan Dengan 2 Kaki Belakang
-----Dinosaur -----
Triceratops
Jenis Herbivora
Berkembang Biak dengan Bertelur
Triceratops Makanan Tumbuhan
Berat12 ton
Berjalan Dengan 4 Kaki
-----Dinosaur -----
Oviraptor
Karnivora
Berkembang Biak Dengan Bertelur
Oviraptor Makan Telur Dinosaur
Berat36 kg
Berjalan Dengan 2 Kaki Belakang

```

```
----Pemburu Dan Anak Pemburu----
Pemburu Sedang Mengawasi Telur Dinosaur
Pemburu Harus Berhati-hati Dengan
TyrannosaurusRex
=====
Anak Pemburu Memantau Telur Dinosaur!!
TyrannosaurusRex Makanan Daging
=====
Anak Pemburu Memantau Telur Dinosaur!!
Triceratops Makanan Tumbuhan
=====
Anak Pemburu Memantau Telur Dinosaur!!
Oviraptor Makan Telur Dinosaur
-----
BUILD SUCCESS
-----
```