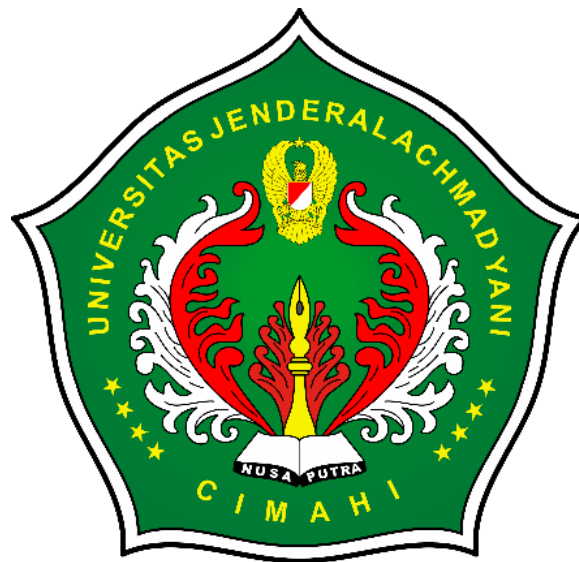


**LAPORAN PRAKTIKUM  
PEMROGRAMAN OBJEK 2**

**MODUL 1  
REVIEW PEMROGRAMAN OBJEK 1**

**DISUSUN OLEH :  
Fathir Ahmad Nurpadli - 2250081132**



**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS DAN INFORMATIKA  
UNIVERSITAS JENDERAL ACHMAD YANI  
TAHUN 2024**



# DAFTAR ISI

DAFTAR ISI.....	i
DAFTAR GAMBAR.....	vii
BAB I. HASIL PRAKTIKUM.....	1
I.1 Program I-1 Lingkaran.java .....	1
I.1.A. Source Code .....	1
I.1.B. Hasil .....	1
I.1.C. Analisa .....	2
I.2 Program I-2 AritmatikaDemo.java.....	2
I.2.A. Source Code .....	2
I.2.B. Hasil .....	5
I.2.C. Analisa .....	6
I.3 Program I-3 Operator.java.....	7
I.3.A. Source Code .....	7
I.3.B. Hasil .....	7
I.3.C. Analisa .....	7
I.4 Program I-5 TestAND.java .....	8
I.4.A. Source Code .....	8
I.4.B. Hasil .....	9
I.4.C. Analisa .....	9
I.5 Program I-6 TestOR.java .....	9
I.5.A. Source Code .....	9
I.5.B. Hasil .....	10
I.5.C. Analisa .....	10
I.6 Program I-7 Person.java.....	11
I.6.A. Source Code .....	11

I.6.B.	Hasil .....	12
I.6.C.	Analisa .....	12
I.7	Program I-8 AlbumLagu.java .....	12
I.7.A.	Source Code .....	12
I.7.B.	Hasil .....	14
I.7.C.	Analisa .....	14
I.8	Program I-9 Person.java.....	15
I.8.A.	Source Code .....	15
I.8.B.	Hasil .....	16
I.8.C.	Analisa .....	16
I.9	Program I-10a Kucing.java .....	16
I.9.A.	Source Code .....	16
I.9.B.	Hasil .....	18
I.9.C.	Analisa .....	18
I.10	Program I-10b LingkunganRumah.java.....	18
I.10.A.	Source Code .....	18
I.10.B.	Hasil .....	19
I.10.C.	Analisa.....	20
I.11	Program I-11 Lion.java .....	20
I.11.A.	Source Code .....	20
I.11.B.	Hasil .....	21
I.11.C.	Analisa.....	21
I.12	Program I-12 Horse.java .....	21
I.12.A.	Source Code .....	21
I.12.B.	Hasil .....	22
I.12.C.	Analisa.....	23
I.13	Program I-13 Kangaroo.java.....	23

I.13.A.	Source Code .....	23
I.13.B.	Hasil .....	24
I.13.C.	Analisa.....	24
I.14	Program I-14 Zoo.java .....	24
I.14.A.	Source Code .....	24
I.14.B.	Hasil .....	25
I.14.C.	Analisa.....	26
I.15	Program I-15 Dog.java.....	26
I.15.A.	Source Code .....	26
I.15.B.	Hasil .....	27
I.15.C.	Analisa.....	27
I.16	Program I-16 Elevator.java .....	27
I.16.A.	Source Code .....	27
I.16.B.	Hasil .....	29
I.16.C.	Analisa.....	30
I.17	Program I-17 ElevatorTest.java .....	30
I.17.A.	Source Code .....	30
I.17.B.	Hasil .....	31
I.17.C.	Analisa.....	31
I.18	Program I-18 ElevatorTestTwo.java .....	32
I.18.A.	Source Code .....	32
I.18.B.	Hasil .....	32
I.18.C.	Analisa.....	33
I.19	Program I-19 StudentRecord.java .....	33
I.19.A.	Source Code .....	33
I.19.B.	Hasil .....	38
I.19.C.	Analisa.....	38

I.20	Program I-20 StudentRecordExample.java.....	39
I.20.A.	Source Code .....	39
I.20.B.	Hasil .....	40
I.20.C.	Analisa.....	40
I.21	Program I-21 Person.java.....	41
I.21.A.	Source Code .....	41
I.21.B.	Hasil .....	42
I.21.C.	Analisa.....	42
I.22	Program I-22 Student.java.....	43
I.22.A.	Source Code .....	43
I.22.B.	Hasil .....	44
I.22.C.	Analisa.....	44
I.23	Program I-23 Pakaian.java .....	44
I.23.A.	Source Code .....	44
I.23.B.	Hasil .....	46
I.23.C.	Analisa.....	47
I.24	Program I-24 Polimorfisme.....	47
I.24.A.	Source Code .....	47
I.24.B.	Hasil .....	50
I.24.C.	Analisa.....	50
I.25	Program I-25 PrivateElevator2.....	51
I.25.A.	Source Code .....	51
I.25.B.	Hasil .....	55
I.25.C.	Analisa.....	55
I.26	Program I-26 Abstract Class .....	56
I.26.A.	Source Code .....	56
I.26.B.	Hasil .....	57

I.26.C.	Analisa.....	57
I.27	Program I-27 Interface .....	58
I.27.A.	Source Code .....	58
I.27.B.	Hasil .....	59
I.27.C.	Analisa.....	60
<b>BAB II.</b>	<b>TUGAS PRAKTIKUM .....</b>	<b>61</b>
II.1	Tugas I-1 .....	61
II.2	Tugas I-2 .....	61
II.3	Tugas I-5 .....	61
II.4	Tugas I-6 .....	61
II.5	Tugas I-7 .....	62
II.6	Tugas I-8 KelasPB.java.....	62
II.6.A.	Source Code .....	62
II.6.B.	Hasil .....	66
II.6.C.	Analisa.....	66
II.7	Tugas I-9 TestPerson.java .....	66
II.7.A.	Source Code .....	66
II.7.B.	Hasil .....	67
II.7.C.	Analisa.....	68
II.8	Tugas I-10b .....	68
II.8.A.	Source Code .....	68
II.8.B.	Hasil .....	69
II.8.C.	Analisa.....	69
II.9	Tugas I-11 .....	69
II.10	Tugas I-13.....	70
II.10.A.	Source Code .....	70
II.10.B.	Hasil .....	74

II.10.C.	Analisa.....	75
II.11	Tugas I-14.....	75
II.11.A.	Source Code .....	75
II.11.B.	Hasil .....	77
II.11.C.	Analisa.....	78
II.12	Tugas I-15.....	78
II.12.A.	Source Code .....	78
II.12.B.	Hasil .....	80
II.12.C.	Analisa.....	80
II.13	Tugas Akhir ComputerStudentRecord.java.....	81
II.13.A.	Source Code .....	81
II.13.B.	Hasil .....	82
II.13.C.	Analisa.....	82
II.14	Tugas Akhir Abstract Class Shape .....	83
II.14.A.	Source Code .....	83
II.14.B.	Hasil .....	85
II.14.C.	Analisa.....	85
BAB III.	KESIMPULAN .....	86



## DAFTAR GAMBAR

Gambar 1 Hasil Kompilasi Javadoc pada index.html .....	2
Gambar 2 Output Sebelum di Perbaiki .....	6
Gambar 3 Output Setelah di Perbaiki .....	6
Gambar 4 Output dari Program Operator .....	7
Gambar 5 Output dari Program TestAND .....	9
Gambar 6 Output dari Program TestOR .....	10
Gambar 7 Output dari Program Person.....	12
Gambar 8 Output dari Program AlbumLagu .....	14
Gambar 9 Output dari Program Person.....	16
Gambar 10 Output dari Program Kucing .....	18
Gambar 11 Output dari Program LingkunganRumah.....	19
Gambar 12 Output dari Program Lion .....	21
Gambar 13 Output dari Program Horse.java .....	22
Gambar 14 Output dari Program Kangaroo.....	24
Gambar 15 Output dari Zoo.java .....	25
Gambar 16 Output dari Program Dog.....	27
Gambar 17 Output dari Program Elevator .....	29
Gambar 18 Output dari Program ElevatorTest .....	31
Gambar 19 Output dari Program ElevatorTestTwo .....	33
Gambar 20 Output dari Program StudentRecord.....	38
Gambar 21 Output dari Program StudentRecordExample.....	40
Gambar 22 Output dari Program Person.....	42
Gambar 23 Output dari Program Student .....	44
Gambar 24 Output dari Program Pakaian .....	47
Gambar 25 Output dari Program Polimorfisme .....	50
Gambar 26 Output dari Program PrivateElevator2.....	55
Gambar 27 Output dari Program Abstract Class .....	57
Gambar 28 Output dari Program Interface .....	60
Gambar 29 Output dari Program KelasPB.....	66
Gambar 30 Output dari Program TestPerson.....	67
Gambar 31 Output dari Program LingkuranRumah .....	69

Gambar 32 Output dari Program Elevator .....	74
Gambar 33 Output dari Program ElevatorTest .....	74
Gambar 34 Output dari Program ElevatorTestTwo .....	75
Gambar 35 Output dari Program ElevatorTest .....	78
Gambar 36 Output dari Program ElevatorTestTwo .....	78
Gambar 37 Output dari Program StudentRecord .....	80
Gambar 38 Output dari Program ComputerStudentRecord .....	82
Gambar 39 Output dari Program Abstract Class Shape .....	85

# BAB I. HASIL PRAKTIKUM

## I.1 Program I-1 Lingkaran.java

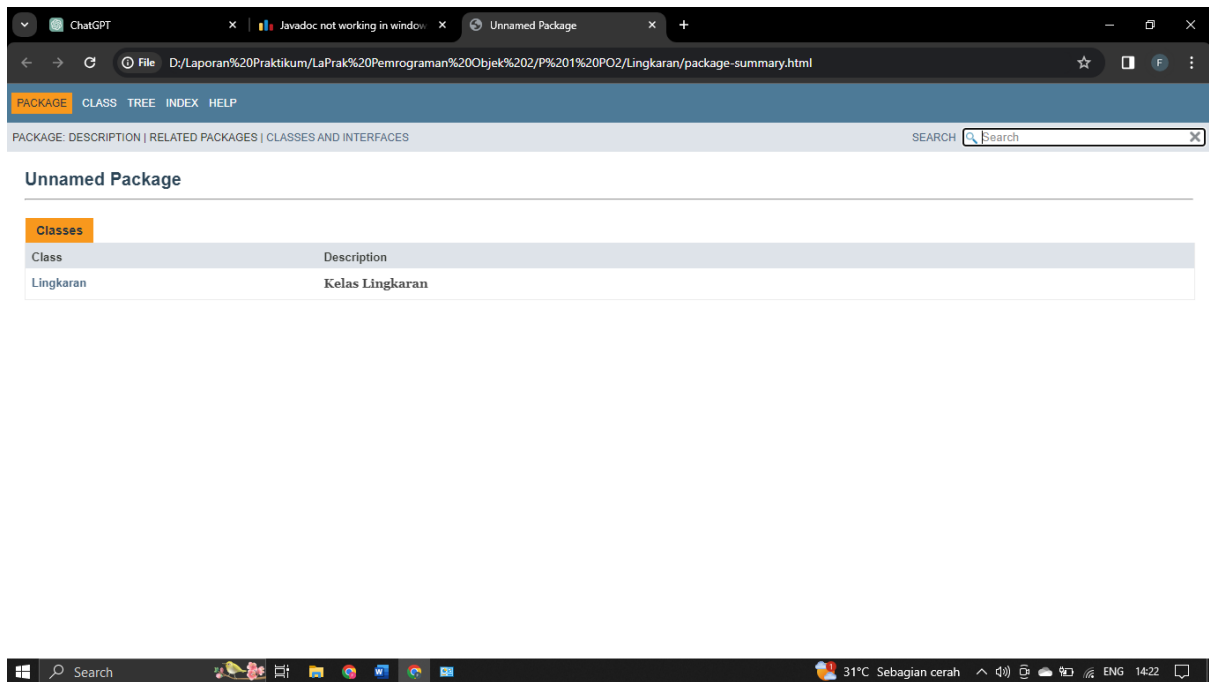
### I.1.A. Source Code

```
/**
 * <h1><b>Kelas Lingkaran</b></h1>
 * kelas ini merepresentasikan Lingkaran sebagai suatu tipe data
 * sebuah bangunan datar sudut,
 * berupa himpunan titik-titik yang berjarak sama ke sebuah titik
pusat
 * Di bawah <i>field</i> yang dimilikinya adalah jejari, yang
menyatakan jarak
 * titik-titik itu ke titik pusat.
 */
public class Lingkaran {
    float jejari;

    Lingkaran() {
        jejari = 0;
    }

    Lingkaran(float r) {
        jejari = r;
    }
}
```

### I.1.B. Hasil



*Gambar 1 Hasil Kompilasi Javadoc pada index.html*

### I.1.C. Analisa

Program di atas adalah sebuah kelas Java yang disebut "Lingkaran" yang merepresentasikan lingkaran sebagai tipe data dalam sebuah bangun datar. Program ini memiliki dua konstruktor, yang satu tanpa parameter dan yang lainnya dengan parameter yang menerima nilai jari.

## I.2 Program I-2 AritmatikaDemo.java

### I.2.A. Source Code

Sebelum

```
public class AritmatikaDemo {
    public static void main(String []args) {
        int i = 37;
        int j = 42;
        double x = 27,475;
```

bernilai 79

```
double y = 7.22;

System.out.println("variable values. . .");

System.out.println("  i = " + i);
System.out.println("  j = " + j);
System.out.println("  x = " + x);
System.out.println("  y = " + y);


System.out.println("Adding. . .");
System.out.println("  i + j = " + i + j); //harus

System.out.println("  x + y = " + x + y);


System.out.println("Subtracting. . .");
System.out.println("  i - j = " + (i - j));
System.out.println("  x - y = " + (x - y));


System.out.println("Multiplying. . .");
System.out.println("  i * j = " + i * j);
System.out.println("  x * y = " + (x * y));


System.out.println("Dividing. . .");
System.out.println("  i / j = " + (i / j));
System.out.println("  x / y = " + x / y);


System.out.println("Computing the remainder. . .");
System.out.println("  i % j = " + i % j);
```

```

        System.out.println("  x % y = " + (x % y));

        System.out.println("Mixing tipes. . .");

        System.out.println("  j + y = " + (j + y)); //Analisa
        System.out.println("  i * x = " + (i * x)); //Analisa
    }
}

```

Sesudah

```

public class AritmatikaDemo {
    public static void main(String []args) {
        int i = 37;
        int j = 42;
        double x = 27.475;
        double y = 7.22;
        System.out.println("variable values. . .");
        System.out.println("  i = " + i);
        System.out.println("  j = " + j);
        System.out.println("  x = " + x);
        System.out.println("  y = " + y);

        System.out.println("Adding. . .");
        System.out.println("  i + j = " + i + j); //harus
        System.out.println("  x + y = " + x + y);

        System.out.println("Subtracting. . .");
    }
}

```

bernilai 79

```

System.out.println(" i - j = " + (i - j));
System.out.println(" x - y = " + (x - y));

System.out.println("Multiplying. . .");
System.out.println(" i * j = " + i * j);
System.out.println(" x * y = " + (x * y));

System.out.println("Dividing. . .");
System.out.println(" i / j = " + (i / j));
System.out.println(" x / y = " + x / y);

System.out.println("Computing the remainder. . .");
System.out.println(" i % j = " + i % j);
System.out.println(" x % y = " + (x % y));

System.out.println("Mixing types. . .");
System.out.println(" j + y = " + (j + y)); //Analisa
System.out.println(" i * x = " + (i * x)); //Analisa
    }
}

```

## **I.2.B. Hasil**

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac AritmatikaDemo.java
AritmatikaDemo.java:5: error: <identifier> expected
    double x = 27,475;
               ^
1 error

D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 2 Output Sebelum di Perbaiki*

```
C:\Windows\System32\cmd.exe
1 error

D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac AritmatikaDemo.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java AritmatikaDemo
variable values. . .
    i = 37
    j = 42
    x = 27.475
    y = 7.22
Adding. . .
    i + j = 3742
    x + y = 27.4757.22
Subtracting. . .
    i - j = -5
    x - y = 20.255000000000003
Multiplying. . .
    i * j = 1554
    x * y = 198.36950000000002
Dividing. . .
    i / j = 0
    x / y = 3.805401662049862
Computing the remainder. . .
    i % j = 37
    x % y = 5.815000000000002
Mixing types. . .
    j + y = 49.22
    i * x = 1016.575

D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 3 Output Setelah di Perbaiki*

### **I.2.C. Analisa**

Program di atas adalah contoh sederhana dari operasi aritmatika dalam bahasa pemrograman Java. Pada awalnya, variabel-variabel integer `i` dan `j` serta variabel double `x` dan `y` dideklarasikan dan diinisialisasi dengan nilai tertentu. Kemudian, program mencetak nilai-nilai dari variabel tersebut. Selanjutnya, program melakukan operasi penjumlahan, pengurangan, perkalian, pembagian, dan modulus antara variabel-variabel tersebut, serta mencetak hasilnya. Pada bagian "Mixing types", program mencoba operasi penjumlahan antara variabel integer dan double, serta operasi perkalian antara variabel integer dan double.

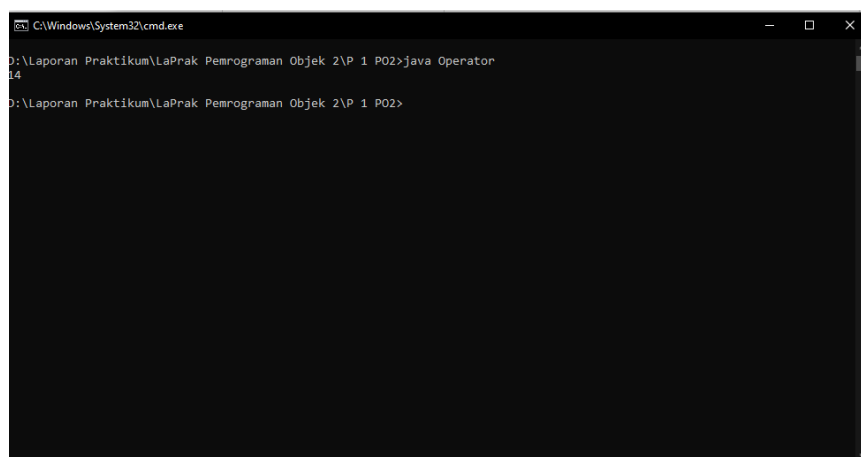


### I.3 Program I-3 Operator.java

#### I.3.A. Source Code

```
public class Operator {  
    public static void main(String []args) {  
        int i = 10;  
        int j = 3;  
        int k = 0;  
        k = ++j + i;  
        k = j++ + i;  
        k = --j + i;  
        k = j-- + i;  
        System.out.println(k);  
    }  
}
```

#### I.3.B. Hasil



Gambar 4 Output dari Program Operator

#### I.3.C. Analisa

Program di atas merupakan contoh penggunaan operator penambahan (++), pengurangan (--), dan penjumlahan (+) dalam bahasa pemrograman Java. Pada awalnya, variabel-variabel `i`, `j`, dan `k` dideklarasikan dan diinisialisasi dengan nilai-nilai tertentu. Kemudian, program melakukan operasi penambahan satu terhadap variabel `j` menggunakan operator ++. Hasil operasi tersebut kemudian ditambahkan dengan nilai variabel `i` dan disimpan dalam variabel `k`. Selanjutnya, program melakukan operasi penjumlahan antara nilai variabel `j` (yang telah ditingkatkan sebelumnya) dan nilai variabel `i`, dan hasilnya disimpan dalam variabel `k`.

## **I.4 Program I-5 TestAND.java**

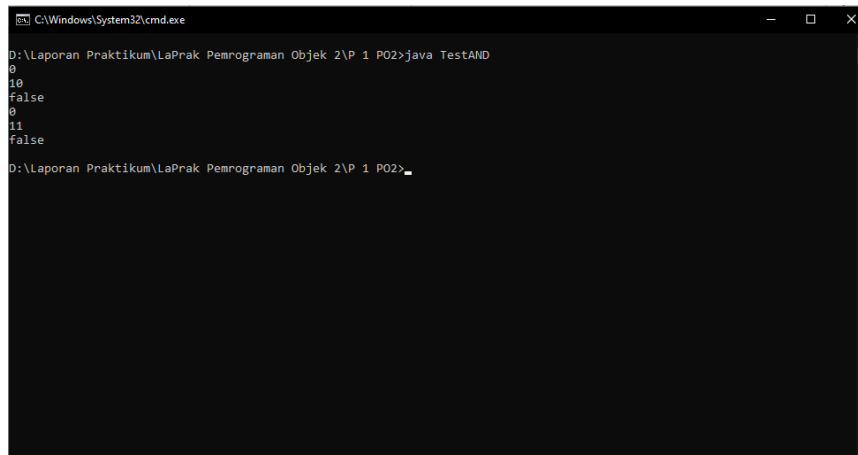
### **I.4.A. Source Code**

```
public class TestAND {  
    public static void main(String []args) {  
        int i = 0;  
        int j = 10;  
        boolean test = false;  
  
        test = (i > 10) && (j++ > 9);  
        System.out.println(i);  
        System.out.println(j);  
        System.out.println(test);  
  
        test = (i > 10) & (j++ > 9);  
        System.out.println(i);  
        System.out.println(j);  
        System.out.println(test);  
    }  
}
```

```
}

}
```

#### I.4.B. Hasil



```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java TestAND
0
10
false
0
11
false
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 5 Output dari Program TestAND*

#### I.4.C. Analisa

Program di atas adalah contoh penggunaan operator logika AND dalam bahasa pemrograman Java. Pada awalnya, variabel `i` diinisialisasi dengan nilai 0, variabel `j` diinisialisasi dengan nilai 10, dan variabel boolean `test` diinisialisasi dengan nilai false. Program kemudian melakukan dua operasi yang sama, tetapi menggunakan operator logika yang berbeda. Pertama, program menggunakan operator `&&` untuk melakukan operasi logika AND antara pernyataan  $(i > 10)$  dan  $(j++ > 9)$ .

### I.5 Program I-6 TestOR.java

#### I.5.A. Source Code

```
public class TestOR {

    public static void main(String []args) {

        int i = 0;

        int j = 10;
```

```

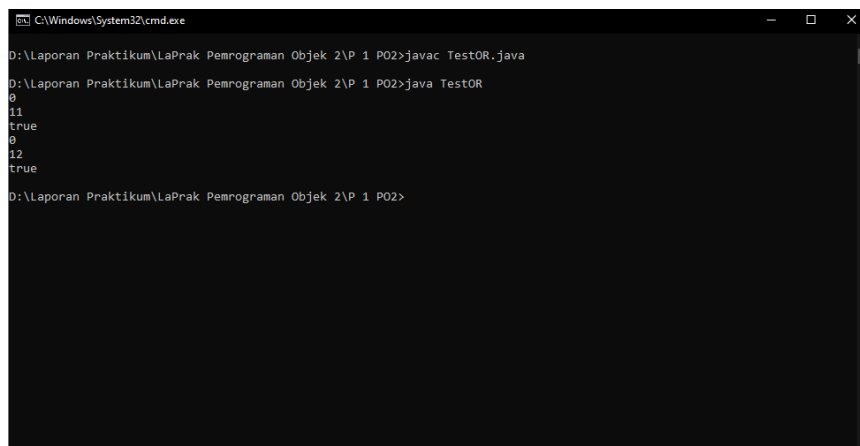
        boolean test = false;

        test = (i > 10) || (j++ > 9);
        System.out.println(i);
        System.out.println(j);
        System.out.println(test);

        test = (i > 10) | (j++ > 9);
        System.out.println(i);
        System.out.println(j);
        System.out.println(test);
    }
}

```

### I.5.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac TestOR.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java TestOR
0
11
true
0
12
true
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 6 Output dari Program TestOR*

### I.5.C. Analisa

Program "TestOR" adalah contoh penggunaan operator logika OR dalam bahasa pemrograman Java. Pada awalnya, variabel `i` diinisialisasi dengan nilai 0, variabel `j` diinisialisasi dengan nilai 10, dan variabel boolean `test` diinisialisasi dengan nilai false.

Program kemudian melakukan dua operasi yang sama, tetapi menggunakan operator logika yang berbeda. Pertama, program menggunakan operator `||` untuk melakukan operasi logika OR antara pernyataan  $(i > 10)$  dan  $(j++ > 9)$ .

## I.6 Program I-7 Person.java

### I.6.A. Source Code

```
import java.lang.*;

public class Person {

    public String name;

    public char gender;

    public int age;

    public String dateOfBirth;

    public float height;

    public float weight;

    public String address;


    public void cetakBiodata(String name, char gender, String
addres) {

        System.out.println("nama " +name+ " ,\n jenis kelamin
" +gender+ " ,\n Alamat " +address);

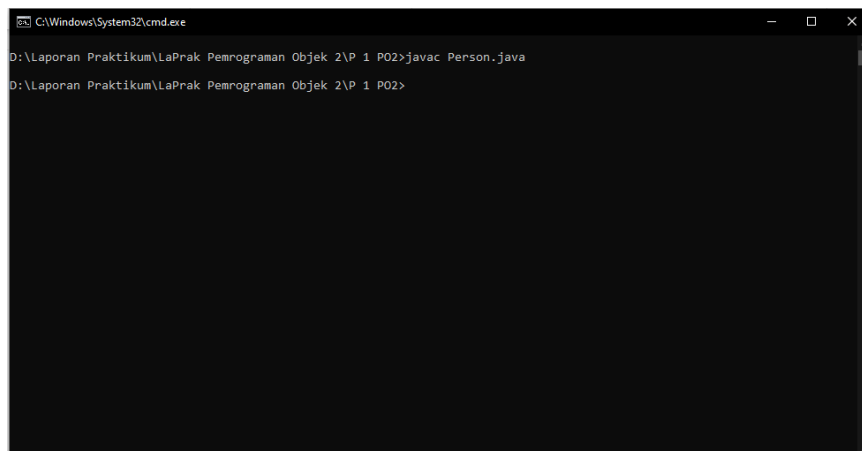
    }


    public void cetakFisik(int age, String dateOfBirth, float
height, float weight) {

        System.out.println("Umur : " + age);
```

```
        System.out.println("Tanggal Lahir : " + dateOfBirth);  
        System.out.println("Tinggi : " + height);  
        System.out.println("Berat : " + weight);  
    }  
}
```

### I.6.B. Hasil



*Gambar 7 Output dari Program Person*

### I.6.C. Analisa

Program di atas adalah sebuah kelas dalam bahasa pemrograman Java yang digunakan untuk merepresentasikan data individu, seperti nama, jenis kelamin, usia, tanggal lahir, tinggi, berat badan, dan alamat. Kelas ini memiliki beberapa variabel anggota untuk menyimpan informasi-informasi tersebut dan dua metode cetakBiodata() dan cetakFisik() untuk mencetak informasi biodata dan informasi fisik individu. Metode cetakBiodata() mencetak nama, jenis kelamin, dan alamat, sedangkan metode cetakFisik() mencetak umur, tanggal lahir, tinggi, dan berat badan.

## I.7 Program I-8 AlbumLagu.java

### I.7.A. Source Code

```
class AlbumLagu{
    String judul;
    String artis;
    int tahunRilis;
    //konstruktor
    public Albumlagu(String judul, String artis, int
tahunRilis){
        this.judul = judul;
        this.artis = artis;
        this.tahunRilis = tahunRilis;
    }

    public String infoJudul() {
        return(judul);
    }

    public String infoArtis() {
        return(artis);
    }

    public int infoTahunRilis() {
        return(tahunRilis);
    }

    public static void main(String []args) {
```

```

        AlbumLagu album = new AlbumLagu("I Love Java Code",
        "Rzk", 2016);

        System.out.println("Judul      Album      :      "      +
album.infoJudul());

        System.out.println("Artis : " + album.infoArtis());

        System.out.println("Tahun      :      "      +
album.infoTahunRilis());

    }

}

```

### I.7.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac AlbumLagu.java
AlbumLagu.java:6: error: invalid method declaration; return type required
    public AlbumLagu(String judul, String artis, int tahunRilis){
            ^
1 error
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 8 Output dari Program AlbumLagu*

### I.7.C. Analisa

Program "AlbumLagu" adalah sebuah kelas dalam bahasa pemrograman Java yang digunakan untuk merepresentasikan informasi tentang album musik, termasuk judul album, artis, dan tahun rilisnya. Kelas ini memiliki beberapa atribut, yaitu "judul", "artis", dan "tahunRilis", yang merepresentasikan judul album, nama artis, dan tahun rilis album. Selain itu, kelas ini memiliki sebuah konstruktor yang digunakan untuk menginisialisasi objek AlbumLagu dengan nilai-nilai atribut yang diberikan saat pembuatan objek.



## I.8 Program I-9 Person.java

### I.8.A. Source Code

```
import java.lang.*;

public class Person {

    public String name;

    public char gender;

    public int age;

    public String dateOfBirth;

    public float height;

    public float weight;

    public String address;

    public void cetakBiodata(String name, char gender, String
address) {

        System.out.println("nama " +name+ " ,\n jenis kelamin
" +gender+ " ,\n Alamat " +address);

    }

    public void cetakFisik(int age, String dateOfBirth, float
height, float weight) {

        System.out.println("Umur : " + age);

        System.out.println("Tanggal Lahir : " + dateOfBirth);

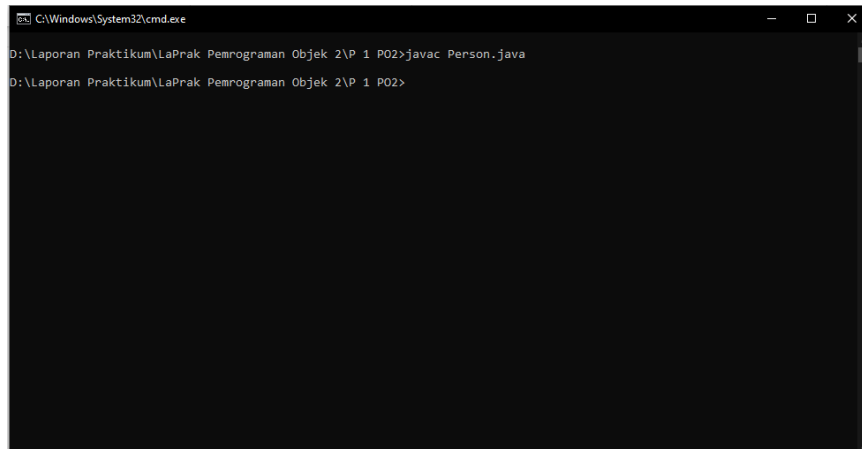
        System.out.println("Tinggi : " + height);

        System.out.println("Berat : " + weight);

    }

}
```

### I.8.B. Hasil



*Gambar 9 Output dari Program Person*

### I.8.C. Analisa

Program di atas adalah sebuah kelas dalam bahasa pemrograman Java yang digunakan untuk merepresentasikan data individu, seperti nama, jenis kelamin, usia, tanggal lahir, tinggi, berat badan, dan alamat. Kelas ini memiliki beberapa variabel anggota untuk menyimpan informasi-informasi tersebut dan dua metode cetakBiodata() dan cetakFisik() untuk mencetak informasi biodata dan informasi fisik individu. Metode cetakBiodata() mencetak nama, jenis kelamin, dan alamat, sedangkan metode cetakFisik() mencetak umur, tanggal lahir, tinggi, dan berat badan.

## I.9 Program I-10a Kucing.java

### I.9.A. Source Code

```
import java.awt.Color;

public class Kucing {
    public String nama;
    public Color warnaBulu;
```

```

public int usia;

public double bb;

public boolean statusJinak;

public String majikan;

public void cetakInformasi() {
    System.out.println("Nama: " + nama);
    System.out.println("Warna Bulu: " + warnaBulu);
    System.out.println("Usia: " + usia + " tahun");
    System.out.println("Berat Badan: " + bb + " kg");
    System.out.println("Status Jinak: " + (statusJinak ?
"Jinak" : "Liar"));
    System.out.println("Majikan: " + majikan);
}

public void diadopsi(String m) {
    majikan = m;
    statusJinak = true;
}

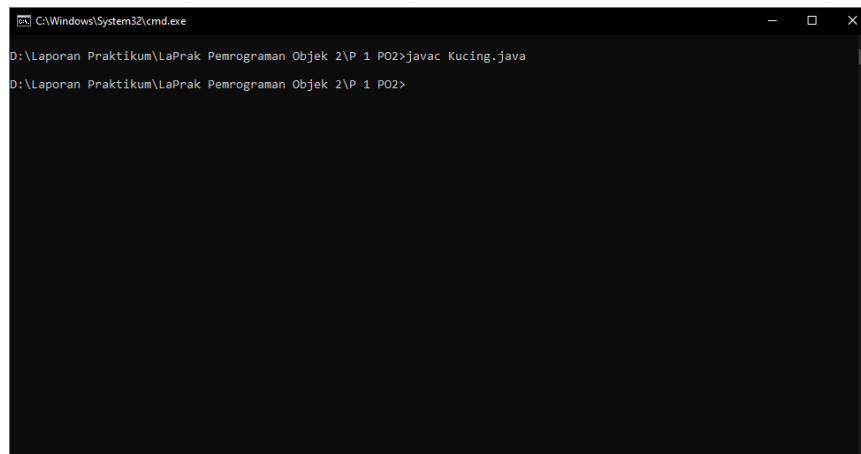
public boolean apakahJinak() {
    return statusJinak;
}

public void dilepas() {
    majikan = null;
}

```

```
        statusJinak = false;
    }
}
```

### I.9.B. Hasil



*Gambar 10 Output dari Program Kucing*

### I.9.C. Analisa

Program "Kucing" adalah sebuah kelas dalam bahasa pemrograman Java yang merepresentasikan kucing. Kelas ini memiliki beberapa atribut seperti nama, warna bulu, usia, berat badan, status jinak, dan nama majikan. Selain itu, kelas ini juga memiliki beberapa metode yang digunakan untuk mengelola informasi tentang kucing tersebut. Metode cetakInformasi() digunakan untuk mencetak informasi lengkap tentang kucing, termasuk atribut-atributnya. Metode diadopsi(String m) digunakan untuk mengadopsi kucing dengan memberikan nama majikan baru dan mengatur status jinaknya menjadi true. Metode apakahJinak() digunakan untuk memeriksa apakah kucing tersebut jinak atau tidak. Terakhir, metode dilepas() digunakan untuk melepas kucing dari pemiliknya dengan menghapus nama majikan dan mengatur status jinaknya menjadi false.

## I.10 Program I-10b LingkunganRumah.java

### I.10.A. Source Code

```

import java.awt.Color;

public class LingkunganRumah {

    public static void main(String[] args) {

        Kucing michael = new Kucing();

        Kucing garfield = new Kucing();

        michael.warnaBulu = new Color(0, 1, 1);

        michael.nama = "Michael";

        michael.usia = 3;

        michael.bb = 4.5;

        michael.diadopsi("Rezki");

        // Memanggil metode cetakInformasi() untuk menampilkan
informasi kucing "michael"

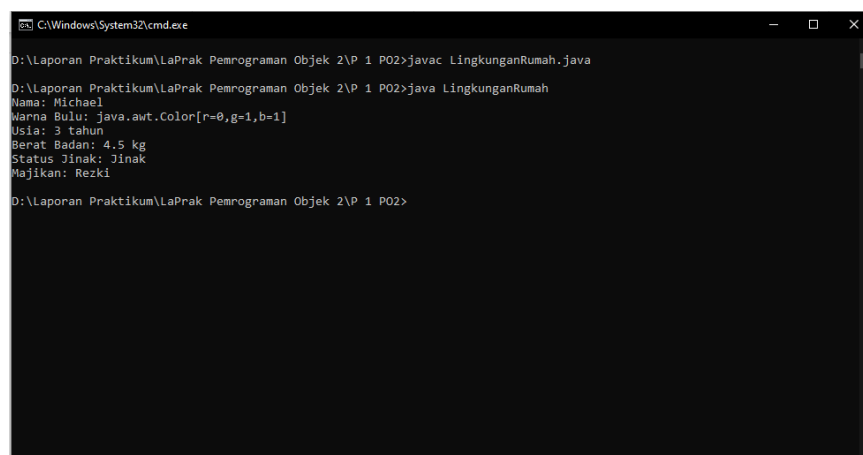
        michael.cetakInformasi();

    }

}

```

### I.10.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac LingkunganRumah.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java LingkunganRumah
Nama: Michael
Warna Bulu: java.awt.Color[r=0,g=1,b=1]
Usia: 3 tahun
Berat Badan: 4.5 kg
Status Jinak: Jinak
Majikan: Rezki
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 11 Output dari Program LingkunganRumah*

### **I.10.C. Analisa**

Program di atas adalah sebuah program dalam bahasa pemrograman Java yang menciptakan dua objek dari kelas "Kucing" yaitu "michael" dan "garfield". Pada objek "michael", beberapa atribut seperti warna bulu, nama, usia, dan berat badan diinisialisasi dengan nilai tertentu, dan kemudian metode "diadopsi()" dipanggil untuk menetapkan nama majikan. Untuk menampilkan informasi lengkap tentang kucing "michael", perlu ditambahkan pemanggilan metode "cetakInformasi()".

## **I.11 Program I-11 Lion.java**

### **I.11.A. Source Code**

```
import java.awt.Color;

class Lion {

    public String name;

    public Color furColor;

    public int age;

    public double weight;

    public Lion(String name, Color furColor, int age, double
weight) {

        this.name = name;

        this.furColor = furColor;

        this.age = age;

        this.weight = weight;

    }
}
```

```

        public void roar() {

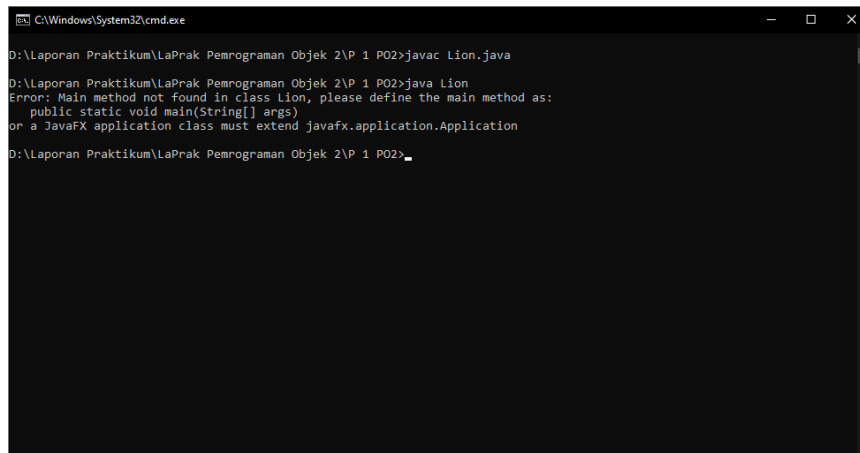
            System.out.println(name + " roars loudly!");

        }

    }
}

```

### I.11.B. Hasil



*Gambar 12 Output dari Program Lion*

### I.11.C. Analisa

Program di atas adalah sebuah kelas dalam bahasa pemrograman Java yang digunakan untuk merepresentasikan singa. Kelas ini memiliki beberapa atribut seperti nama, warna bulu, usia, dan berat badan, yang diinisialisasi melalui konstruktor saat pembuatan objek. Selain itu, kelas ini juga memiliki sebuah metode bernama "roar()" yang digunakan untuk mencetak pesan yang menunjukkan singa sedang mengaum dengan keras.

## I.12 Program I-12 Horse.java

### I.12.A. Source Code

```

import java.awt.Color;

class Horse {

```

```
public String name;

public Color furColor;

public int age;

public double weight;

public Horse(String name, Color furColor, int age, double
weight) {

    this.name = name;

    this.furColor = furColor;

    this.age = age;

    this.weight = weight;

}

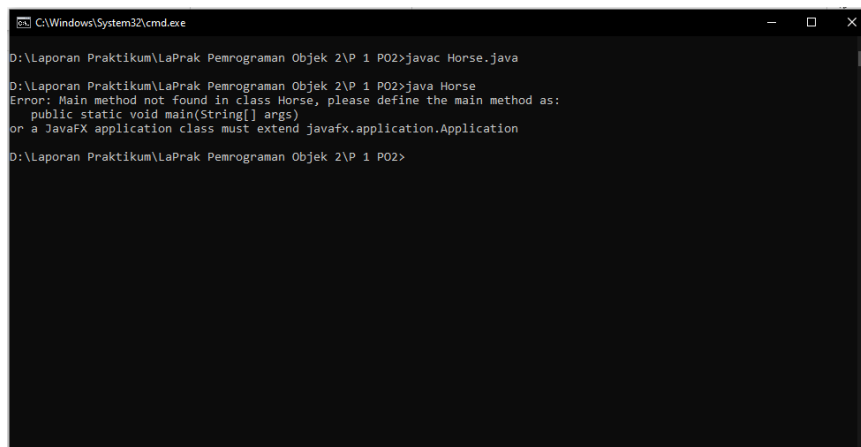
public void gallop() {

    System.out.println(name + " gallops across the field!");

}

}
```

### I.12.B. Hasil



```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Horse.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Horse
Error: Main method not found in class Horse, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 13 Output dari Program Horse.java*



### **I.12.C. Analisa**

Program di atas merupakan sebuah kelas dalam bahasa pemrograman Java yang bertujuan untuk merepresentasikan kuda. Kelas ini memiliki beberapa atribut seperti nama, warna bulu, usia, dan berat badan, yang diinisialisasi melalui konstruktor saat pembuatan objek. Selain itu, kelas ini juga memiliki sebuah metode bernama "gallop()" yang digunakan untuk mencetak pesan yang menunjukkan kuda sedang berlari kencang melintasi padang rumput.

## **I.13 Program I-13 Kangaroo.java**

### **I.13.A. Source Code**

```
import java.awt.Color;

class Kangaroo {

    public String name;

    public Color furColor;

    public int age;

    public double weight;

    public Kangaroo(String name, Color furColor, int age, double
weight) {

        this.name = name;

        this.furColor = furColor;

        this.age = age;

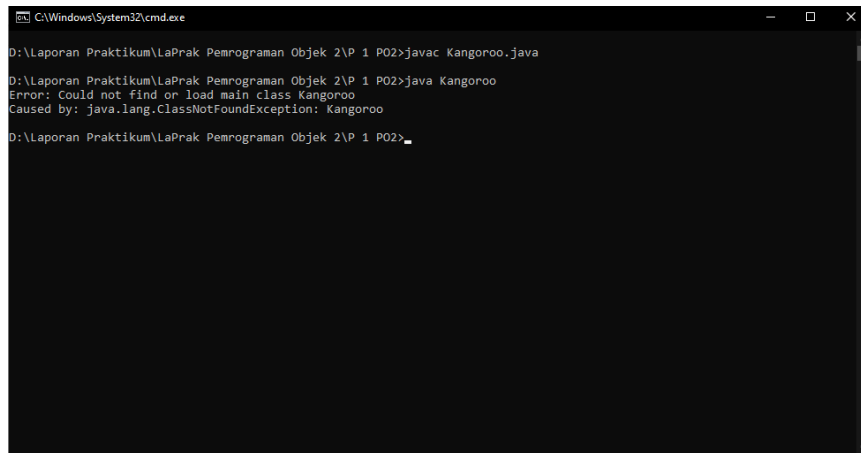
        this.weight = weight;

    }

}
```

```
public void jump() {  
    System.out.println(name + " jumps high!");  
}  
}
```

### I.13.B. Hasil



*Gambar 14 Output dari Program Kangoroo*

### I.13.C. Analisa

Program di atas adalah sebuah kelas dalam bahasa pemrograman Java yang bertujuan untuk merepresentasikan kanguru. Kelas ini memiliki beberapa atribut seperti nama, warna bulu, usia, dan berat badan, yang diinisialisasi melalui konstruktor saat pembuatan objek. Selain itu, kelas ini juga memiliki sebuah metode bernama "jump()" yang digunakan untuk mencetak pesan yang menunjukkan kanguru sedang melompat tinggi.

## I.14 Program I-14 Zoo.java

### I.14.A. Source Code

```
import java.awt.Color;
```

```

public class Zoo {

    public static void main(String[] args) {

        Lion lion = new Lion("Simba", Color.ORANGE, 5, 180.0);

        Horse horse = new Horse("Spirit", Color.BLACK, 4,
600.0);

        Kangaroo kangaroo = new Kangaroo("Joey", Color.GRAY, 2,
50.0);

        // Display information and perform actions

        lion.roar();

        System.out.println();

        horse.gallop();

        System.out.println();

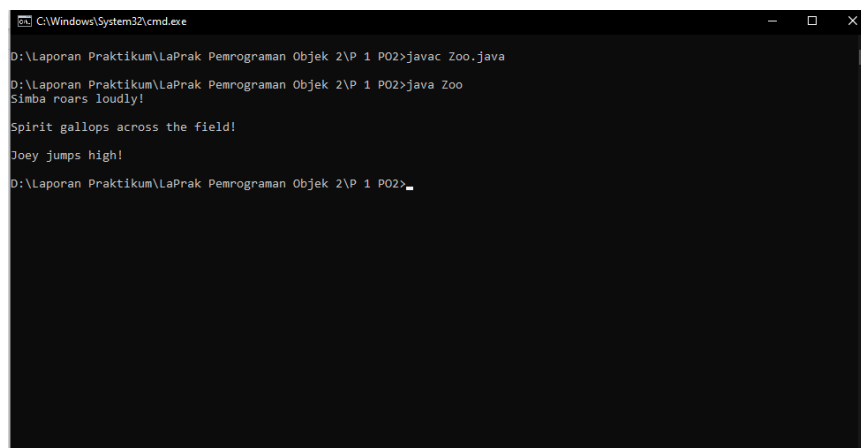
        kangaroo.jump();

    }

}

```

#### I.14.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Zoo.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Zoo
Simba roars loudly!

Spirit gallops across the field!

Joey jumps high!

D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 15 Output dari Zoo.java*

### **I.14.C. Analisa**

Program di atas adalah sebuah program dalam bahasa pemrograman Java yang bertujuan untuk mensimulasikan sebuah kebun binatang dengan menerapkan kelas-kelas "Lion", "Horse", dan "Kangaroo". Pada program ini, tiga objek kelas yang berbeda diciptakan, yaitu "lion" (singa), "horse" (kuda), dan "kangaroo" (kanguru), dengan memberikan nilai-nilai spesifik untuk atribut-atribut seperti nama, warna bulu, usia, dan berat badan melalui konstruktor masing-masing kelas. Setelah pembuatan objek, program ini menampilkan informasi dan melakukan aksi yang sesuai dengan jenis hewan yang direpresentasikan. Sebagai contoh, program memanggil metode "roar()" untuk objek "lion" yang menunjukkan singa mengaum, "gallop()" untuk objek "horse" yang menunjukkan kuda berlari kencang, dan "jump()" untuk objek "kangaroo" yang menunjukkan kanguru melompat tinggi.

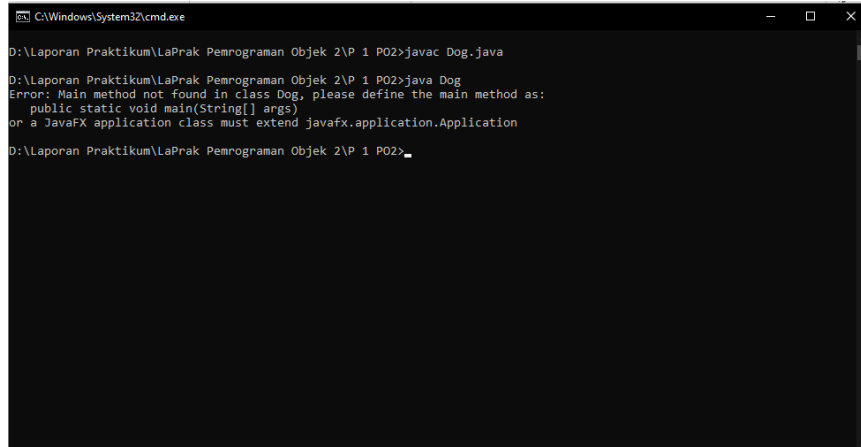
## **I.15 Program I-15 Dog.java**

### **I.15.A. Source Code**

```
public class Dog {  
    private int weight;  
    public Dog() {  
        weight = 42;  
    }  
    public int getWeight() {  
        return weight;  
    }  
  
    public void setWeight(int newWeight) {  
        weight = newWeight;  
    }  
}
```

}

### I.15.B. Hasil



```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Dog.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Dog
Error: Main method not found in class Dog, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>_
```

*Gambar 16 Output dari Program Dog*

### I.15.C. Analisa

Program di atas adalah sebuah kelas dalam bahasa pemrograman Java yang merepresentasikan sebuah anjing. Kelas ini memiliki sebuah instance variable `weight` yang bersifat private, yang artinya hanya dapat diakses langsung oleh metode-metode dalam kelas tersebut. Dalam konstruktor kelas, nilai awal `weight` diatur ke 42. Selain itu, kelas ini memiliki dua metode: `getWeight()` yang mengembalikan nilai dari `weight`, dan `setWeight(int newWeight)` yang digunakan untuk mengatur nilai `weight` dengan nilai baru yang diberikan.

## I.16 Program I-16 Elevator.java

### I.16.A. Source Code

```
public class Elevator {

    public boolean doorOpen = false;

    public int currentFloor = 1;

    public final int TOP_FLOOR = 5;
```

```
public final int BOTTOM_FLOOR = 1;

public void openDoor() {
    System.out.println("Opening door.");
    doorOpen = true;
    System.out.println("Door is open.");
}

public void closeDoor() {
    System.out.println("Closing door.");
    doorOpen = false;
    System.out.println("Door is closed.");
}

public void goUp() {
    System.out.println("Going up one floor.");
    currentFloor++;
    System.out.println("Floor: " + currentFloor);
}

public void goDown() {
    System.out.println("Going down one Floor.");
    currentFloor--;
    System.out.println("Floor: " + currentFloor);
}

public void setFloor(int desiredFloor) {
```

```

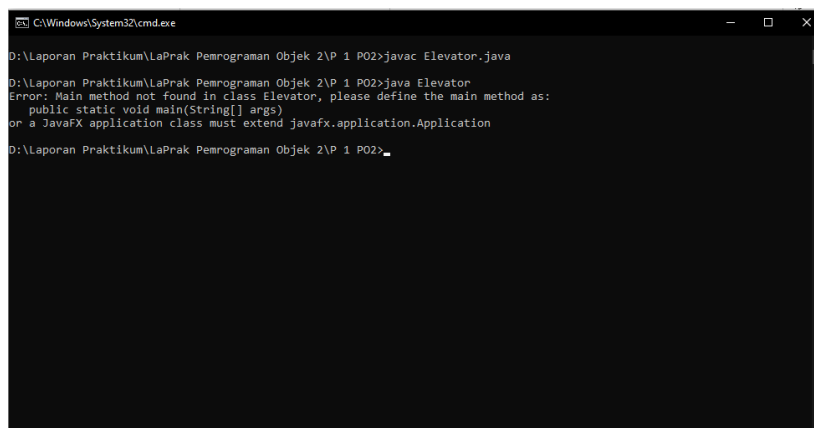
        while (currentFloor != desiredFloor) {
            if(currentFloor < desiredFloor) {
                goUp();
            } else {
                goDown();
            }
        }
    }

    public int getFloor() {
        return currentFloor;
    }

    public boolean checkDoorStatus() {
        return doorOpen;
    }
}

```

### I.16.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Elevator.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Elevator
Error: Main method not found in class Elevator, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>_

```

*Gambar 17 Output dari Program Elevator*

### I.16.C. Analisa

Program di atas adalah sebuah kelas dalam bahasa pemrograman Java yang merepresentasikan sebuah lift. Kelas ini memiliki beberapa atribut seperti `doorOpen` yang menunjukkan apakah pintu lift terbuka atau tidak, serta `currentFloor`, `TOP_FLOOR`, dan `BOTTOM_FLOOR` yang merupakan informasi tentang lantai saat ini, lantai paling atas, dan lantai paling bawah. Selain itu, kelas ini memiliki beberapa instance method yang digunakan untuk melakukan operasi-operasi terkait dengan lift, seperti `openDoor()` dan `closeDoor()` untuk membuka dan menutup pintu, `goUp()` dan `goDown()` untuk naik dan turun satu lantai, serta `setFloor(int desiredFloor)` untuk mengatur lantai yang diinginkan. Metode `getFloor()` digunakan untuk mendapatkan informasi tentang lantai saat ini, sedangkan `checkDoorStatus()` digunakan untuk memeriksa status pintu.

## I.17 Program I-17 ElevatorTest.java

### I.17.A. Source Code

```
public class ElevatorTest {  
    public static void main(String []args) {  
        Elevator myElevator = new Elevator();  
        myElevator.openDoor();  
        myElevator.closeDoor();  
        myElevator.goUp();  
        myElevator.goUp();  
        myElevator.goUp();  
        myElevator.openDoor();  
        myElevator.closeDoor();  
        myElevator.goDown();  
        myElevator.openDoor();  
    }  
}
```



```

        myElevator.closeDoor();

        myElevator.goDown();

        myElevator.setFloor (myElevator.TOP_FLOOR);

        myElevator.openDoor();

    }

}

```

### I.17.B. Hasil

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java ElevatorTest
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one Floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one Floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Going up one floor.
Floor: 5
Opening door.

```

*Gambar 18 Output dari Program ElevatorTest*

### I.17.C. Analisa

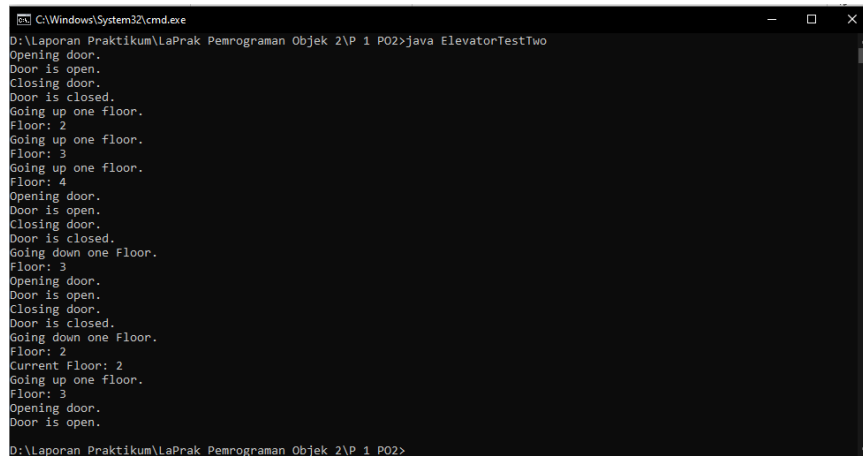
Program di atas adalah sebuah program dalam bahasa pemrograman Java yang digunakan untuk menguji fungsionalitas kelas "Elevator". Pada program ini, sebuah objek `myElevator` dari kelas "Elevator" dibuat. Kemudian, serangkaian operasi dilakukan pada lift yang direpresentasikan oleh objek tersebut, seperti membuka pintu, menutup pintu, naik satu lantai, turun satu lantai, dan mengatur lantai ke lantai paling atas. Setiap operasi tersebut dipanggil menggunakan metode yang sesuai dari objek `myElevator`. Dengan melakukan serangkaian operasi ini, program menguji apakah kelas "Elevator" bekerja dengan benar sesuai dengan implementasinya. Hasil dari setiap operasi juga akan dicetak untuk memantau proses yang sedang berlangsung.

## **I.18 Program I-18 ElevatorTestTwo.java**

### **I.18.A. Source Code**

```
public class ElevatorTestTwo {  
    public static void main(String []args) {  
        Elevator myElevator = new Elevator();  
        myElevator.openDoor();  
        myElevator.closeDoor();  
        myElevator.goUp();  
        myElevator.goUp();  
        myElevator.goUp();  
        myElevator.openDoor();  
        myElevator.closeDoor();  
        myElevator.goDown();  
        myElevator.openDoor();  
        myElevator.closeDoor();  
        myElevator.goDown();  
        int curFloor = myElevator.getFloor();  
        System.out.println("Current Floor: " + curFloor);  
        myElevator.setFloor(curFloor + 1);  
        myElevator.openDoor();  
    }  
}
```

### **I.18.B. Hasil**



```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java ElevatorTestTwo
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one Floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one Floor.
Floor: 2
Current Floor: 2
Going up one floor.
Floor: 3
Opening door.
Door is open.
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 19 Output dari Program ElevatorTestTwo*

### **I.18.C. Analisa**

Program di atas adalah sebuah program dalam bahasa pemrograman Java yang juga digunakan untuk menguji fungsionalitas kelas "Elevator". Pada program ini, sebuah objek `myElevator` dari kelas "Elevator" dibuat, dan serangkaian operasi dilakukan pada lift yang direpresentasikan oleh objek tersebut, seperti membuka pintu, menutup pintu, naik satu lantai, turun satu lantai, serta mengambil informasi tentang lantai saat ini dan mengatur lantai ke lantai berikutnya. Hasil dari setiap operasi dicetak untuk memantau proses yang sedang berlangsung. Dengan melakukan serangkaian operasi ini, program menguji apakah kelas "Elevator" bekerja dengan benar sesuai dengan implementasinya.

## **I.19 Program I-19 StudentRecord.java**

### **I.19.A. Source Code**

```
public class StudentRecord {

    private String name;

    private String address;

    private int age;

    private double mathGrade;

    private double englishGrade;
```

```

private double scienceGrade;

private double average;

private static int STUDENTCOUNT;


// Default constructor
public StudentRecord() {
    STUDENTCOUNT++;
}


// Constructor with name and address parameters
public StudentRecord2(String name, String address) {
    this.name = name;
    this.address = address;
    STUDENTCOUNT++;
}


// Constructor with math, english, and science grades
parameters
public StudentRecord2(double mathGrade, double
englishGrade, double scienceGrade) {
    this.mathGrade = mathGrade;
    this.englishGrade = englishGrade;
    this.scienceGrade = scienceGrade;
    STUDENTCOUNT++;
}

```

```
// Constructor with name and gender parameters
public StudentRecord2(String name, char gender) {
    this.name = name;
    // Additional logic for handling gender if needed
    STUDENTCOUNT++;
}

// Getter and setter methods for all attributes

public static int getStudentRecord() {
    return STUDENTCOUNT;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getAddress() {
    return address;
}

public void setAddress(String address) {
```

```
        this.address = address;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public double getMathGrade() {
        return mathGrade;
    }

    public void setMathGrade(double mathGrade) {
        this.mathGrade = mathGrade;
    }

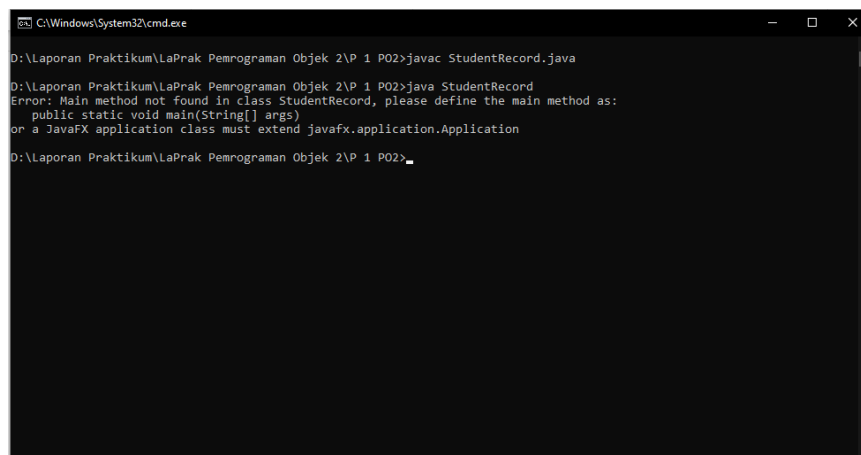
    public double getEnglishGrade() {
        return englishGrade;
    }

    public void setEnglishGrade(double englishGrade) {
        this.englishGrade = englishGrade;
    }
}
```

```
public double getScienceGrade() {  
    return scienceGrade;  
}  
  
public void setScienceGrade(double scienceGrade) {  
    this.scienceGrade = scienceGrade;  
}  
  
public double getAverage() {  
    double result = 0;  
    result = (mathGrade + englishGrade + scienceGrade)/3;  
    return result;  
}  
  
public void print(String studentName) {  
    System.out.println("Name = " + studentName);  
    System.out.println("Address = " + address);  
    System.out.println("Age = " + age);  
}  
  
// Updated print method to accept three double parameters  
public void print(double englishGrade, double mathGrade,  
double scienceGrade) {  
    System.out.println("Name = " + studentName);
```

```
        System.out.println("English Grade = " + englishGrade);  
        System.out.println("Math Grade = " + mathGrade);  
        System.out.println("Science Grade = " + scienceGrade);  
    }  
}
```

### I.19.B. Hasil



```
C:\Windows\System32\cmd.exe  
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac StudentRecord.java  
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java StudentRecord  
Error: Main method not found in class StudentRecord, please define the main method as:  
    public static void main(String[] args)  
or a JavaFX application class must extend javafx.application.Application  
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 20 Output dari Program StudentRecord*

### I.19.C. Analisa

Program di atas merupakan sebuah kelas yang merepresentasikan data mengenai catatan siswa. Kelas ini memiliki beberapa atribut seperti nama, alamat, umur, nilai matematika, nilai bahasa Inggris, dan nilai sains. Selain itu, terdapat pula atribut rata-rata nilai. Kelas ini memiliki beberapa konstruktor yang memungkinkan pembuatan objek dengan berbagai macam parameter, seperti nama dan alamat, nilai matematika, bahasa Inggris, dan sains, serta nama dan jenis kelamin. Kelas ini juga memiliki metode-metode getter dan setter untuk mengakses dan mengubah nilai-nilai atributnya. Terdapat pula metode `getAverage()` yang menghitung rata-rata dari nilai matematika, bahasa Inggris, dan sains.



## I.20 Program I-20 StudentRecordExample.java

### I.20.A. Source Code

```
public class StudentRecordExample {  
    public static void main(String[] args) {  
        StudentRecord dayu = new StudentRecord();  
        StudentRecord lani = new StudentRecord();  
        StudentRecord bambang = new StudentRecord();  
  
        StudentRecord karyono = new StudentRecord("Karyono");  
        StudentRecord songjongki = new StudentRecord("Song Jong-  
Ki", "Cibaduyut");  
        StudentRecord masbejo = new StudentRecord(80, 90, 100);  
  
        dayu.setName("Dayu Anderson");  
        lani.setName("Lani Rukoyah");  
        bambang.setName("Cristhoper Bambang");  
  
        System.out.println(dayu.getName());  
  
        System.out.println("Count          =          "          +  
StudentRecord.getStudentRecord());  
        StudentRecord2 annaRecord = new StudentRecord2();  
        annaRecord.setName("Anna");  
        annaRecord.setAddress("Philippines");  
        annaRecord.setAge(15);  
        annaRecord.setMathGrade(80);  
        annaRecord.setEnglishGrade(95.5);  
    }  
}
```

```

        annaRecord.setScienceGrade(100);

        annaRecord.print(annaRecord.getName());

        annaRecord.print(annaRecord.getEnglishGrade(),

        annaRecord.getMathGrade(),

        annaRecord.getScienceGrade());

        dayu.print(dayu.getName());

    }

}

```

### I.20.B. Hasil

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>java StudentRecordExample
Dayu Anderson
Count = 6
Name = Anna
Address = Philippines
Age = 15
Math Grade = 80.0
English Grade = 95.5
Science Grade = 100.0
English Grade = 95.5
Math Grade = 80.0
Science Grade = 100.0
Name = Dayu Anderson
Address = null
Age = 0
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>_

```

*Gambar 21 Output dari Program StudentRecordExample*

### I.20.C. Analisa

Program di atas adalah sebuah contoh penggunaan kelas `StudentRecord`. Pada program ini, beberapa objek `StudentRecord` dibuat dengan menggunakan berbagai konstruktor yang tersedia. Objek `dayu`, `lani`, dan `bambang` dibuat menggunakan konstruktor default tanpa parameter, sementara `karyono` dibuat dengan memberikan nama sebagai argumen untuk konstruktor yang menerima satu parameter bertipe `String`. Selanjutnya, objek `songjongki` dibuat dengan memberikan nama dan alamat sebagai argumen untuk konstruktor yang menerima dua parameter bertipe `String`, dan objek `masbejo` dibuat dengan memberikan nilai-nilai untuk nilai matematika, bahasa Inggris, dan sains. Setelah

membuat objek, beberapa metode setter dipanggil untuk mengatur nama, alamat, usia, dan nilai-nilai akademik untuk objek `annaRecord`. Kemudian, metode `print` dipanggil untuk mencetak informasi tentang objek `annaRecord` serta nilai-nilainya. Metode `print` juga dipanggil untuk mencetak nama objek `dayu`. Selain itu, program juga mencetak jumlah total objek `StudentRecord` yang telah dibuat menggunakan metode `getStudentRecord`.

## I.21 Program I-21 Person.java

### I.21.A. Source Code

```
public class Persons {  
    protected String name;  
    protected String address;  
  
    public Persons() {  
        System.out.println("Inside Person:Constructor");  
        name = "";  
        address = "";  
    }  
  
    public Persons(String name, String address) {  
        this.name = name;  
        this.address = address;  
    }  
  
    public String getName() {  
        return name;  
    }  
}
```

```

    }

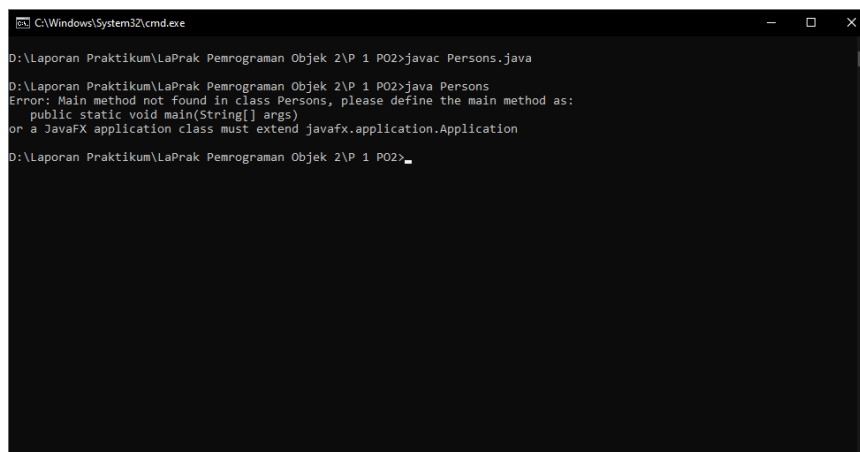
    public String getAddress() {
        return address;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setAddress(String add) {
        this.address = add;
    }
}

```

### I.21.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Persons.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Persons
Error: Main method not found in class Persons, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>_

```

*Gambar 22 Output dari Program Person*

### I.21.C. Analisa

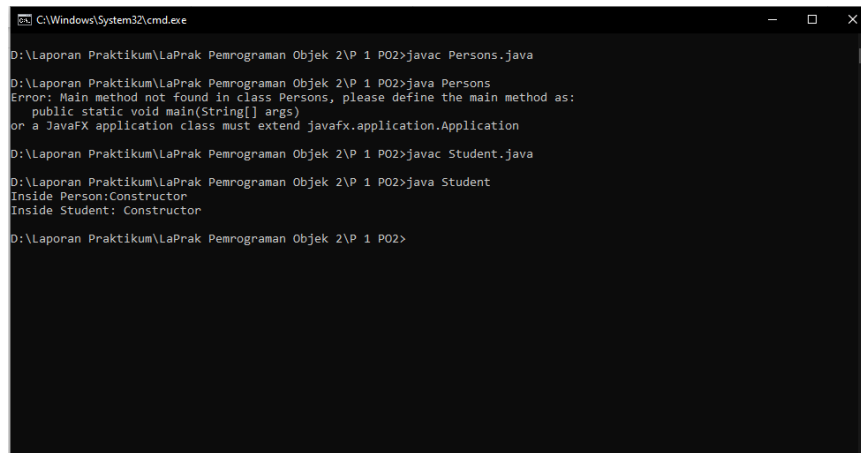
Program di atas adalah sebuah kelas yang menyediakan representasi umum untuk objek yang memiliki nama dan alamat. Kelas ini memiliki dua konstruktor, satu tanpa parameter dan satu dengan dua parameter, yang digunakan untuk menginisialisasi nilai dari variabel `name` dan `address`. Selain itu, terdapat metode-metode `getName()` dan `getAddress()` yang digunakan untuk mengakses nilai variabel `name` dan `address`, serta metode `setName()` dan `setAddress()` yang digunakan untuk mengatur nilai variabel `name` dan `address`. Metode konstruktor tanpa parameter akan menginisialisasi `name` dan `address` dengan string kosong, sementara metode konstruktor dengan parameter akan menginisialisasi `name` dan `address` sesuai dengan nilai yang diberikan. Saat objek `Persons` dibuat, pesan "Inside Person:Constructor" akan dicetak ke konsol untuk memberikan informasi bahwa konstruktor telah dipanggil.

## **I.22 Program I-22 Student.java**

### **I.22.A. Source Code**

```
public class Student extends Persons{  
    public Student(){  
        //super("SomeName", "SomeAddress");  
        //super();  
        //super.name = "name";  
        System.out.println("Inside Student: Constructor");  
    }  
    public static void main(String []args){  
        Student anna = new Student();  
    }  
}
```

### I.22.B. Hasil



```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Persons.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Persons
Error: Main method not found in class Persons, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Student.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Student
Inside Person:Constructor
Inside Student: Constructor
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 23 Output dari Program Student*

### I.22.C. Analisa

Program di atas adalah sebuah subkelas dari kelas `Persons` yang menggambarkan entitas siswa. Kelas ini memiliki sebuah konstruktor tanpa parameter yang mencetak pesan "Inside Student: Constructor" ke konsol saat objek `Student` dibuat. Konstruktor ini tidak secara langsung memanggil konstruktor dari kelas induk `Persons`, meskipun ada beberapa percobaan pemanggilan konstruktor induk yang di-comment. Tanpa pemanggilan konstruktor induk, secara default, konstruktor tanpa parameter dari kelas induk akan dipanggil oleh konstruktor kelas anak. Oleh karena itu, saat objek `Student` dibuat, konstruktor dari kelas `Persons` akan dipanggil, yang kemudian akan mencetak pesan "Inside Person:Constructor" ke konsol untuk memberikan informasi bahwa konstruktor kelas induk telah dipanggil.

## I.23 Program I-23 Pakaian.java

### I.23.A. Source Code

```
public class Pakaian {
    private int ID = 0;
    private String keterangan = "-keterangan diperlukan-";
    private double harga = 0.0;
```

```
private int jmlStok = 0;

private static int UNIQUE_ID = 0;


public Pakaian() {
    ID = UNIQUE_ID++;
}


public int getID() {
    return ID;
}


public void setKeterangan(String d) {
    keterangan = d;
}


public String getKeterangan() {
    return keterangan;
}


public double getHarga() {
    return harga;
}


public void setHarga(double p) {
    harga = p;
}
```

```

        public int getJmlStok() {
            return jmlStok;
        }

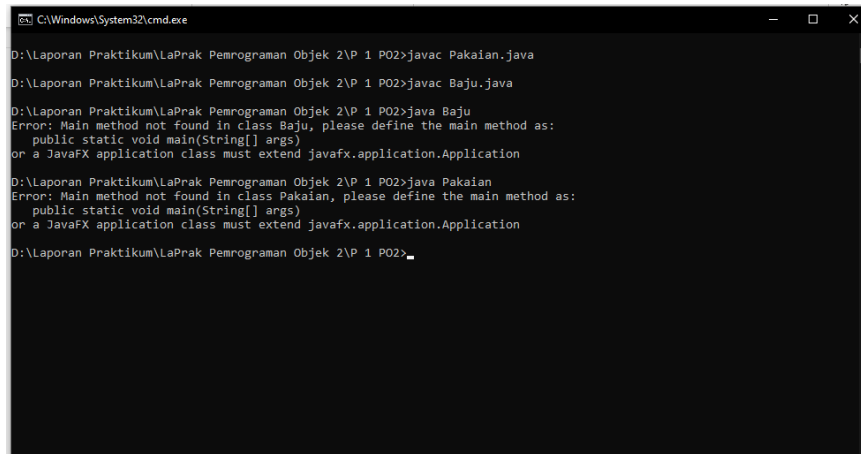
        public void setJmlStok(int q) {
            jmlStok = q;
        }
    }

    public class Baju extends Pakaian{
        //Kode Warna R = Merah, B = Biru, G = Hijau, U = Belum
        Ditentukan
        public char kodeWarna = 'U';
        //Method ini menampilkan nilai untuk suatu item
        public void tampilInformasiBaju(){
            System.out.println("ID Baju : " + getID());
            System.out.println("Keterangan      :      "      +
            getKeterangan());
            System.out.println("Kode Warna : " + kodeWarna);
            System.out.println("Harga Baju : " + getHarga());
            System.out.println("Jumlah Stok : " + getJmlStok());
        }
    }
}

```

### **I.23.B. Hasil**





```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Pakaian.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Baju.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Baju
Error: Main method not found in class Baju, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Pakaian
Error: Main method not found in class Pakaian, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 24 Output dari Program Pakaian*

### **I.23.C. Analisa**

Program di atas terdiri dari dua kelas, yaitu `Pakaian` dan `Baju`, yang menggambarkan suatu sistem manajemen pakaian. Kelas `Pakaian` memiliki atribut seperti ID, keterangan, harga, dan jumlah stok, dengan metode untuk mengatur dan mengambil nilai-nilai atribut tersebut. Kelas `Baju` merupakan subkelas dari `Pakaian` yang menambahkan atribut `kodeWarna` yang menunjukkan warna baju dengan kode tertentu dan metode `tampilInformasiBaju()` untuk menampilkan informasi lengkap tentang baju, termasuk ID, keterangan, kode warna, harga, dan jumlah stok. Konstruktor kelas `Pakaian` digunakan untuk memberikan nilai unik pada ID setiap kali objek `Pakaian` dibuat, sedangkan kode warna diinisialisasi dengan 'U' yang menunjukkan belum ditentukan. Program ini memanfaatkan konsep inheritance di mana kelas `Baju` mewarisi atribut dan metode dari kelas `Pakaian`, sehingga dapat menggunakan fungsi-fungsi tersebut secara langsung.

## **I.24 Program I-24 Polimorfisme**

### **I.24.A. Source Code**

```
public class Persons {
    protected String name;
    protected String address;
```

```
public Persons() {  
    System.out.println("Inside Person:Constructor");  
    name = "";  
    address = "";  
}  
  
public Persons(String name, String address) {  
    this.name = name;  
    this.address = address;  
}  
  
public String getName() {  
    return name;  
}  
  
public String getAddress() {  
    return address;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public void setAddress(String add) {  
    this.address = add;  
}
```

```

    }
}

public class Student extends Persons{
    public Student(){
        //super("SomeName", "SomeAddress");
        //super();
        //super.name = "name";
        System.out.println("Inside Student: Constructor");
    }

    public String getName() {
        System.out.println("Student Name : " +name);
        return name;
    }

    public static void main(String []args){
        Student anna = new Student();
    }
}

public class Employee extends Person{
    public String getName(){
        System.out.println("Employee Name : " + name);
        return name;
    }
}

```

```

        public static void main(String []args){

            Person ref;

            Student studentObject = new Student();

            Employee employeeObject = new Employee();

            ref = studentObject; //Person menunjuk kepada object
Student
class dipanggil

            String temp = ref.getName(); //getName dari student
class dipanggil

            System.out.println(temp);

            ref = employeeObject; //Person menunjuk kepada object
Employee
class dipanggil

            temp = ref.getName(); //getName dari Employee class
dipanggil

            System.out.println(temp);

        }

    }

```

### I.24.B. Hasil

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Persons.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Student.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Employee.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Persons
Error: Main method not found in class Persons, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Student
Inside Person:Constructor
Inside Student: Constructor
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Employee
Inside Person:Constructor
Inside Student: Constructor
Inside Person:Constructor
Student Name :
Employee Name :
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

Gambar 25 Output dari Program Polimorfisme

### I.24.C. Analisa

Program di atas terdiri dari tiga kelas: `Persons`, `Student`, dan `Employee`. Kelas `Persons` memiliki dua konstruktor untuk menginisialisasi atribut `name` dan `address`, serta metode getter dan setter untuk masing-masing atribut tersebut. Kelas `Student` merupakan subkelas dari `Persons`, yang memiliki konstruktor tanpa parameter dan metode `getName()` yang mencetak nama mahasiswa. Kelas `Employee` juga merupakan subkelas dari `Persons`, dengan metode `getName()` yang mencetak nama karyawan. Pada metode `main` dalam kelas `Employee`, dilakukan pembuatan objek `Student` dan `Employee`, kemudian dilakukan penggunaan polimorfisme dengan membuat referensi `ref` yang menunjuk ke objek `Student` dan `Employee` secara bergantian, yang pada gilirannya akan memanggil metode `getName()` yang sesuai dengan objek yang ditunjuk oleh `ref`.

## I.25 Program I-25 PrivateElevator2

### I.25.A. Source Code

```
public class PrivateElevator2{

    private boolean bukaPintu = false;

    private int lantaiSkrng = 1;

    private int berat = 0;

    private final int KAPASITAS = 1000;

    private final int LANTAI_ATAS = 5;

    private final int LANTAI_BAWAH = 1;

    public void buka(){

        bukaPintu = true;

    }

    public void tutup(){

        hitungKapasitas();

        if(berat <= KAPASITAS){
```

```

        bukaPintu = false;
    } else {
        System.out.println("Elevator kelebihan beban");
        System.out.println("Pintu akan tetap terbuka
sampai seseorang keluar");
    }
}

//pada dunia nyata, elevator menggunakan sensor berat untuk
memeriksa beban,

//tetapi agar lebih sederhana, digunakan bilangan acak
untuk berat

private void hitungKapasitas(){
    berat = (int)(Math.random()*1500);
    System.out.println("Berat : " + berat);
}

public void naik(){
    if(!bukaPintu){
        if(lantaiSkrg < LANTAI_ATAS){
            lantaiSkrg++;
            System.out.println(lantaiSkrg);
        } else {
            System.out.println("Sudah mencapai Lantai
atas");
        }
    } else {
        System.out.println("Pintu masih Terbuka");
    }
}

```

```

    }

    public void turun(){
        if(!bukaPintu){
            if(lantaiSkrng < LANTAI_BAWAH){
                lantaiSkrng--;
                System.out.println(lantaiSkrng);
            } else {
                System.out.println("Sudah mencapai Lantai
bawah");
            }
        } else {
            System.out.println("Pintu masih Terbuka");
        }
    }

    public void setLantai(int tujuan){
        if((tujuan >= LANTAI_BAWAH) && (tujuan <=
LANTAI_ATAS)){
            while(lantaiSkrng != tujuan){
                if(lantaiSkrng < tujuan){
                    naik();
                } else {
                    turun();
                }
            }
        } else {
            System.out.println("Lantai Salah");
        }
    }
}

```

```

        }

    }

    public int getLantai(){
        return lantaiSkrng;
    }

    public boolean getStatusPintu(){
        return bukaPintu;
    }
}

public class PrivateElevator2Test{
    public static void main(String []args){
        PrivateElevator2 privateElevator = new
PrivateElevator2();

        privateElevator.buka();
        privateElevator.tutup();
        privateElevator.turun();
        privateElevator.naik();
        privateElevator.naik();
        privateElevator.buka();
        privateElevator.tutup();
        privateElevator.turun();
        privateElevator.buka();
        privateElevator.turun();
        privateElevator.tutup();
        privateElevator.turun();
    }
}

```



```

        privateElevator.turun();

        int lantai = privateElevator.getLantai();

        if(lantai != 5 && !privateElevator.getStatusPintu()){

            privateElevator.setLantai(5);

        }

        privateElevator.setLantai(10);

        privateElevator.buka();

    }

}

```

### I.25.B. Hasil

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac PrivateElevator2.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac PrivateElevator2Test.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java PrivateElevator2Test
Berat : 157
Sudah mencapai Lantai bawah
2
3
Berat : 1281
Elevator kelebihan beban
Pintu akan tetap terbuka sampai seseorang keluar
Pintu masih Terbuka
Pintu masih Terbuka
Berat : 235
Sudah mencapai Lantai bawah
Sudah mencapai Lantai bawah
4
5
Lantai Salah
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 26 Output dari Program PrivateElevator2*

### I.25.C. Analisa

Program ini mengimplementasikan fungsionalitas elevator dalam kelas `PrivateElevator2`. Elevator memiliki atribut seperti `bukaPintu` untuk menunjukkan status pintu terbuka atau tertutup, `lantaiSkrng` untuk menunjukkan lantai saat ini, dan `berat` untuk menunjukkan beban saat ini. Metode `buka()`, `tutup()`, `naik()`, `turun()`, dan `setLantai()` mengatur perilaku elevator sesuai dengan aksi yang diambil pengguna. Aksi seperti membuka pintu, menaikkan atau menurunkan elevator, serta menetapkan lantai tujuan hanya dapat dilakukan jika pintu tertutup. Program pengujian dalam kelas `PrivateElevator2Test` memanggil berbagai metode untuk menguji fungsionalitas elevator, seperti membuka dan

menutup pintu, naik turun ke lantai tertentu, serta mengatur lantai tujuan yang valid dan tidak valid. Program juga mencetak informasi tentang berat beban pada setiap aksi elevator.

## **I.26 Program I-26 Abstract Class**

### **I.26.A. Source Code**

```
public abstract class LivingThing {  
    public void breath() {  
        System.out.println("Living Thing breathing. . .");  
    }  
  
    public void eat() {  
        System.out.println("Living Thing eating. . .");  
    }  
  
    /**  
     * Abstract method walk  
     * Kita ingin method ini di-overridden oleh subclasses  
     */  
    public abstract void walk();  
}  
  
public class Human extends LivingThing {  
    // Konstruktor dan metode lainnya untuk kelas Human
```

```

@Override

public void walk() {

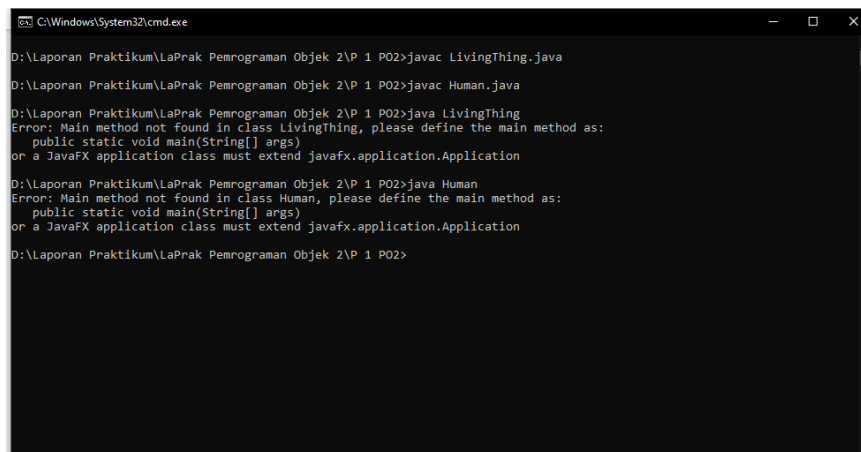
    System.out.println("Human is walking...");

}

}

```

### I.26.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac LivingThing.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Human.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java LivingThing
Error: Main method not found in class LivingThing, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Human
Error: Main method not found in class Human, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 27 Output dari Program Abstract Class*

### I.26.C. Analisa

Program ini merupakan contoh implementasi pewarisan dan abstraksi dalam pemrograman berorientasi objek. Kelas `LivingThing` adalah kelas abstrak yang memiliki dua metode konkret, yaitu `breath()` dan `eat()`, serta satu metode abstrak `walk()`. Metode `walk()` menjadi abstrak karena perilaku berjalan dapat bervariasi tergantung pada jenis makhluk hidup. Kelas `Human` merupakan turunan dari kelas `LivingThing` yang mengimplementasikan metode `walk()` sesuai dengan perilaku berjalan manusia. Dengan menggunakan abstraksi, program ini memisahkan definisi umum tentang makhluk hidup dari perilaku khusus yang dimiliki manusia.

## I.27 Program I-27 Interface

### I.27.A. Source Code

```
Q// File: Line.java

public class Line implements Relation {

    private double x1;

    private double x2;

    private double y1;

    private double y2;


    public Line(double x1, double x2, double y1, double y2) {

        this.x1 = x1;

        this.x2 = x2;

        this.y1 = y1;

        this.y2 = y2;

    }


    public double getLength() {

        double length = Math.sqrt((x2 - x1) * (x2 - x1) + (y2 -
y1) * (y2 - y1));

        return length;

    }

    public boolean isGreater(Object a, Object b) {

        double aLen = ((Line) a).getLength();

        double bLen = ((Line) b).getLength();

        return (aLen > bLen);

    }

}
```

```

    public boolean isLess(Object a, Object b) {
        double aLen = ((Line) a).getLength();
        double bLen = ((Line) b).getLength();
        return (aLen < bLen);
    }

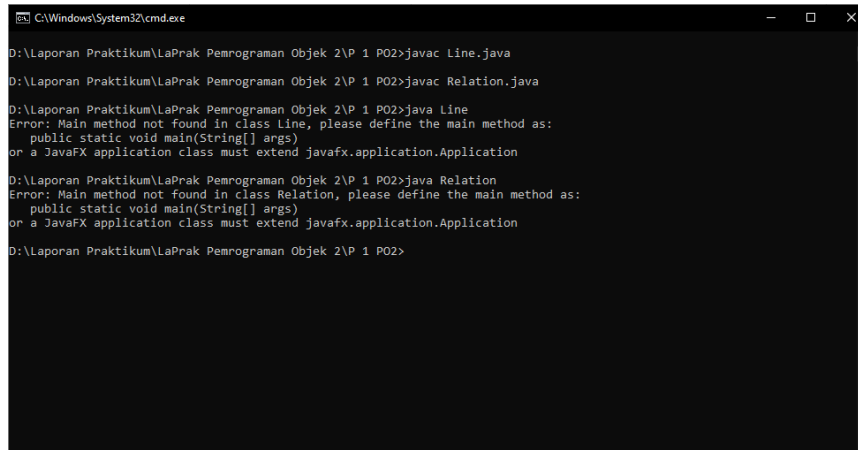
    public boolean isEqual(Object a, Object b) {
        double aLen = ((Line) a).getLength();
        double bLen = ((Line) b).getLength();
        return (aLen == bLen);
    }
}

// File: Relation.java

public interface Relation {
    public boolean isGreater(Object a, Object b);
    public boolean isLess(Object a, Object b);
    public boolean isEqual(Object a, Object b);
}

```

### **I.27.B. Hasil**



```
C:\Windows\System32\cmd.exe

D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Line.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Relation.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Line
Error: Main method not found in class Line, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Relation
Error: Main method not found in class Relation, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
```

*Gambar 28 Output dari Program Interface*

### **I.27.C. Analisa**

Program ini terdiri dari dua file, yaitu `Line.java` dan `Relation.java`. Antarmuka `Relation` mendefinisikan tiga metode untuk memeriksa hubungan antara dua objek: `isGreater`, `isLess`, dan `isEqual`. Kelas `Line` mengimplementasikan antarmuka `Relation` dan menghitung panjang garis dengan metode `getLength()`. Metode-metode lain dalam kelas `Line` menggunakan panjang garis untuk membandingkan dua objek berdasarkan hubungan yang didefinisikan dalam antarmuka `Relation`.

## **BAB II. TUGAS PRAKTIKUM**

### **II.1 Tugas I-1**

1. Program sudah di compile dan sudah di Analisa di BAB Hasil Praktikum Program I-1

### **II.2 Tugas I-2**

1. Program sudah di compile dan sudah di Analisa di BAB Hasil Praktikum Program I-2
2. Program sudah dimodifikasi dan sudah diperbaiki di BAB Hasil Praktikum Program I-2

### **II.3 Tugas I-5**

1. Program sudah di Analisa di BAB Hasil Praktikum Program I-5
2. Jika operan di sebelah kiri dari operator && bernilai false, maka operan di sebelah kanannya tidak dievaluasi, karena hasil keseluruhan dari operasi AND akan selalu false. Ini dikenal sebagai "short-circuiting". Sementara Baik operan di sebelah kiri maupun kanan dari operator & akan dievaluasi, bahkan jika operan di sebelah kiri bernilai false. Tidak ada "short-circuiting" di sini.
3. Dalam operator AND (&&), jika operand di sebelah kiri operator merupakan false, maka hasil akhir dari ekspresi secara keseluruhan pasti akan menjadi false. Dalam hal ini, evaluasi operand di sebelah kanan operator tidak perlu dilakukan karena hasil keseluruhan ekspresi sudah pasti. Oleh karena itu, dalam short circuit evaluation, evaluasi dihentikan setelah operand pertama jika itu sudah cukup untuk menentukan hasil akhir.

### **II.4 Tugas I-6**

1. Program sudah di Analisa di BAB Hasil Praktikum Program I-5
2. Perbedaan antara || dan | adalah bahwa || melakukan evaluasi kondisional (short-circuiting) berdasarkan hasil dari operand sebelumnya, sementara | melakukan operasi bitwise pada setiap bit dari operand tanpa memperhatikan hasil dari operand sebelumnya.
3. Dalam operator OR (||), jika operand di sebelah kiri operator merupakan true, maka hasil akhir dari ekspresi secara keseluruhan pasti akan menjadi true. Dalam hal ini, evaluasi operand di sebelah kanan operator tidak perlu dilakukan karena hasil keseluruhan ekspresi sudah pasti. Oleh karena itu, dalam short circuit evaluation,

evaluasi dihentikan setelah operand pertama jika itu sudah cukup untuk menentukan hasil akhir.

## II.5 Tugas I-7

1. Objek adalah instansi dari sebuah kelas. Objek mewakili sesuatu yang nyata atau konseptual dalam dunia nyata, seperti manusia, mobil, atau buku. Objek memiliki atribut yang merepresentasikan karakteristiknya, dan metode yang merepresentasikan perilakunya.
2. Kucing, Elevator, Person
3. b. “Good Bye”
  - c. 1234
  - e. \$hello\$
  - f. JAVA
  - g. hello, there
  - i. 4you
  - j. \_doWork

## II.6 Tugas I-8 KelasPB.java

### II.6.A. Source Code

```
// Kelas Point

class Point {

    // Instance variables

    private double x;

    private double y;

    private double z;


    // Static variable

    private static int count = 0;


    // Constructor

    public Point(double x, double y, double z) {
```



```

        this.x = x;

        this.y = y;

        this.z = z;

        count++; // Menambah nilai count setiap kali objek Point
dibuat
    }

    // Instance method untuk menampilkan koordinat titik
    public void displayCoordinates() {
        System.out.println("Koordinat titik: (" + x + ", " + y
+ ", " + z + ")");
    }

    // Static method untuk menampilkan jumlah objek Point yang
telah dibuat
    public static void displayPointCount() {
        System.out.println("Jumlah objek Point yang telah
dibuat: " + count);
    }
}

// Kelas Buku
class Buku {
    // Instance variables
    private String penulis;
    private String judul;
    private String nomorISBN;

```

```

// Constructor
public Buku(String penulis, String judul, String nomorISBN)
{
    this.penulis = penulis;
    this.judul = judul;
    this.nomorISBN = nomorISBN;
}

// Instance method untuk menampilkan informasi buku
public void displayInfo() {
    System.out.println("Buku: " + judul);
    System.out.println("Penulis: " + penulis);
    System.out.println("Nomor ISBN: " + nomorISBN);
}

// Static method untuk menampilkan informasi tentang
aplikasi buku
public static void displayApplicationInfo() {
    System.out.println("Aplikasi Buku");
    System.out.println("Versi 1.0");
    System.out.println("Dikembangkan oleh: Fathir Ahmad
Nurpadli");
}
}

```

```

public class KelasPB {
    public static void main(String[] args) {
        // Membuat objek-objek Point

        Point point1 = new Point(1.0, 2.0, 3.0);
        Point point2 = new Point(4.0, 5.0, 6.0);

        // Memanggil instance method untuk menampilkan koordinat
titik

        point1.displayCoordinates();
        point2.displayCoordinates();

        // Memanggil static method untuk menampilkan jumlah
objek Point yang telah dibuat

        Point.displayPointCount();

        System.out.println();

        // Membuat objek Buku

        Buku buku1 = new Buku("Fathir Ahmad Nurpadli", "Java
Programming", "999-0123456789");

        // Memanggil instance method untuk menampilkan informasi
buku

        buku1.displayInfo();

        // Memanggil static method untuk menampilkan informasi
tentang aplikasi buku

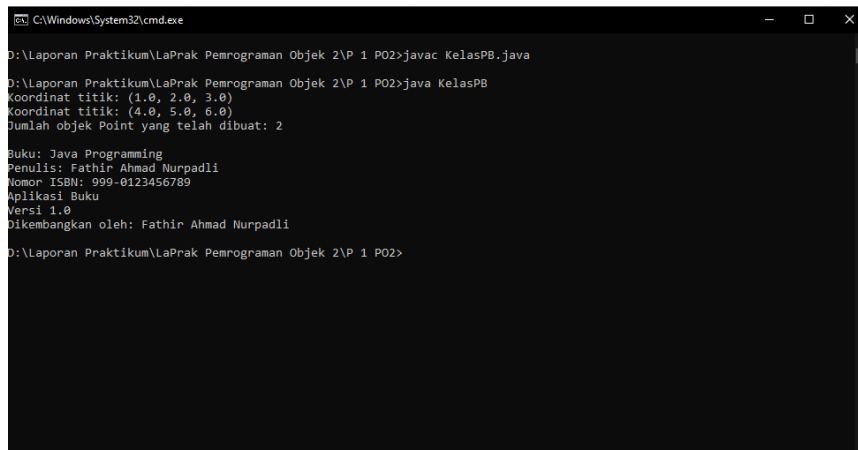
```

```

        Buku.displayApplicationInfo();
    }
}

```

## II.6.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>javac KelasPB.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>java KelasPB
Koordinat titik: (1.0, 2.0, 3.0)
Koordinat titik: (4.0, 5.0, 6.0)
Jumlah objek Point yang telah dibuat: 2
Buku: Java Programming
Penulis: Fathir Ahmad Nurpadli
Nomor ISBN: 999-0123456789
Aplikasi Buku
Versi 1.0
Dikembangkan oleh: Fathir Ahmad Nurpadli
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>

```

*Gambar 29 Output dari Program KelasPB*

## II.6.C. Analisa

Kelas Point memiliki instance variables x, y, dan z, serta static variable count untuk menghitung jumlah objek Point yang telah dibuat. Konstruktor digunakan untuk menginisialisasi koordinat titik. Instance method displayCoordinates() menampilkan koordinat titik, sedangkan static method displayPointCount() menampilkan jumlah objek Point yang telah dibuat. Kelas Buku memiliki instance variables penulis, judul, dan nomorISBN. Konstruktor digunakan untuk menginisialisasi atribut-atribut buku. Instance method displayInfo() digunakan untuk menampilkan informasi tentang buku, sedangkan static method displayApplicationInfo() digunakan untuk menampilkan informasi tentang aplikasi buku.

## II.7 Tugas I-9 TestPerson.java

### II.7.A. Source Code

```

public class TestPerson {
    public static void main(String[] args) {
        Person dokter = new Person();
    }
}

```

```

        dokter.name = "Fathir Ahmad Nurpadli";

        dokter.gender = 'L';

        dokter.age = 20;

        dokter.dateOfBirth = "01/01/2000";

        dokter.height = 175.0; // Menggunakan titik sebagai
pemisah desimal

        dokter.weight = 85.0; // Menggunakan titik sebagai
pemisah desimal

        dokter.address = "Kabupaten Bandung Barat";


        dokter.cetakBiodata(dokter.name,        dokter.gender,
dokter.address);


        dokter.cetakFisik(dokter.age,        dokter.dateOfBirth,
dokter.height, dokter.weight);

    }

}

```

## II.7.B. Hasil

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>java TestPerson.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>java TestPerson
nama Fathir Ahmad Nurpadli ,
jenis kelamin L ,
Alamat Kabupaten Bandung Barat
Umur : 20
Tanggal Lahir : 01/01/2000
Tinggi : 175.0
Berat : 85.0
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>

```

*Gambar 30 Output dari Program TestPerson*

### II.7.C. Analisa

Program di atas adalah program Java yang digunakan untuk menguji kelas "Person". Pada awalnya, objek "dokter" dari kelas "Person" dibuat. Kemudian, atribut-atribut objek "dokter" seperti nama, jenis kelamin, usia, tanggal lahir, tinggi, berat badan, dan alamat diinisialisasi dengan nilai-nilai tertentu. Perlu dicatat bahwa nilai tinggi dan berat badan menggunakan tipe data double dan menggunakan titik sebagai pemisah desimal. Setelah inisialisasi, metode-metode cetakBiodata() dan cetakFisik() dipanggil pada objek "dokter" untuk mencetak informasi biodata dan informasi fisik.

## II.8 Tugas I-10b

### II.8.A. Source Code

```
import java.awt.Color;

public class LingkunganRumah {

    public static void main(String[] args) {

        Kucing michael = new Kucing();

        Kucing garfield = new Kucing();

        michael.warnaBulu = new Color(0, 1, 1);

        michael.nama = "Michael";

        michael.usia = 3;

        michael.bb = 4.5;

        michael.diadopsi("Rezki");

        // Memanggil metode cetakInformasi() untuk menampilkan
informasi kucing "michael"

        michael.cetakInformasi();

        garfield.warnaBulu = new Color(0, 2, 5);

        garfield.nama = "Garfield";
```

```

        garfield.usia = 5;

        garfield.bb = 5.0;

        garfield.diadopsi("Fathir");

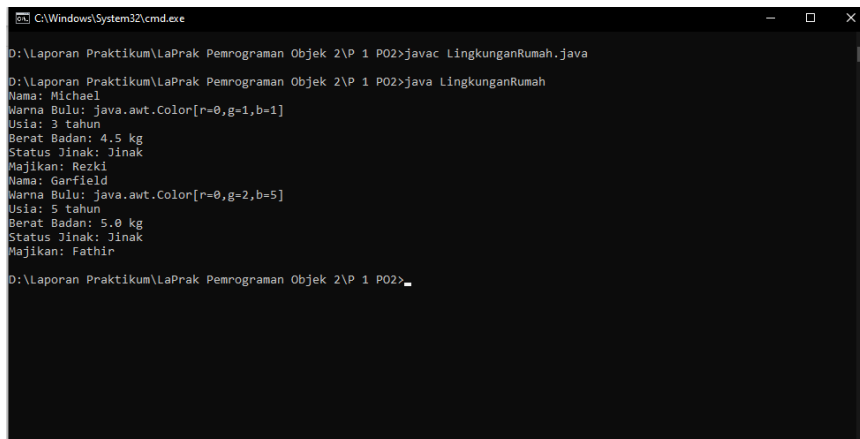

        garfield.cetakInformasi();

    }

}

```

## II.8.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac LingkunganRumah.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java LingkunganRumah
Nama: Michael
Warna Bulu: java.awt.Color[r=0,g=1,b=1]
Usia: 3 tahun
Berat Badan: 4.5 kg
Status Jinak: Jinak
Majikan: Rezeki
Nama: Garfield
Warna Bulu: java.awt.Color[r=2,g=2,b=5]
Usia: 5 tahun
Berat Badan: 5.0 kg
Status Jinak: Jinak
Majikan: Fathir
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>_

```

*Gambar 31 Output dari Program LingkunganRumah*

## II.8.C. Analisa

Program di atas adalah sebuah program dalam bahasa pemrograman Java yang menciptakan dua objek dari kelas "Kucing" yaitu "michael" dan "garfield". Pada objek "garfield", beberapa atribut seperti warna bulu, nama, usia, dan berat badan diinisialisasi dengan nilai tertentu, dan kemudian metode "diadopsi()" dipanggil untuk menetapkan nama majikan. Untuk menampilkan informasi lengkap tentang kucing "garfield", perlu ditambahkan pemanggilan metode "cetakInformasi()".

## II.9 Tugas I-11

1. a) Sebuah blueprint atau cetak biru untuk menciptakan objek. Class mendefinisikan atribut (data) dan metode (fungsi) yang akan dimiliki oleh setiap objek yang dibuat dari

kelas tersebut. Misalnya, dalam sebuah class "Mobil" kita dapat mendefinisikan atribut seperti merek, warna, dan kecepatan, serta metode seperti "maju" dan "mundur".

b) Objek adalah instance konkret dari suatu class. Ketika sebuah class dibuat, itu adalah hanya deskripsi umum dari objek. Namun, ketika Anda membuat sebuah objek dari class tersebut, Anda mengalokasikan memori untuk objek tersebut dan mengisi nilai-nilai dari atribut-atributnya. Misalnya, jika kita membuat sebuah objek "Mobil" dengan merek "Toyota", warna "Merah", dan kecepatan "50 km/jam", maka itu adalah sebuah instance konkret dari class "Mobil".

c) Instance variable adalah variabel yang nilainya unik untuk setiap objek yang dibuat dari suatu class. Setiap objek memiliki salinan sendiri dari setiap instance variable kelas tersebut. Dengan kata lain, setiap objek memiliki data yang berbeda. Misalnya, jika kita memiliki instance variable "merek", "warna", dan "kecepatan" dalam class "Mobil", maka setiap objek "Mobil" yang dibuat akan memiliki nilai yang berbeda untuk ketiga variabel tersebut tergantung pada cara objek tersebut diinisialisasi.

2. Program sudah di lengkapi pada BAB Hasil Praktikum dari Program I-11 hingga I-14

## II.10 Tugas I-13

### II.10.A. Source Code

```
public class Elevator {  
  
    public boolean doorOpen = false;  
  
    public int currentFloor = 1;  
  
    public final int TOP_FLOOR = 5;  
  
    public final int BOTTOM_FLOOR = 1;  
  
  
    public void openDoor() {  
  
        System.out.println("Opening door.");  
  
        doorOpen = true;  
  
        System.out.println("Door is open.");  
  
    }  
  
  
    public void closeDoor() {
```



```

        System.out.println("Closing door.");
        doorOpen = false;
        System.out.println("Door is closed.");
    }

    public void goUp() {
        System.out.println("Going up one floor.");
        currentFloor++;
        System.out.println("Floor: " + currentFloor);
    }

    public void goDown() {
        System.out.println("Going down one Floor.");
        currentFloor--;
        System.out.println("Floor: " + currentFloor);
    }

    public void setFloor(int desiredFloor) {
        if (desiredFloor >= BOTTOM_FLOOR && desiredFloor <=
TOP_FLOOR) {
            while (currentFloor != desiredFloor) {
                if(currentFloor < desiredFloor) {
                    goUp();
                } else {
                    goDown();
                }
            }
        }
    }

```

```

        }
    } else {
        System.out.println("Desired floor is out of
range.");
    }
}

public int getFloor() {
    return currentFloor;
}

public boolean checkDoorStatus() {
    return doorOpen;
}
}

public class ElevatorTest {
    public static void main(String []args) {
        Elevator myElevator = new Elevator();
        myElevator.openDoor();
        myElevator.closeDoor();
        myElevator.goUp();
        myElevator.goUp();
        myElevator.goUp();
        myElevator.openDoor();
        myElevator.closeDoor();
    }
}

```

```

        myElevator.goDown();
        myElevator.openDoor();
        myElevator.closeDoor();
        myElevator.goDown();
        myElevator.setFloor(myElevator.TOP_FLOOR);
        myElevator.openDoor();
    }
}

```

```

public class ElevatorTestTwo {
    public static void main(String []args) {
        Elevator myElevator = new Elevator();
        myElevator.openDoor();
        myElevator.closeDoor();
        myElevator.goUp();
        myElevator.goUp();
        myElevator.goUp();
        myElevator.openDoor();
        myElevator.closeDoor();
        myElevator.goDown();
        myElevator.openDoor();
        myElevator.closeDoor();
        myElevator.goDown();
        int curFloor = myElevator.getFloor();
        System.out.println("Current Floor: " + curFloor);
        myElevator.setFloor(curFloor + 1);
    }
}

```

```

        myElevator.openDoor();
    }
}

```

## II.10.B. Hasil

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>javac ElevatorTest.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>javac ElevatorTestTwo.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>java Elevator
Error: Main method not found in class Elevator, please define the main method as:
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>java ElevatorTest
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.

```

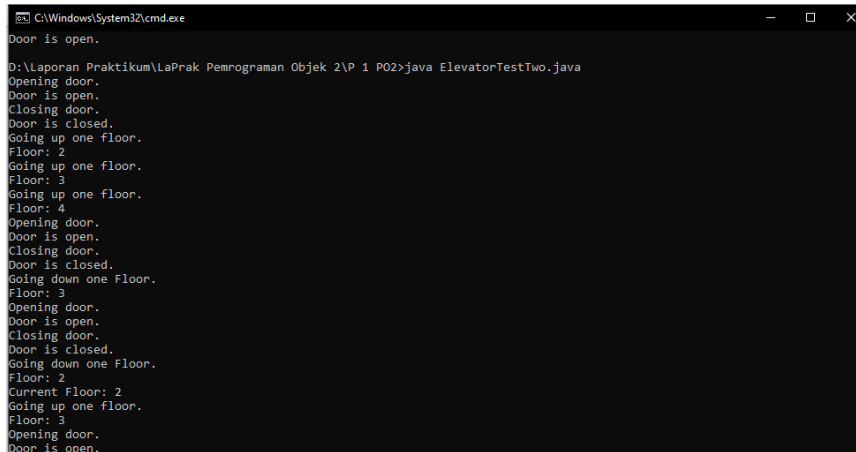
*Gambar 32 Output dari Program Elevator*

```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\1 P02>java ElevatorTest
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Going up one floor.
Floor: 5
Opening door.

```

*Gambar 33 Output dari Program ElevatorTest*



```
C:\Windows\System32\cmd.exe
Door is open.
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>java ElevatorTestTwo.java
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 2
Current Floor: 2
Going up one floor.
Floor: 3
Opening door.
Door is open.
```

*Gambar 34 Output dari Program ElevatorTestTwo*

### II.10.C. Analisa

Perbedaan utama antara kode sebelum dan sesudah penambahan adalah pada kelas `Elevator` pada bagian metode `setFloor(int desiredFloor)`. Pada kode sebelumnya, tidak ada pengecekan apakah lantai yang dimasukkan ke metode `setFloor()` berada dalam rentang lantai yang valid atau tidak. Namun, setelah penambahan, kondisi telah ditambahkan untuk memeriksa apakah lantai yang dimasukkan berada dalam rentang yang valid (`BOTTOM\_FLOOR` hingga `TOP\_FLOOR`). Jika lantai yang dimasukkan berada di luar rentang tersebut, akan dicetak pesan kesalahan.

## II.11 Tugas I-14

### II.11.A. Source Code

```
public class Elevator {

    private boolean doorOpen = false;

    private int currentFloor = 1;

    private final int TOP_FLOOR = 5;

    private final int BOTTOM_FLOOR = 1;

    public void openDoor() {

        System.out.println("Opening door.");

        doorOpen = true;
```

```

        System.out.println("Door is open.");
    }

    public void closeDoor() {
        System.out.println("Closing door.");
        doorOpen = false;
        System.out.println("Door is closed.");
    }

    public void goUp() {
        System.out.println("Going up one floor.");
        currentFloor++;
        System.out.println("Floor: " + currentFloor);
    }

    public void goDown() {
        System.out.println("Going down one Floor.");
        currentFloor--;
        System.out.println("Floor: " + currentFloor);
    }

    public void setFloor(int desiredFloor) {
        if (desiredFloor >= BOTTOM_FLOOR && desiredFloor <=
TOP_FLOOR) {
            while (currentFloor != desiredFloor) {
                if(currentFloor < desiredFloor) {

```

```

        goUp();
    } else {
        goDown();
    }
}

} else {
    System.out.println("Desired floor is out of
range.");
}

}

public int getFloor() {
    return currentFloor;
}

public boolean isDoorOpen() {
    return doorOpen;
}

}

```

#### II.11.B. Hasil

```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java ElevatorTest
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Going up one floor.
Floor: 5
```

*Gambar 35 Output dari Program ElevatorTest*

```
C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java ElevatorTestTwo.java
Opening door.
Door is open.
Closing door.
Door is closed.
Going up one floor.
Floor: 2
Going up one floor.
Floor: 3
Going up one floor.
Floor: 4
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 3
Opening door.
Door is open.
Closing door.
Door is closed.
Going down one floor.
Floor: 2
Current Floor: 2
Going up one floor.
Floor: 3
Opening door.
Door is open.
```

*Gambar 36 Output dari Program ElevatorTestTwo*

### II.11.C. Analisa

Pada program di atas instance variable doorOpen dan currentFloor telah diubah menjadi private untuk menerapkan prinsip enkapsulasi. Hal ini memungkinkan akses terhadap variabel hanya melalui metode-metode publik yang disediakan oleh kelas Elevator.

## II.12 Tugas I-15

### II.12.A. Source Code

```
public class StudentRecord {

    private String name;

    private static int studentCount = 0;

    // Konstruktor tanpa parameter
```



```

public StudentRecord() {
    studentCount++;
}

// Konstruktor dengan parameter nama
public StudentRecord(String name) {
    this.name = name;
    studentCount++;
}

// Konstruktor dengan parameter nama dan nim
public StudentRecord(String name, int nim) {
    this.name = name;
    studentCount++;
}

// Konstruktor dengan parameter nama, nim, dan jurusan
public StudentRecord(String name, int nim, String jurusan)
{
    this.name = name;
    studentCount++;
}

// Konstruktor dengan parameter nama dan alamat
public StudentRecord(String name, String address) {
    this.name = name;

```

```

        studentCount++;
    }

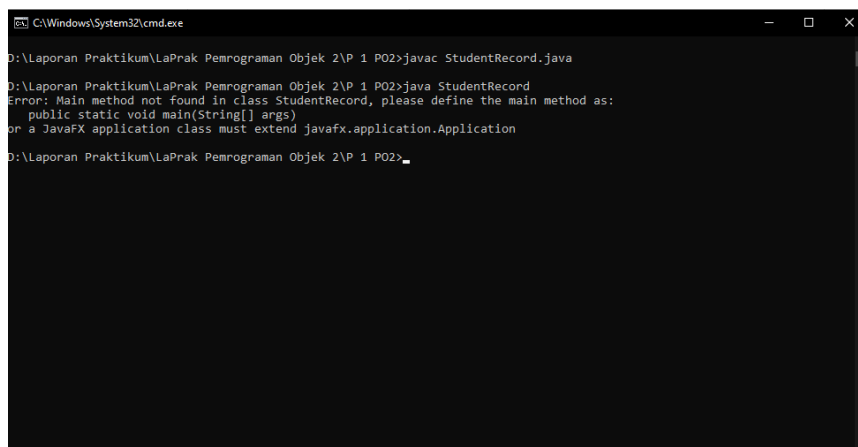
    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public static int getStudentCount() {
        return studentCount;
    }
}

```

## II.12.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>javac StudentRecord.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>java StudentRecord
Error: Main method not found in class StudentRecord, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 PO2>_

```

*Gambar 37 Output dari Program StudentRecord*

## II.12.C. Analisa

Pada program di atas, terdapat 4 konstruktor yang berbeda, masing-masing menerima parameter dengan jumlah dan tipe data yang berbeda. Setiap kali salah satu dari konstruktor tersebut dipanggil untuk membuat objek baru, variabel `studentCount` akan bertambah, sehingga kita dapat melacak jumlah total objek yang telah dibuat menggunakan metode `getStudentCount()`.

## II.13 Tugas Akhir `ComputerStudentRecord.java`

### II.13.A. Source Code

```
public class ComputerStudentRecord extends StudentRecord {  
    private String major;  
  
    public ComputerStudentRecord() {  
        super(); // Memanggil konstruktor dari kelas induk  
    }  
  
    // Overriding metode setName dari kelas induk  
    @Override  
    public void setName(String name) {  
        super.setName(name); // Memanggil metode setName dari  
        kelas induk  
        System.out.println("Computer Student Name set to: " +  
        name);  
    }  
  
    // Overriding metode getName dari kelas induk  
    @Override  
    public String getName() {  
        String studentName = super.getName(); // Memanggil  
        metode getName dari kelas induk  
    }  
}
```

```

        System.out.println("Computer Student Name retrieved: "
+ studentName);

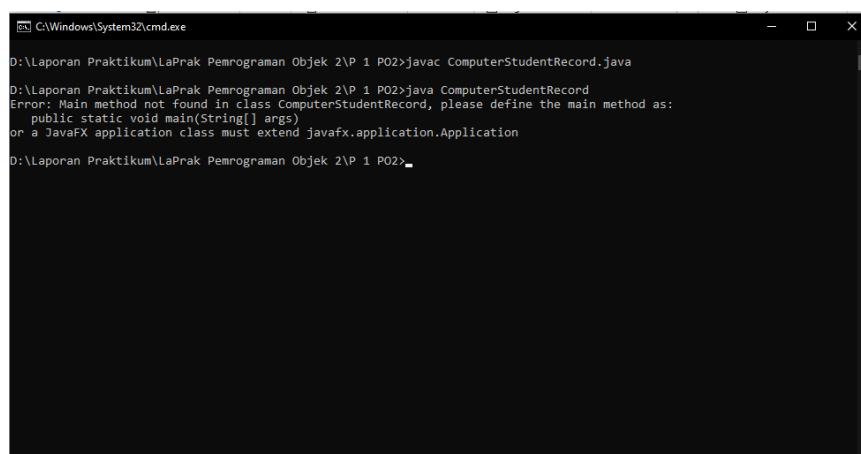
        return studentName;
    }

    // Menambahkan metode untuk mengatur jurusan
    public void setMajor(String major) {
        this.major = major;
    }

    // Menambahkan metode untuk mendapatkan jurusan
    public String getMajor() {
        return major;
    }
}

```

### II.13.B. Hasil



The screenshot shows a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The command entered is `D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac ComputerStudentRecord.java`. The output shows the command being executed, followed by an error message: `Error: Main method not found in class ComputerStudentRecord, please define the main method as: public static void main(String[] args) or a JavaFX application class must extend javafx.application.Application`. The prompt then returns to `D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>_`.

*Gambar 38 Output dari Program ComputerStudentRecord*

### II.13.C. Analisa

Kelas di atas merupakan turunan dari kelas `StudentRecord` yang memperluas fungsionalitasnya dengan menambahkan atribut `major` untuk menyimpan informasi jurusan mahasiswa. Konstruktor kelas ini memanggil konstruktor kelas induk menggunakan kata kunci `super()`. Metode `setName` di-overriding untuk memperluas fungsionalitas dari metode dengan menampilkan pesan saat nama mahasiswa diatur. Begitu pula dengan metode `getName` yang di-overriding untuk menampilkan pesan saat nama mahasiswa diambil. Selain itu, kelas ini juga menambahkan metode `setMajor` untuk mengatur jurusan dan `getMajor` untuk mendapatkan nilai jurusan.

## II.14 Tugas Akhir Abstract Class Shape

### II.14.A. Source Code

```
public abstract class Shape {  
    public abstract double getArea();  
    public abstract String getName();  
}  
  
public class Circle extends Shape {  
    private double radius;  
  
    public Circle(double radius) {  
        this.radius = radius;  
    }  
  
    @Override  
    public double getArea() {  
        return Math.PI * radius * radius;  
    }  
  
    @Override
```

```
public String getName() {  
    return "Circle";  
}  
  
// Metode tambahan untuk mendapatkan radius  
public double getRadius() {  
    return radius;  
}  
}
```

```
public class Square extends Shape {  
    private double side;  
  
    public Square(double side) {  
        this.side = side;  
    }
```

```
@Override  
public double getArea() {  
    return side * side;  
}
```

```
@Override  
public String getName() {  
    return "Square";  
}
```

```

// Metode tambahan untuk mendapatkan panjang sisi

public double getSide() {

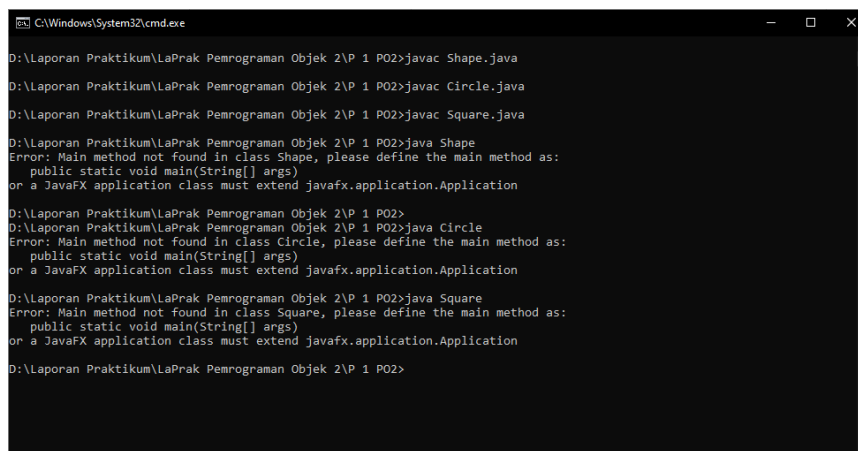
    return side;

}

}

```

## II.14.B. Hasil



```

C:\Windows\System32\cmd.exe
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Shape.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Circle.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>javac Square.java
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Shape
Error: Main method not found in class Shape, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Circle
Error: Main method not found in class Circle, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>java Square
Error: Main method not found in class Square, please define the main method as:
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
D:\Laporan Praktikum\LaPrak Pemrograman Objek 2\P 1 P02>

```

*Gambar 39 Output dari Program Abstract Class Shape*

## II.14.C. Analisa

Program di atas merupakan implementasi polimorfisme dan abstraksi dalam pemrograman berorientasi objek. Kelas `Shape` adalah kelas abstrak yang memiliki dua metode abstrak `getArea()` dan `getName()`, yang mewakili fungsi umum dari berbagai bentuk geometris. Kelas `Circle` dan `Square` merupakan turunan dari kelas `Shape`, masing-masing merepresentasikan lingkaran dan persegi. Kedua kelas turunan ini mengimplementasikan metode `getArea()` dan `getName()` sesuai dengan karakteristik bentuk geometris yang mereka wakili. Di samping itu, kelas `Circle` memiliki metode tambahan `getRadius()` untuk mendapatkan nilai radius lingkaran, sementara kelas `Square` memiliki metode tambahan `getSide()` untuk mendapatkan nilai panjang sisi persegi.

### **BAB III. KESIMPULAN**

Pada Pertemuan kali ini me review Kembali praktikum dari semester sebelumnya. Dimana mempelajari berbagai Variable, method dan sebagainya. Pada Pertemuan kali ini juga membahas Kembali tentang inheritance / keturunan, Polimorfisme, Abstract Class, dan Interface.