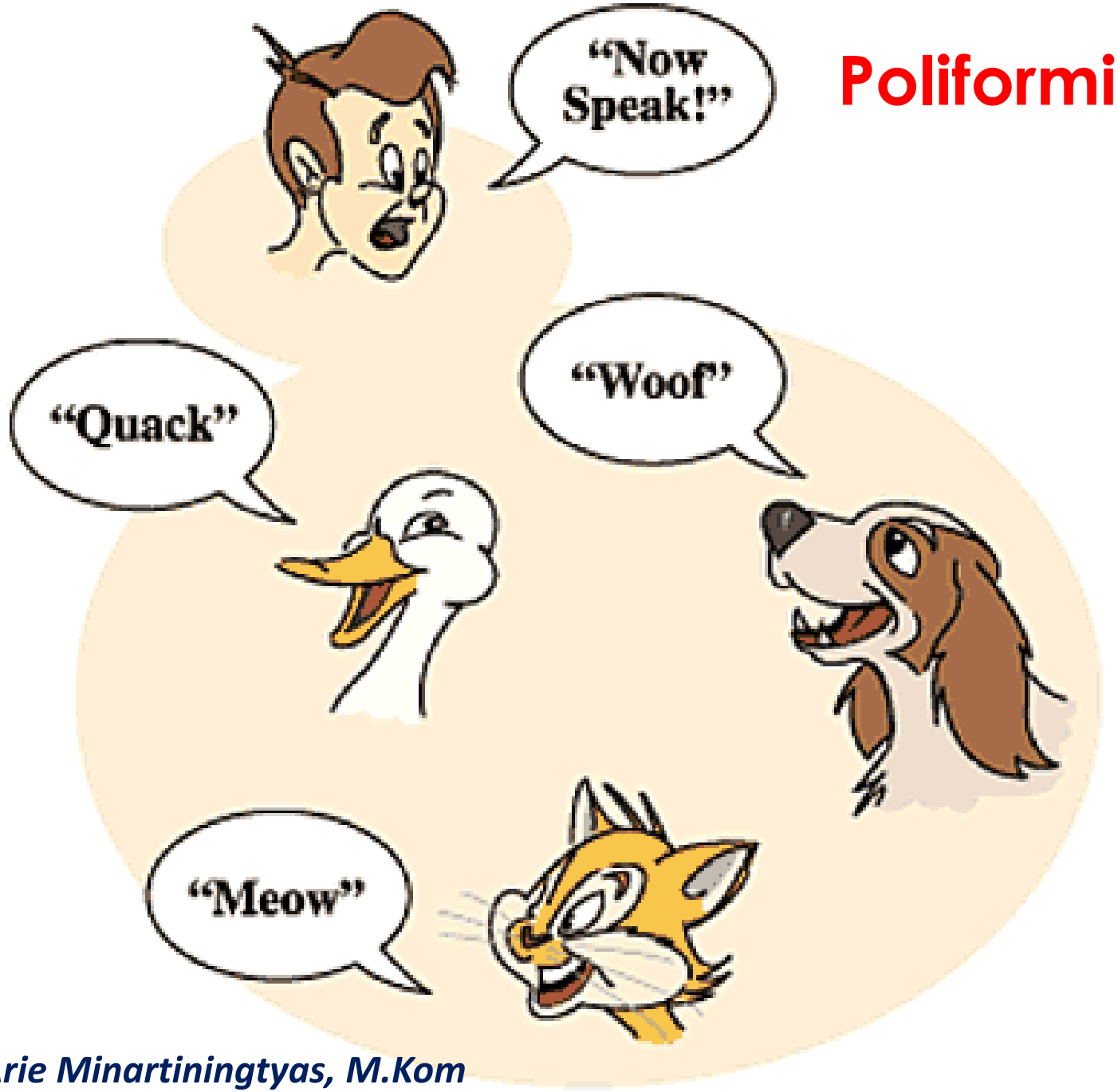
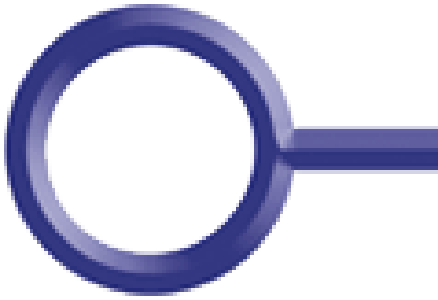
Poliformisme

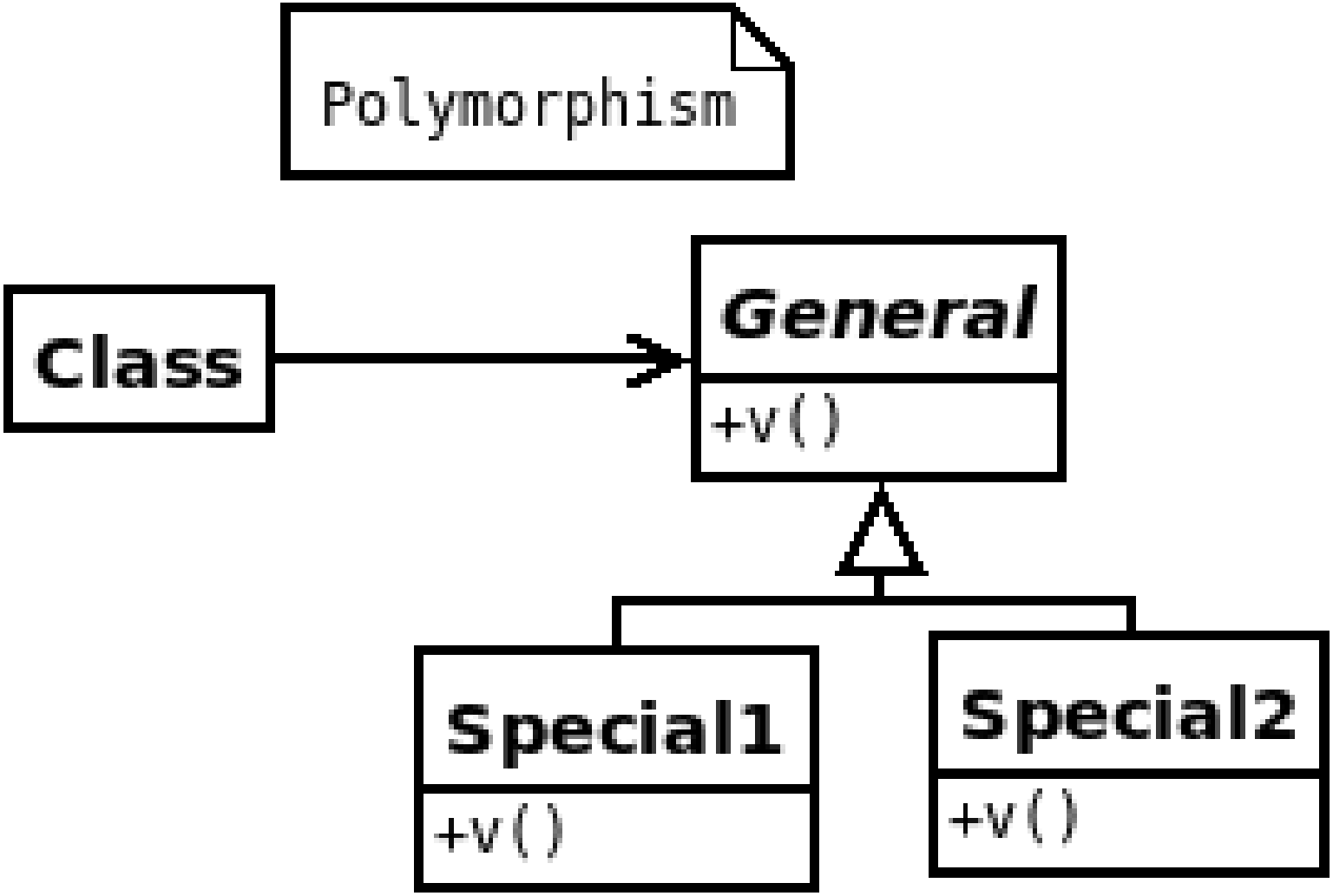


Brigida Arie Minartiningtyas, M.Kom

polymorphism



Kemampuan suatu objek untuk mengungkap banyak hal melalui satu cara yang sama.



Polimorfisme mengijinkan kelas induk untuk mendefinisikan sebuah method general (bersifat umum) untuk semua kelas turunannya, dan selanjutnya kelas-kelas turunan dapat meperbarui implementasi dari method tersebut secara lebih spesifik sesuai dengan karakteristiknya masing-masing

```java
public class Penyanyi {
    public void bernyanyi(){
        System.out.println ("Karakteristik nadanya belum didefinisikan");
    }
}
```

```java
public class PenyanyiJazz extends Penyanyi {
    public void bernyanyi(){
        System.out.println("Bernyanyi dengan irama Jazz");
    }
}
```

```java
public class PenyanyiPop extends Penyanyi {
    public void bernyanyi (){
        System.out.println("Bernyanyi dengan irama Pop");
    }
}
```

```java
public class DemoPolimorfisme {
    public static void main (String [] args){
        Penyanyi p;

        PenyanyiJazz andien = new PenyanyiJazz();
        PenyanyiPop rossa = new PenyanyiPop();

        p = andien;
        p.bernyanyi();

        p = rossa;
        p.bernyanyi();
    }
}
```

DemoPolimorfisme > main >

tput - Pertemuan9 (run)

```
run:
Bernyanyi dengan irama Jazz
Bernyanyi dengan irama Pop
BUILD SUCCESSFUL (total time: 2 seconds)
```

```java
public class Bentuk {
    public double luas(){
        System.out.println("Tidak didefinisikan");
        return 0;
    }
}
```

```java
11  public class PersegiPanjang extends Bentuk {
12      private double panjang;
13      private double lebar;
14
15      PersegiPanjang (int p, int l){
16          panjang = p;
17          lebar = l;
18      }
19
        public double luas(){
21          System.out.println("Dalam objek persegi panjang : ");
22          return (panjang*lebar);
23      }
24  }
```

```java
public class DemoPolimorfisme {
    public static void main (String [] args){
        Penyanyi p;

        PenyanyiJazz andien = new PenyanyiJazz();
        PenyanyiPop rossa = new PenyanyiPop();

        p = andien;
        p.bernyanyi();

        p = rossa;
        p.bernyanyi();
    }
}
```

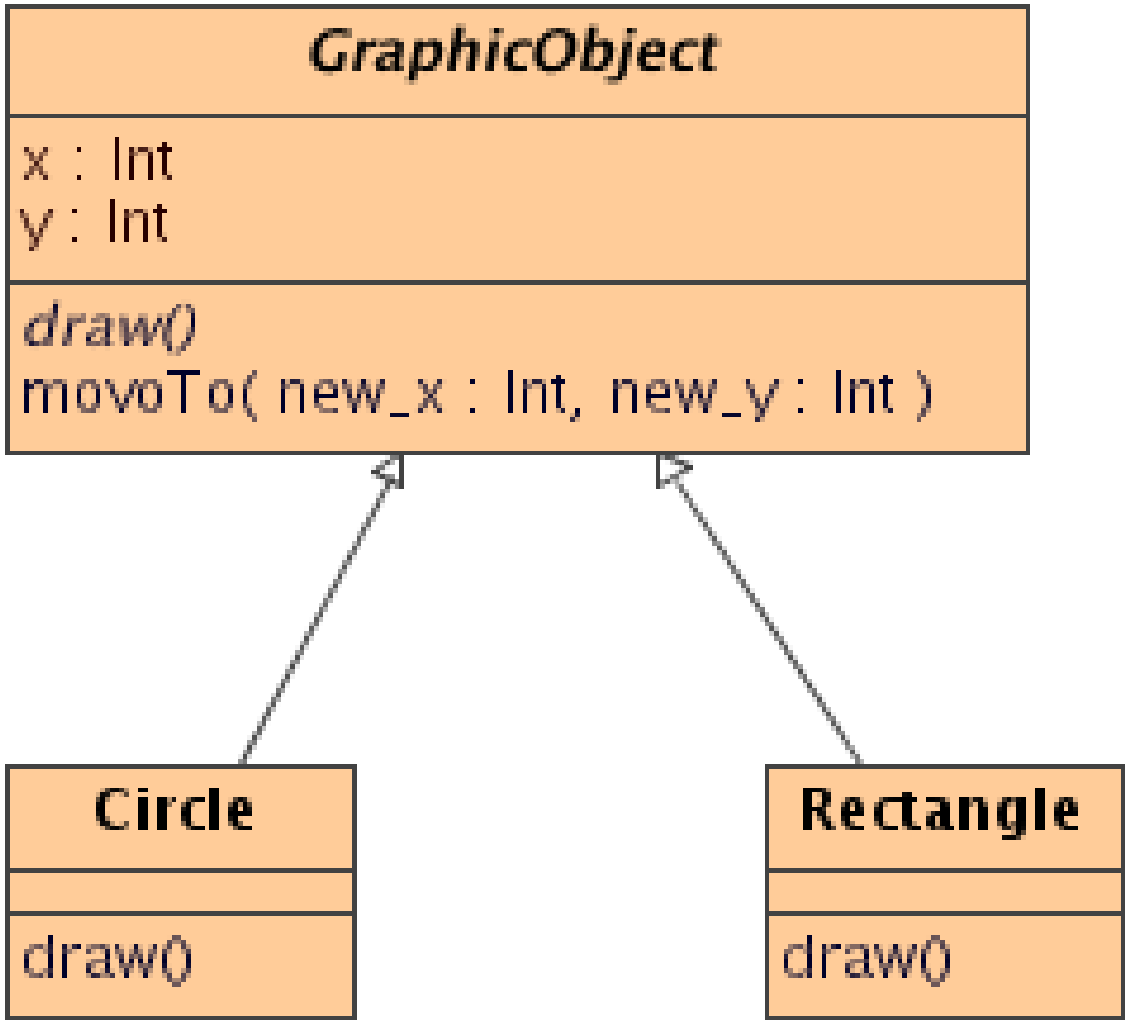
DemoPolimorfisme > main >

put - Pertemuan9 (run)

```
run:
Bernyanyi dengan irama Jazz
Bernyanyi dengan irama Pop
BUILD SUCCESSFUL (total time: 1 second)
```

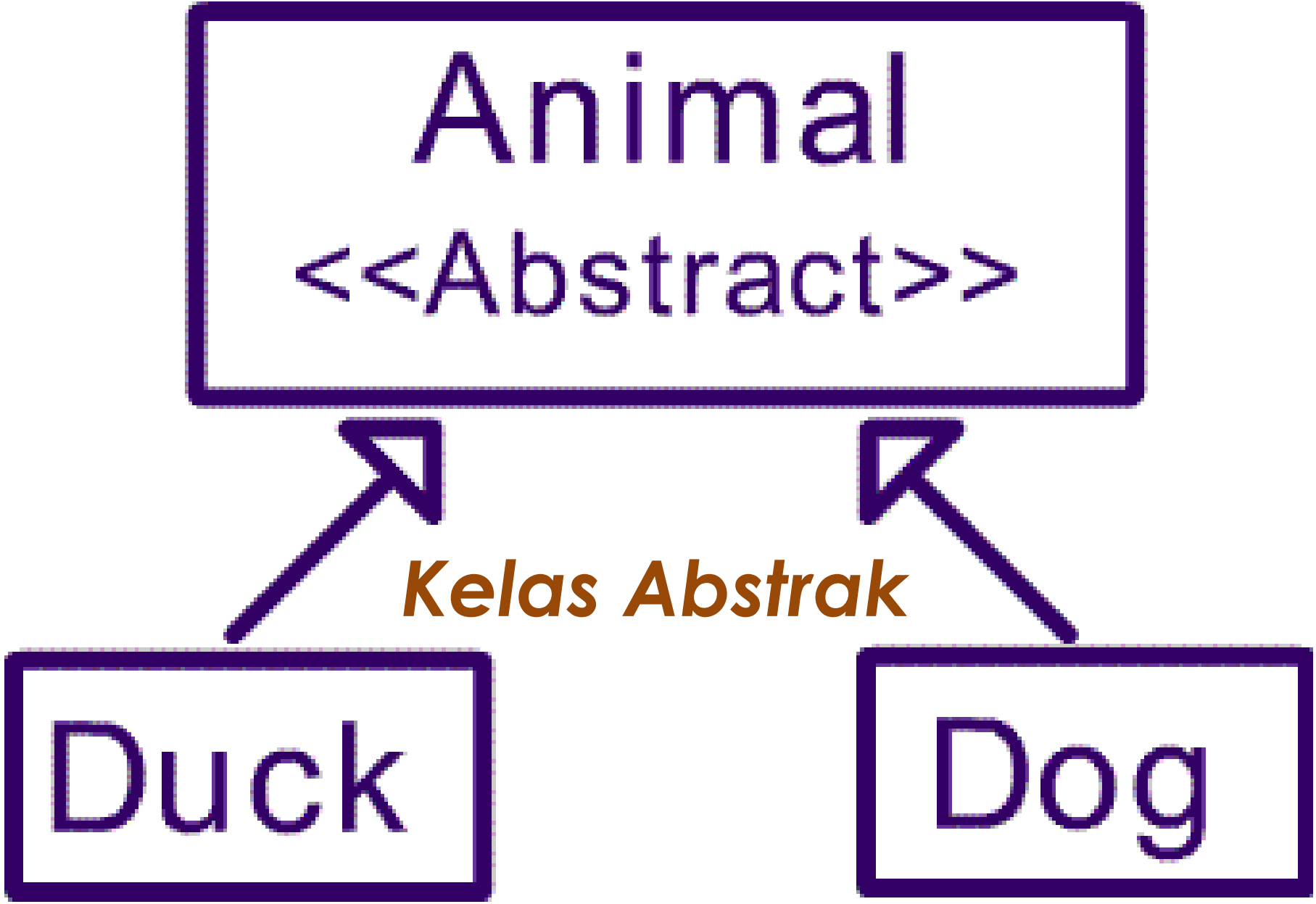
Kelas Abstrak

```
abstract class nama-kelas{
        badan kelas
}
```



Method Abstract

abstract tipe namaMethod (daftar-parameter)



Pendefinisian kelas induk yang di dalamnya terdapat deklarasi method yang tidak memerlukan implementasi sama sekali

```java
public abstract class A {
    abstract public void coba();

    public void cobajuga(){
        System.out.println("Method non abstrak di dalam kelas abstrak");
    }
}
```

```java
public class B extends A {
    public void coba(){
        System.out.println("Method di dalam kelas B");
    }
}
```

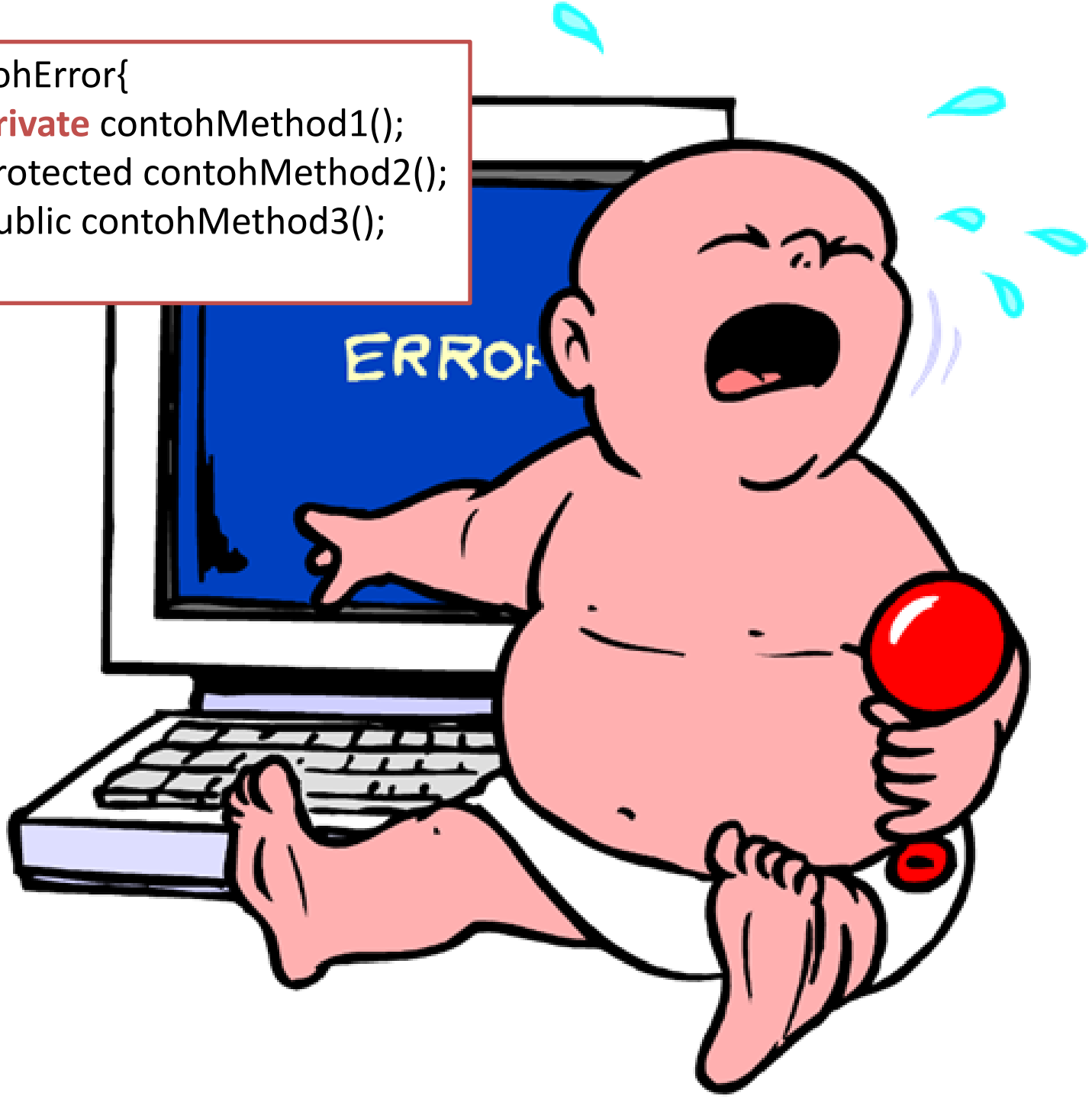
```java
public class DemoAbstract {
    public static void main (String [] args){
        A ref;

        B obj = new B();

        ref = obj;

        ref.coba();
        ref.cobajuga();
    }
}
```

DemoAbstract > main >

tput - Pertemuan9 (run)

```
run:
Method di dalam kelas B
Method non abstrak di dalam kelas abstrak
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
abstract class ContohError{
        abstract private contohMethod1();
        abstract protected contohMethod2();
        abstract public contohMethod3();
}
```



Mengenal Kata Kunci Final

Fungsi kata kunci Final:

1. **Apabila digunakan untuk mendeklarasikan variabel, maka nilai dari variabel tersebut tidak dapat diubah atau diperankan sebagai sebuah konstanta**

Bentuk umum deklarasi konstanta dalam Java

Tingkat-akses final tipe-data nama-konstanta = nilai;

```java
public class stack {
    private final int MAX = 10;
    private int [] stack = new int [MAX];
    private int atas;

    stack(){
        atas = -1;
    }

    void push (int nilai){
        if (atas!=9){
            stack [++atas] = nilai;
        }else{
            System.out.println ("Stack Penuh");
        }
    }

    int pop (){
        if (atas>=0){
            return stack [atas--];
        }else{
            System.out.println("Stack masih kosong...");
            return 0;
        }
    }
}
```

konstanta

```java
public class DemoFinal {
    public static void main (String[] args){
        stack s = new stack();

        for (int i =1;i<=10;i++){
            s.push(i*10);
        }

        for (int i=1; i<=10;i++){
            System.out.println(s.pop());
        }
    }
}
```

DemoFinal > main > for (int i = 1; i <= 10; i++)

put - Pertemuan7 (run)

```
run:
100
90
80
70
60
50
40
30
20
10
BUILD SUCCESSFUL (total time: 1 second)
```

Apabila digunakan untuk mendeklarasikan method, maka method tersebut sudah tidak dapat di override atau diperbarui lagi oleh kelas-kelas turunannya

```
public class Induk {
    public final void coba(){
        System.out.println ("Contoh method final");
    }
}
```

```
11  public class Turunan extends Induk {
13      public void coba(){
            System.out.println ("Tidak diizinkan...");
14      }
15  }
```

coba() in Turunan cannot override coba() in Induk
overridden method is final

Add @Override Annotation

(Alt-Enter shows hints)

Apabila digunakan untuk mendefinisikan kelas, maka kelas tersebut sudah tidak dapat diturunkan lagi menjadi kelas baru lainnya

```
Class induk{
        badan kelas induk
}

Class turunan extends induk{
        badan kelas turunan
}

Final class turunan2 extends turunan{
        badan kelas turunan2
}

Class turunan3 extends turunan2{
        badan kelas turunan3
}
```