**LAPORAN PRAKTKUM**

**INHERITANCE, SUPER, DAN OVERRIDING**

Dibuat untuk memenuhi salah satu tugas mata kuliah Pemrograman Berbasis

Objek yang diampu oleh Bapak Ardhian Ekawijana



**Disusun Oleh:**

NIM : 241511094

NAMA : ZAHRA ALDILA

KELAS : 2C

PROGRAM STUDI : D3 – TEKNIK INFORMATIKA

JURUSAN : TEKNIK KOMPUTER DAN INFORMATIKA

**Program Studi D-3 Teknik Informatika**

**Jurusan Teknik Komputer dan Informatika Politeknik Negeri Bandung**

**2025**

# DAFTAR ISI

[DAFTAR ISI i](#_Toc208993856)

[DAFTAR GAMBAR ii](#_Toc208993857)

[A. Pendahuluan 1](#_Toc208993858)

[B. Latihan Modul 1](#_Toc208993859)

[C. Kasus Koperasi 4](#_Toc208993860)

[D. Kesimpulan 7](#_Toc208993861)

# DAFTAR GAMBAR

[Gambar 1.1. Output Task 1.1 1](#_Toc208993845)

[Gambar 1.2. Output Task 1.2 2](#_Toc208993846)

[Gambar 1.3. Output Task 1.3 2](#_Toc208993847)

[Gambar 1.4. Output Task 2.1 Circle 3](#_Toc208993848)

[Gambar 1.5. Output Task 2.1 Rectangle 3](#_Toc208993849)

[Gambar 1.6. Output Task 2.1 Square 3](#_Toc208993850)

[Gambar 1.7. Daftar Barang 4](#_Toc208993851)

[Gambar 1.8. Daftar Makanan (Subclass Barang) 5](#_Toc208993852)

[Gambar 1.9. Daftar Elektronik (Subclass Barang) 5](#_Toc208993853)

[Gambar 1.10. Output transaksi Makanan (overriding getHargaAkhir dengan diskon) 6](#_Toc208993854)

[Gambar 1.11. Output transaksi Elektronik (overriding getHargaAkhir dengan biaya tambahan) 7](#_Toc208993855)

## Pendahuluan

Praktikum ini membahas penerapan konsep inheritance, super, dan overriding dalam bahasa Java.

Latihan dibagi menjadi dua bagian:

1. Menyelesaikan Task 1.1 – 2.1 dari modul (Circle, Cylinder, Shape, Rectangle, Square).
2. Menerapkan konsep yang sama pada kasus program koperasi dengan menambahkan subclass Makanan dan Elektronik dari class Barang.

Link Github:

<https://github.com/zahraldila/TugasPBOWeek4.git>

## Latihan Modul

1. Task 1.1

Menambahkan constructor baru dan getter/setter untuk atribut color.

|  |
| --- |
| public Circle(double r, String c) {  radius = r;  color = c;  }  public String getColor() { return color; }  public void setColor(String c) { this.color = c; } |

Ouput

A black screen with white text

AI-generated content may be incorrect.

Gambar 1.1. Output Task 1.1

1. Task 1.2

getArea() dioverride untuk menghitung luas permukaan silinder. Agar getVolume() tetap benar, method ini memakai super.getArea() untuk menghitung luas alas.

|  |
| --- |
| @Override  public double getArea() {  return 2 \* Math.PI \* getRadius() \* height + 2 \* super.getArea();  }  public double getVolume() {  return super.getArea() \* height;  } |

Output

A screenshot of a computer screen

AI-generated content may be incorrect.

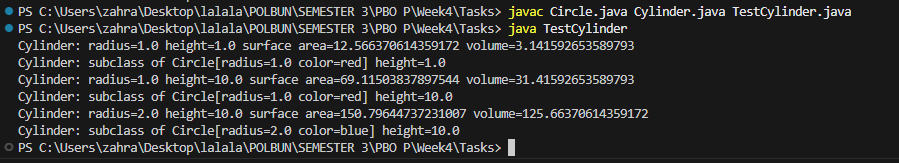
Gambar 1.2. Output Task 1.2

1. Task 1.3

Menambahkan method toString() di Cylinder yang menggunakan super.toString() dari Circle.

|  |
| --- |
| @Override  public String toString() {  return "Cylinder: subclass of " + super.toString() + " height=" + height;  } |

Output



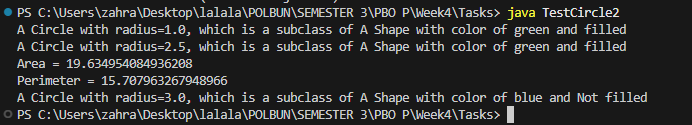
Gambar 1.3. Output Task 1.3

1. Task 2.1

Membuat superclass Shape dan subclass Circle, Rectangle, serta Square.  
Square meng-override setWidth dan setLength agar selalu konsisten, serta override toString().

|  |
| --- |
| @Override  public void setWidth(double side) { setSide(side); }  @Override  public void setLength(double side) { setSide(side); }  @Override  public String toString() {  return "A Square with side=" + getSide()  + ", which is a subclass of " + super.toString();  } |

Output



Gambar 1.4. Output Task 2.1 Circle

A screen shot of a computer screen

AI-generated content may be incorrect.

Gambar 1.5. Output Task 2.1 Rectangle

A screen shot of a computer program

AI-generated content may be incorrect.

Gambar 1.6. Output Task 2.1 Square

## Kasus Koperasi

Setelah menyelesaikan latihan pada modul, konsep inheritance, super, dan overriding kemudian diterapkan pada studi kasus program koperasi. Pada kasus ini, class Barang dijadikan sebagai superclass, sementara class Makanan dan Elektronik dibuat sebagai subclass yang mewarisi atribut dan method dari Barang. Melalui pewarisan ini, perilaku perhitungan harga dapat disesuaikan di masing-masing subclass dengan memanfaatkan overriding dan pemanggilan super. Dengan demikian, program koperasi menjadi lebih fleksibel dalam mengatur aturan harga berdasarkan jenis barang.

1. Inheritance

Inheritance diterapkan dengan menjadikan Barang sebagai kelas induk (superclass). Kelas Makanan dan Elektronik dibuat sebagai turunan (subclass) yang mewarisi atribut serta method dari Barang. Kedua subclass tersebut kemudian menyesuaikan perilaku perhitungan harga dengan melakukan overriding sesuai kebutuhan masing-masing.

|  |
| --- |
| // Subclass Makanan mewarisi superclass Barang  public class Makanan extends Barang {  ...  }  // Subclass Elektronik mewarisi superclass Barang  public class Elektronik extends Barang {  ...  } |

Output



Gambar 1.7. Daftar Barang



Gambar 1.8. Daftar Makanan (Subclass Barang)



Gambar 1.9. Daftar Elektronik (Subclass Barang)

1. Super

Kata kunci super digunakan dalam konstruktor subclass untuk memanggil konstruktor induk (super(idBarang, namaBarang, harga, stok)). Selain itu, super juga dipakai dalam method overriding, misalnya super.getHargaAkhir(jumlah), agar logika dasar dari kelas induk tetap bisa dimanfaatkan sebelum ditambahkan aturan khusus di subclass.

|  |
| --- |
| public Makanan(String id, String nama, double harga, int stok, int h) {  super(id, nama, harga, stok); // memanggil konstruktor induk  this.hariMenujuExpired = h;  } |

1. Overriding

Overriding dilakukan ketika subclass mengubah perilaku method dari superclass. Pada program ini, Makanan meng-override method getHargaAkhir() agar harga disesuaikan dengan diskon jika barang mendekati kadaluarsa. Sementara Elektronik meng-override method yang sama untuk menambahkan biaya layanan tetap.

1. Makanan

Makanan override method getHargaAkhir() supaya barang mendekati expired mendapat diskon.

|  |
| --- |
| @Override  public double getHargaAkhir(int jumlah) {  double totalNormal = super.getHargaAkhir(jumlah);  if (hariMenujuExpired <= 2) return totalNormal \* 0.8; // diskon 20%  else if (hariMenujuExpired <= 5) return totalNormal \* 0.9; // diskon 10%  return totalNormal;  } |

Output



Gambar 1.10. Output transaksi Makanan (overriding getHargaAkhir dengan diskon)

1. Elektronik

Elektronik override method getHargaAkhir() untuk menambahkan biaya layanan tetap.

|  |
| --- |
| @Override  public double getHargaAkhir(int jumlah) {  double total = super.getHargaAkhir(jumlah);  return total + 5000; // biaya tambahan  } |

Output



Gambar 1.11. Output transaksi Elektronik (overriding getHargaAkhir dengan biaya tambahan)

## Kesimpulan

Dari praktikum ini dapat disimpulkan bahwa penerapan konsep inheritance, super, dan overriding sangat membantu dalam pengembangan program berorientasi objek. Melalui inheritance, atribut dan method dari kelas induk dapat digunakan kembali oleh kelas turunan sehingga kode menjadi lebih ringkas dan mudah diperluas. Kata kunci super digunakan baik untuk memanggil konstruktor induk maupun method induk, sehingga logika dasar tetap dipertahankan saat subclass menambahkan perilaku baru. Dengan overriding, setiap subclass mampu menyesuaikan perilaku method sesuai kebutuhan, seperti pada Makanan yang memberikan diskon dan Elektronik yang menambahkan biaya layanan. Polimorfisme juga terlihat saat variabel bertipe superclass memanggil method yang telah dioverride di subclass, sehingga perilaku program dapat berbeda sesuai jenis objek sebenarnya. Secara keseluruhan, konsep-konsep ini membuat program lebih fleksibel, mudah dipelihara, dan siap untuk dikembangkan lebih lanjut.