

Nama : La Ode Muhammad Gazali
NIM : 222212696
Kelas : 2KS2

MODUL 3 PRAKTIKUM PEMROGRAMAN BERBASIS OBJEK

A. Inheritance

- Orang.java

```
1  package praktikum3;
2
3  import java.util.Date;
4
5  /**
6   *
7   * @author U53R
8   */
9  public class Orang {
10     private String nama;
11     private Date tanggalLahir;
12     private String gaji;
13
14     public Orang() {
15     }
16
17     public Orang(String nama) {
18         this.nama = nama;
19     }
20
21     public Orang(String nama, Date tanggalLahir) {
22         this.nama = nama;
23         this.tanggalLahir = tanggalLahir;
24     }
25
26     public String getNama() {
27         return nama;
28     }
29
30     public void setNama(String nama) {
31         this.nama = nama;
32     }
33
34     public Date getTanggalLahir() {
35         return tanggalLahir;
36     }
37
38     public void setTanggalLahir(Date tanggalLahir) {
39         this.tanggalLahir = tanggalLahir;
40     }
41 }
```

- Pegawai.java

```
1 package praktikum3;
2
3 import java.util.Date;
4
5
6 /**
7  *
8  * @author U53R
9  */
10 public class Pegawai extends Orang {
11     private String NIP;
12     private String namaKantor;
13     private String unitKerja;
14     private String gaji;
15
16     public Pegawai() {
17     }
18
19     public Pegawai(String NIP, String namaKantor, String unitKerja) {
20         this.NIP = NIP;
21         this.namaKantor = namaKantor;
22         this.unitKerja = unitKerja;
23     }
24
25     public Pegawai(String NIP, String namaKantor, String unitKerja, String nama, Date tanggalLahir) {
26         super(nama, tanggalLahir);
27         this.NIP = NIP;
28         this.namaKantor = namaKantor;
29         this.unitKerja = unitKerja;
30     }
31
32     public String getNIP() {
33         return NIP;
34     }
35
36     public void setNIP(String NIP) {
37         this.NIP = NIP;
38     }
39
40     public String getNamaKantor() {
41         return namaKantor;
42     }
43
44     public void setNamaKantor(String namaKantor) {
45         this.namaKantor = namaKantor;
46     }
47
48     public String getUnitKerja() {
49         return unitKerja;
50     }
51
52     public void setUnitKerja(String unitKerja) {
53         this.unitKerja = unitKerja;
54     }
55 }
```

- Kantor.java (Kelas Main)

```
1 package praktikum3;
2
3 import java.util.ArrayList;
4 import java.util.Date;
5 import java.util.List;
6
7 /**
8  *
9  * @author U53R
10  */
11 public class Kantor {
12     public static void main(String[] args) {
13         Orang lutfi = new Orang();
14         lutfi.setNama("Lutfi");
15         lutfi.setTanggalLahir(new Date(2001, 1, 20));
16
17         Orang rahma = new Orang("Rahma");
18         rahma.setTanggalLahir(new Date(1997, 8, 31));
19
20         Pegawai tuti = new Pegawai("6836492379", "STIS", "IT", "Tuti", new Date(1997, 8, 2));
21
22         System.out.println("Ada orang:");
23         System.out.println(lutfi.getNama()+" lahir pada "+lutfi.getTanggalLahir());
24         System.out.println(rahma.getNama()+" lahir pada "+rahma.getTanggalLahir());
25         System.out.println(tuti.getNama()+
26             " lahir pada "+tuti.getTanggalLahir()+
27             " NIP: "+tuti.getNIP()+
28             " kantor: "+tuti.getNamaKantor()+
29             " bagian: "+tuti.getUnitKerja()
30         );
31     }
32 }
```

- Hasil Running

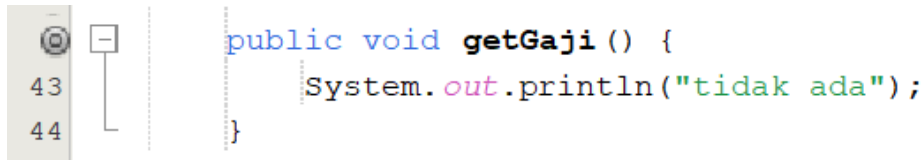
```
run:
Ada orang:
Lutfi lahir pada Wed Feb 20 00:00:00 ICT 3901
Rahma lahir pada Fri Oct 01 00:00:00 ICT 3897
Tuti lahir pada Thu Sep 02 00:00:00 ICT 3897 NIP: 6836492379 kantor: STIS bagian: IT
BUILD SUCCESSFUL (total time: 1 second)
```

Program tersebut merupakan program inheritance, dimana kelas orang memiliki turunan kelas pegawai. Pada main program, dibuat 3 buah objek, antara lain 2 buah objek orang dan 1 objek pegawai. Hasil running program dapat dilihat pada compiler diatas.

B. Polymorphism

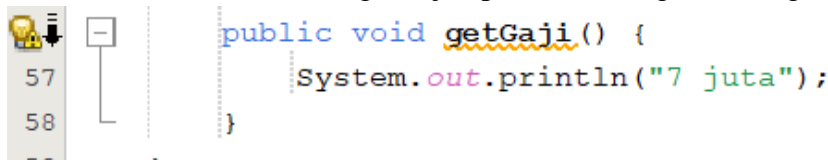
Polymorphism memungkinkan suatu objek memiliki banyak bentuk atau perilaku. Pada program sebelumnya, akan ditambahkan variabel gaji pada kelas Orang dan kelas Pegawai, sebagai berikut:

- Menambahkan method `getGaji()` pada kelas Orang, namun outputnya “tidak ada”



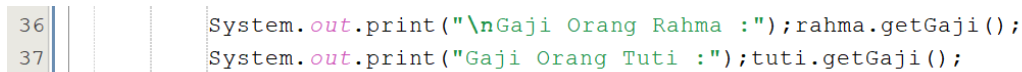
```
43 public void getGaji () {  
44     System.out.println("tidak ada");  
}
```

- Menambahkan method `getGaji()` pada kelas Pegawai dengan gaji sebesar 7 juta



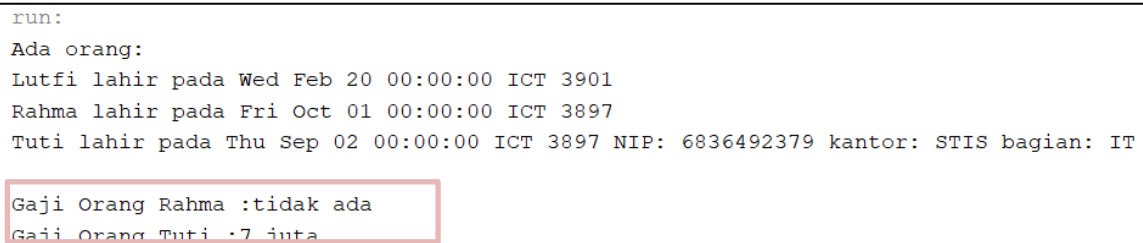
```
57 public void getGaji() {  
58     System.out.println("7 juta");  
}
```

- Memanggil method gaji dari objek Orang dan Pegawai pada kelas Kantor



```
36 System.out.print("\nGaji Orang Rahma :");rahma.getGaji();  
37 System.out.print("Gaji Orang Tuti :");tuti.getGaji();
```

• Hasil running



```
run:  
Ada orang:  
Lutfi lahir pada Wed Feb 20 00:00:00 ICT 3901  
Rahma lahir pada Fri Oct 01 00:00:00 ICT 3897  
Tuti lahir pada Thu Sep 02 00:00:00 ICT 3897 NIP: 6836492379 kantor: STIS bagian: IT  
  
Gaji Orang Rahma :tidak ada  
Gaji Orang Tuti :7 juta
```

C. Aggregation

Dalam kasus program diatas, misalkan di suatu Unit Kerja yang mempunyai nama di kantor, ada beberapa pegawai yang bekerja. Kita bisa membuat kelas Unit Kerja dan menambahkan Pegawai sebagai bagian dari Unit Kerja tersebut.

- UnitKerja.java

```
1  package praktikum3;
2
3  import java.util.List;
4
5  /**
6   *
7   * @author U53R
8   */
9  public class UnitKerja {
10     private String nama;
11     private List<Pegawai> daftarPegawai;
12
13     public UnitKerja(String nama, List<Pegawai> daftarPegawai) {
14         this.nama = nama;
15         this.daftarPegawai = daftarPegawai;
16     }
17
18     public String getNama() {
19         return nama;
20     }
21
22     public List<Pegawai> getDaftarPegawai() {
23         return daftarPegawai;
24     }
25
26     @Override
27     public String toString() {
28         return "UnitKerja{" + "nama=" + nama + ", daftarPegawai=" + daftarPegawai + '}';
29     }
30 }
```

- Memanggil method gaji dari objek Orang dan Pegawai pada kelas Kantor

```
40     List<Pegawai> daftarPegawai = new ArrayList<Pegawai>();
41     daftarPegawai.add(tuti);
42     UnitKerja umum = new UnitKerja("Umum", daftarPegawai);
43     System.out.println("\n" + umum);
```

- UnitKerja.java

```
run:
Ada orang:
Lutfi lahir pada Wed Feb 20 00:00:00 ICT 3901
Rahma lahir pada Fri Oct 01 00:00:00 ICT 3897
Tuti lahir pada Thu Sep 02 00:00:00 ICT 3897 NIP: 6836492379 kantor: STIS bagian: IT

Gaji Orang Rahma :tidak ada
Gaji Orang Tuti :7 juta

UnitKerja{nama=Umum, daftarPegawai=[praktikum3.Pegawai@246b179d]}
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Composition

Komposisi adalah bentuk agregasi terbatas di mana dua entitas sangat bergantung satu sama lain. Dalam kasus program diatas, misalnya Gedung yang terdiri dari ruangan. Setiap Gedung pasti punya ruangan. Jika tidak ada ruangan, maka gedung tidak akan ada.

- Ruang.java

```
1      package praktikum3;
2
3      /**
4       *
5       * @author U53R
6       */
7      public class Ruang {
8          private String namaRuang;
9
10         public Ruang(String namaRuang) {
11             this.namaRuang = namaRuang;
12         }
13
14         public String getNamaRuang() {
15             return namaRuang;
16         }
17
18         public void setNamaRuang(String namaRuang) {
19             this.namaRuang = namaRuang;
20         }
21     }
```

- Gedung.java

```
1      package praktikum3;
2
3      import java.util.ArrayList;
4      import java.util.List;
5
6      /**
7       *
8       * @author U53R
9       */
10     public class Gedung {
11         private List<Ruang> daftarRuang = new ArrayList<Ruang>();
12
13         public Gedung() {
14             Ruang ruang = new Ruang("Utama");
15             daftarRuang.add(ruang);
16         }
17
18         public void addRuang(String namaRuang) {
19             Ruang ruang = new Ruang(namaRuang);
20             daftarRuang.add(ruang);
21         }
22
23         public List<Ruang> getDaftarRuang() {
24             return daftarRuang;
25         }
26     }
```

- MainStis.java

```
1      package praktikum3;
2
3      import java.util.List;
4
5      /**
6       *
7       * @author U53R
8       */
9      public class MainStis {
10         public static void main(String[] args) {
11             Gedung STIS = new Gedung();
12             STIS.addRuang("Lobi");
13             STIS.addRuang("Bagian Umum");
14             STIS.addRuang("Kepala Kantor");
15
16             List<Ruang> ruangan = STIS.getDaftarRuang();
17             for(Ruang ruang : ruangan){
18                 System.out.println("Ruang : "+ruang.getNamaRuang());
19             }
20         }
21     }
```

- Hasil running

```
run:
Ruang : Utama
Ruang : Lobi
Ruang : Bagian Umum
Ruang : Kepala Kantor
BUILD SUCCESSFUL (total time: 0 seconds)
```