

**LAPORAN TUGAS BESAR
PEMROGRAMAN BERORIENTASI OBJEK
KONVERSI SUHU**



Disusun Oleh :

NAMA : FAREL IQBAL MAHARDIKA

NIM : 32602200010

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM SULTAN AGUNG
SEMARANG**

2022

DAFTAR ISI

HALAMAN JUDUL.....	1
DAFTAR ISI.....	2
DAFTAR GAMBAR	3
BAB I Pendahuluan.....	4
1.1 Latar belakang	4
1.2 Tujuan	4
1.3 Manfaat	4
BAB II Konsep Dasar PBO.....	5
2.1 Inheritance	5
2.2 Polimorfisme	5
2.3 Encapsulation	5
2.4 Getter dan Setter	5
2.5 Interface	5
BAB III Struktur Program.....	6
3.1 File Utama (TemperatureMain)	6
3.2 Celcius Converter	7
3.3 FahrenheitConverter	9
3.4 KelvinConverter	10
3.5 Interface TemperatureConverter	11
BAB IV Implementasi Program	13
4.1 Menjalankan Program	13
4.2 Mengoperasikan Konversi Suhu	14
BAB V Kesimpulan	16
DAFTAR PUSTAKA	18

DAFTAR GAMBAR

Gambar 1. 1 File TemperatureMain 1	6
Gambar 1. 2 File TemperatureMain 2	6
Gambar 1. 7 Kelas CelciusConverter 1	7
Gambar 1. 8 Kelas CelciusConverter 2	8
Gambar 1. 9 Kelas FahrenheitConverter	9
Gambar 1. 9 Kelas KelvinConverter	10
Gambar 1. 13 Interface TemperatureConverter	11
Gambar 2. 1 Open Neatbeans.....	13
Gambar 2. 2 Open Projects	13
Gambar 2. 3 Arahkan ke project	13
Gambar 2. 4 Open struktur project	14
Gambar 2. 5 Run program	14
Gambar 2. 6 Tunggu program muncul	14
Gambar 2. 7 Masukan nilai	14
Gambar 2. 8 Hasil konversi	15

BAB I Pendahuluan

1.1 Latar belakang

Dalam era perkembangan teknologi, pemrograman berorientasi objek (PBO) menjadi konsep yang sangat penting dalam pengembangan perangkat lunak. Salah satu konsep utama dalam PBO adalah inheritance, polymorphism, encapsulation, getter dan setter, serta penggunaan interface. Laporan ini akan menjelaskan implementasi konsep-konsep ini dalam sebuah program sederhana konversi suhu.

1.2 Tujuan

Laporan ini bertujuan untuk memberikan pemahaman tentang penggunaan konsep-konsep PBO dalam pengembangan program konversi suhu. Selain itu, laporan ini juga bertujuan untuk memberikan panduan praktis dalam menjalankan program dan menambahkan fungsionalitas baru.

1.3 Manfaat

Manfaat dari laporan ini adalah memberikan pemahaman yang lebih baik tentang konsep-konsep PBO dan bagaimana mengimplementasikannya dalam sebuah program. Hal ini dapat membantu pembaca untuk meningkatkan keterampilan pemrograman berorientasi objek mereka.

BAB II Konsep Dasar PBO

2.1 Inheritance

Inheritance adalah konsep yang memungkinkan sebuah kelas untuk mewarisi properti dan metode dari kelas lain. Dalam program konversi suhu, kita menggunakan inheritance untuk membuat kelas-kelas konverter suhu yang mewarisi metode dari sebuah antarmuka.

2.2 Polimorfisme

Polimorfisme memungkinkan objek untuk digunakan dengan cara yang lebih umum, tanpa harus mengetahui tipe khususnya. Dalam konversi suhu, kita memanfaatkan polimorfisme dengan membuat objek-objek dari kelas yang berbeda namun dapat digunakan secara seragam.

2.3 Encapsulation

Encapsulation adalah konsep untuk menyembunyikan rincian implementasi dan hanya mengekspos fungsionalitas yang diperlukan. Dalam program ini, kita menggunakan encapsulation dengan membuat kelas-kelas konverter suhu yang memiliki metode-metode internal yang tersembunyi.

2.4 Getter dan Setter

Getter dan setter adalah metode untuk mendapatkan dan mengatur nilai properti objek. Dalam kelas konverter suhu, kita menggunakan getter dan setter untuk mengakses dan mengubah nilai suhu.

2.5 Interface

Interface adalah kontrak untuk kelas-kelas yang akan mengimplementasikannya. Dalam program ini, kita menggunakan antarmuka `TemperatureConverter` untuk menetapkan kontrak yang harus dipenuhi oleh kelas-kelas konverter suhu.

BAB III Struktur Program

3.1 File Utama (TemperatureMain)

File utama berisi logika untuk menjalankan program dan berinteraksi dengan pengguna melalui konsol. Program ini menciptakan objek-objek dari kelas konverter suhu dan menampilkan hasil konversi ke konsol.

```
1 package konversisuhu.farel_igbal;
2
3
4 /*
5  * author : Farel Iqbal Mahardika
6  * nim : 52602200019
7  * berikan penjelasan kode ini baris perbaris dengan komentar
8  */
9
10 import javax.swing.*;
11 import java.awt.event.ActionEvent;
12 import java.awt.event.ActionListener;
13 import java.util.Scanner;
14
15 import javax.swing.*;
16
17 public class TemperatureMain {
18
19     public static void main(String[] args) {
20         // Membuat objek Scanner untuk membaca input dari pengguna
21         Scanner scanner = new Scanner(System.in);
22
23         // Meminta pengguna memasukkan suhu dalam Celsius
24         System.out.print("Enter temperature in Celsius: ");
25         double inputValue = scanner.nextDouble();
26
27         // Membuat instance (objek) dari kelas CelciusConverter, FahrenheitConverter, dan KelvinConverter
28         CelciusConverter celsiusConverter = new CelciusConverter(celsius: inputValue);
29         FahrenheitConverter fahrenheitConverter = new FahrenheitConverter(fahrenheit: inputValue);
30         KelvinConverter kelvinConverter = new KelvinConverter(kelvin: inputValue);
31     }
32 }
```

Gambar 1. 1 File TemperatureMain 1

```
1 // Membuat instance (objek) dari kelas CelciusConverter, FahrenheitConverter, dan KelvinConverter
2 CelciusConverter celsiusConverter = new CelciusConverter(celsius: inputValue);
3 FahrenheitConverter fahrenheitConverter = new FahrenheitConverter(fahrenheit: inputValue);
4 KelvinConverter kelvinConverter = new KelvinConverter(kelvin: inputValue);
5
6 // Menampilkan hasil konversi ke layar konsol
7 System.out.println("Celsius: " + celsiusConverter.convert(input: inputValue));
8 System.out.println("Fahrenheit: " + fahrenheitConverter.convert(input: inputValue));
9 System.out.println("Kelvin: " + kelvinConverter.convert(input: inputValue));
10
11 }
```

Gambar 1. 2 File TemperatureMain 2

Penjelasan:

1. Import Statements: Baris pertama hingga keenam adalah import statements yang digunakan untuk mengimpor kelas-kelas dari paket Java, termasuk kelas JOptionPane dan Scanner yang akan digunakan dalam program.
2. Class Declaration: public class TemperatureMain { Mendeklarasikan kelas TemperatureMain. Ini adalah kelas utama yang akan berisi metode main untuk menjalankan program.
3. Main Method: public static void main(String[] args) { Metode utama di

Java yang akan dieksekusi saat program dijalankan. Di dalamnya, kita membuat objek Scanner untuk membaca input dari pengguna.

4. Input Celsius: `System.out.print("Enter temperature in Celsius: ");`
Menampilkan pesan untuk meminta pengguna memasukkan suhu dalam Celsius.
5. Read Input: `double inputValue = scanner.nextDouble();` Membaca input suhu dari pengguna dan menyimpannya dalam variabel `inputValue`.
6. Create Converters: Membuat instance dari kelas `CelciusConverter`, `FahrenheitConverter`, dan `KelvinConverter` menggunakan nilai input yang telah dimasukkan pengguna.
7. Display Results: Menampilkan hasil konversi ke layar konsol menggunakan `System.out.println()`.

3.2 Celcius Converter

Kelas `CelsiusConverter` adalah implementasi dari antarmuka `TemperatureConverter` untuk konversi suhu ke dalam satuan Celsius.

```
package konversisuhu.farel_igbal;

/*
author : Farel Iqbal Mahardika
nim : 32602200010
*/

// Berikan penjelasan kode ini baris perbaris dengan komentar, bagian polimorfisme, getter setter, constructor, inheritance
public class CelsiusConverter implements TemperatureConverter {
    // Deklarasi variabel private untuk menyimpan suhu dalam Celsius
    private double celsius;

    // Konstruktor untuk inisialisasi objek dengan nilai suhu dalam Celsius
    public CelsiusConverter(double celsius) {
        this.celsius = celsius;
    }

    // Implementasi metode convert dari antarmuka TemperatureConverter
    @Override
    public double convert(double input) {
        return input; // Mengembalikan nilai input tanpa perubahan, tidak ada konversi yang dilakukan
    }

    // Implementasi metode getUnit dari antarmuka TemperatureConverter
    @Override
    public String getUnit() {
        return "Celsius"; // Mengembalikan satuan suhu yang diterima oleh kelas ini
    }

    // Getter untuk mendapatkan nilai suhu dalam Celsius
    public double getCelsius() {
```

Gambar 1. 3 Kelas CelciusConverter 1

```

// Getter untuk mendapatkan nilai suhu dalam Celsius
public double getCelsius() {
    return celsius;
}

// Setter untuk mengatur nilai suhu dalam Celsius
public void setCelsius(double celsius) {
    this.celsius = celsius;
}
}

```

Gambar 1. 4 Kelas CelciusConverter 2

Penjelasan:

1. Deklarasi Kelas: `public class CelciusConverter implements TemperatureConverter` { Mendeklarasikan kelas CelciusConverter yang mengimplementasikan antarmuka TemperatureConverter.
2. Variabel Private: `private double celsius;` Mendeklarasikan variabel celsius sebagai variabel instance private yang menyimpan nilai suhu dalam Celsius.
3. Konstruktor: `public CelciusConverter(double celsius) { this.celsius = celsius; }` Konstruktor kelas ini untuk menginisialisasi objek dengan nilai suhu dalam Celsius yang diberikan pada saat pembuatan objek.
4. Override Metode convert: `@Override public double convert(double input) { return input; }` Mengimplementasikan metode convert dari antarmuka TemperatureConverter. Dalam hal ini, metode ini hanya mengembalikan nilai input tanpa mengubahnya, sehingga tidak ada konversi yang dilakukan.
5. Override Metode getUnit: `@Override public String getUnit() { return "Celsius"; }` Mengimplementasikan metode getUnit dari antarmuka TemperatureConverter. Metode ini mengembalikan string "Celsius" sebagai satuan suhu yang digunakan oleh objek ini.
6. Getter getCelsius: `public double getCelsius() { return celsius; }` Memberikan akses untuk mendapatkan nilai suhu dalam Celsius.
7. Setter setCelsius: `public void setCelsius(double celsius) { this.celsius = celsius; }` Memberikan akses untuk mengatur nilai suhu dalam Celsius.

3.3 FahrenheitConverter

Kelas FahrenheitConverter adalah implementasi dari antarmuka TemperatureConverter untuk konversi suhu ke dalam satuan Fahrenheit..

```
public class FahrenheitConverter implements TemperatureConverter {  
    // Deklarasi variabel private untuk menyimpan suhu dalam Fahrenheit  
    private double fahrenheit;  
  
    // Konstruktor untuk inialisasi objek dengan nilai suhu dalam Fahrenheit  
    public FahrenheitConverter(double fahrenheit) {  
        this.fahrenheit = fahrenheit;  
    }  
  
    // Implementasi metode convert dari antarmuka TemperatureConverter  
    @Override  
    public double convert(double input) {  
        // Melakukan konversi suhu dari Fahrenheit ke Celsius  
        return (input - 32) * 5 / 9;  
    }  
  
    // Implementasi metode getUnit dari antarmuka TemperatureConverter  
    @Override  
    public String getUnit() {  
        return "Fahrenheit"; // Mengembalikan satuan suhu yang diterima oleh kelas ini  
    }  
  
    // Getter untuk mendapatkan nilai suhu dalam Fahrenheit  
    public double getFahrenheit() {  
        return fahrenheit;  
    }  
  
    // Setter untuk mengatur nilai suhu dalam Fahrenheit  
    public void setFahrenheit(double fahrenheit) {  
        this.fahrenheit = fahrenheit;  
    }  
}
```

Gambar 1. 5 Kelas FahrenheitConverter

Penjelasan:

1. Deklarasi Kelas: `public class FahrenheitConverter implements TemperatureConverter {` Mendeklarasikan kelas FahrenheitConverter yang mengimplementasikan antarmuka TemperatureConverter.
2. Variabel Private: `private double fahrenheit;` Mendeklarasikan variabel fahrenheit sebagai variabel instance private yang menyimpan nilai suhu dalam Fahrenheit.
3. Konstruktor: `public FahrenheitConverter(double fahrenheit) { this.fahrenheit = fahrenheit; }` Konstruktor kelas ini untuk menginisialisasi objek dengan nilai suhu dalam Fahrenheit yang diberikan pada saat pembuatan objek.
4. Override Metode convert: `@Override public double convert(double input) { return (input - 32) * 5 / 9; }` Mengimplementasikan metode convert dari antarmuka TemperatureConverter. Metode ini melakukan konversi suhu dari Fahrenheit ke Celsius.
5. Override Metode getUnit: `@Override public String getUnit() { return "Fahrenheit"; }` Mengimplementasikan metode getUnit dari antarmuka

TemperatureConverter. Metode ini mengembalikan string "Fahrenheit" sebagai satuan suhu yang digunakan oleh objek ini.

6. Getter getFahrenheit: `public double getFahrenheit() { return fahrenheit; }`
Memberikan akses untuk mendapatkan nilai suhu dalam Fahrenheit.
7. Setter setFahrenheit: `public void setFahrenheit(double fahrenheit) { this.fahrenheit = fahrenheit; }` Memberikan akses untuk mengatur nilai suhu dalam Fahrenheit.

3.4 KelvinConverter

Kelas KelvinConverter adalah implementasi dari antarmuka TemperatureConverter untuk konversi suhu ke dalam satuan Kelvin.

```
public class KelvinConverter implements TemperatureConverter {  
    // Deklarasi variabel private untuk menyimpan suhu dalam Kelvin  
    private double kelvin;  
  
    // Konstruktor untuk inisialisasi objek dengan nilai suhu dalam Kelvin  
    public KelvinConverter(double kelvin) {  
        this.kelvin = kelvin;  
    }  
  
    // Implementasi metode convert dari antarmuka TemperatureConverter  
    @Override  
    public double convert(double input) {  
        // Melakukan konversi suhu dari Kelvin ke Celsius  
        return input - 273.15;  
    }  
  
    // Implementasi metode getUnit dari antarmuka TemperatureConverter  
    @Override  
    public String getUnit() {  
        return "Kelvin"; // Mengembalikan satuan suhu yang diterima oleh kelas ini  
    }  
  
    // Getter untuk mendapatkan nilai suhu dalam Kelvin  
    public double getKelvin() {  
        return kelvin;  
    }  
  
    // Setter untuk mengatur nilai suhu dalam Kelvin  
    public void setKelvin(double kelvin) {  
        this.kelvin = kelvin;  
    }  
}
```

Gambar 1. 6 Kelas KelvinConverter

Penjelasan:

1. Deklarasi Kelas: `public class KelvinConverter implements TemperatureConverter {` Mendeklarasikan kelas KelvinConverter yang mengimplementasikan antarmuka TemperatureConverter.
2. Variabel Private: `private double kelvin;` Mendeklarasikan variabel kelvin sebagai variabel instance private yang menyimpan nilai suhu dalam Kelvin.

3. Konstruktor: `public KelvinConverter(double kelvin) { this.kelvin = kelvin; }` Konstruktor kelas ini untuk menginisialisasi objek dengan nilai suhu dalam Kelvin yang diberikan pada saat pembuatan objek.
4. Override Metode `convert`: `@Override public double convert(double input) { return input - 273.15; }` Mengimplementasikan metode `convert` dari antarmuka `TemperatureConverter`. Metode ini melakukan konversi suhu dari Kelvin ke Celsius.
5. Override Metode `getUnit`: `@Override public String getUnit() { return "Kelvin"; }` Mengimplementasikan metode `getUnit` dari antarmuka `TemperatureConverter`. Metode ini mengembalikan string "Kelvin" sebagai satuan suhu yang digunakan oleh objek ini.
6. Getter `getKelvin`: `public double getKelvin() { return kelvin; }` Memberikan akses untuk mendapatkan nilai suhu dalam Kelvin.
7. Setter `setKelvin`: `public void setKelvin(double kelvin) { this.kelvin = kelvin; }` Memberikan akses untuk mengatur nilai suhu dalam Kelvin.

3.5 Interface TemperatureConverter

Antarmuka `TemperatureConverter` adalah kontrak yang menyatakan bahwa setiap kelas yang mengimplementasikannya harus menyediakan metode untuk konversi suhu dan mendapatkan satuan suhu.

```
package konversisuhu.farel_igbal;

/*
author : Farel Iqbal Mahardika
nim : 32602200010
Berikan penjelasan kode ini baris perbaris dengan komentar, bagian interface
*/

// Deklarasi antarmuka TemperatureConverter
public interface TemperatureConverter {
    // Metode untuk mengkonversi suhu
    double convert(double input);

    // Metode untuk mendapatkan satuan suhu
    String getUnit();
}
```

Gambar 1. 7 Interface TemperatureConverter

Penjelasan:

1. Komentar Lisensi: `/* Click`

`nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt`

to change this license */ dan /* Click

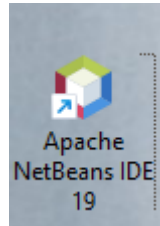
nbfs://nbhost/SystemFileSystem/Templates/Classes/Interface.java to edit this template */ adalah komentar yang menyertakan informasi tentang lisensi dan template default. Komentar ini tidak mempengaruhi jalannya program dan digunakan untuk dokumentasi.

2. Package Declaration: package konversisuhu.farel_iqbal;
Mendeklarasikan bahwa antarmuka ini berada dalam paket konversisuhu.farel_iqbal.
3. Komentar Pengembang: /* author : Farel Iqbal Mahardika nim : 32602200010 */ Komentar yang menyertakan informasi tentang pengembang (author) dan nomor induk mahasiswa (nim).
4. Deklarasi Antarmuka: public interface TemperatureConverter {
Mendeklarasikan antarmuka TemperatureConverter. Antarmuka adalah suatu bentuk kontrak yang menyatakan metode-metode yang harus diimplementasikan oleh kelas-kelas yang menggunakannya.
5. Metode convert: double convert(double input); Mendeklarasikan metode convert yang harus diimplementasikan oleh kelas-kelas yang menggunakan antarmuka ini. Metode ini bertanggung jawab untuk mengkonversi suhu dan mengembalikan hasil konversi dalam bentuk double.
6. Metode getUnit: String getUnit(); Mendeklarasikan metode getUnit yang harus diimplementasikan. Metode ini bertanggung jawab untuk mengembalikan satuan suhu dalam bentuk String.

BAB IV Implementasi Program

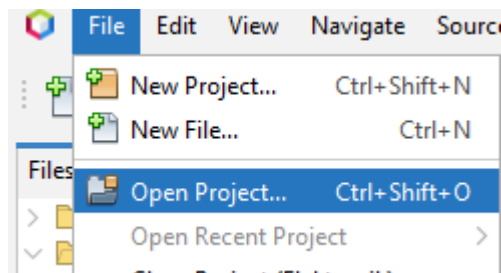
4.1 Menjalankan Program

1. Untuk menjalankan program klik neatbeans

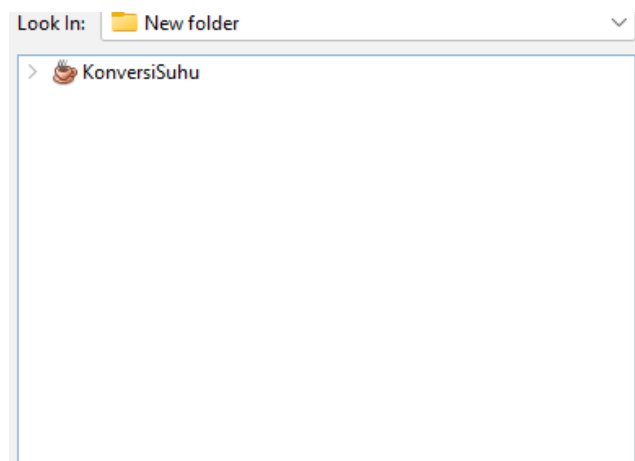


Gambar 2. 1 Open Neatbeans

2. Open new project, arahkan ke Konversi suhu

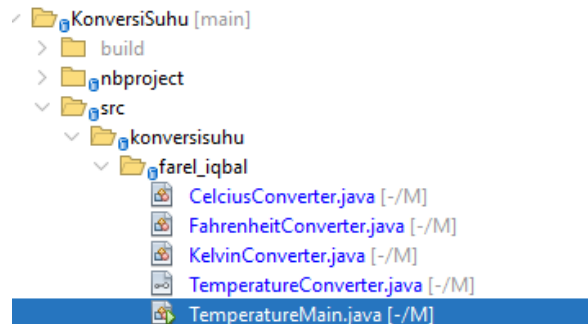


Gambar 2. 2 Open Projects



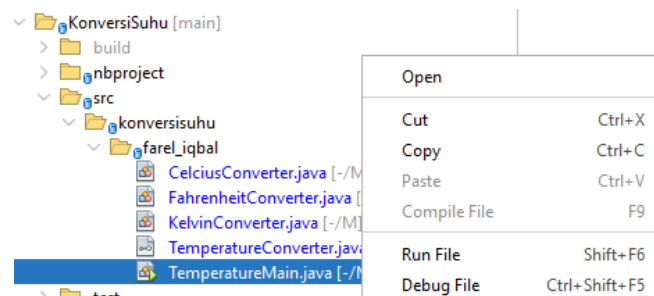
Gambar 2. 3 Arahkan ke project

3. Buka src/konversisuhu/farel_iqbal



Gambar 2. 4 Open struktur project

4. Klik file TemperatureMain.java, lalu klik kanan run file atau shift + f6



Gambar 2. 5 Run program

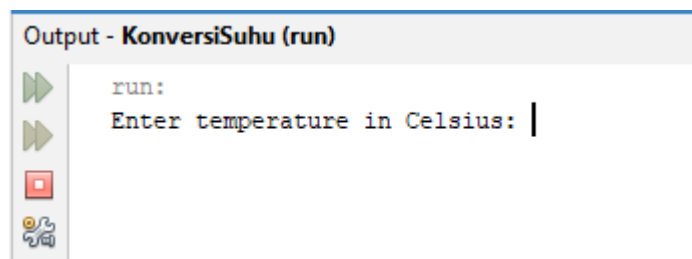
5. Tunggu program muncul



Gambar 2. 6 Tunggu program muncul

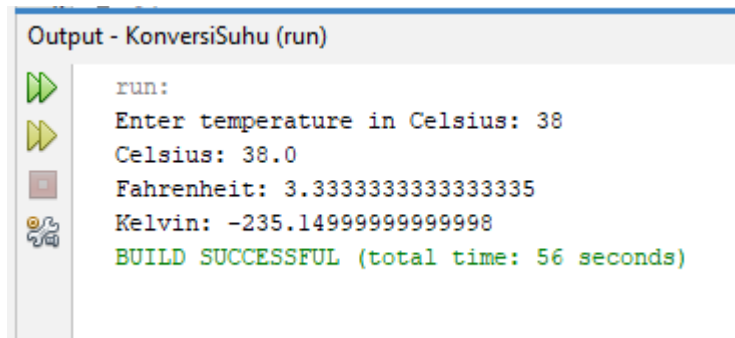
4.2 Mengoperasikan Konversi Suhu

1. Isi berapa nilai temperature dalam celcius



Gambar 2. 7 Masukan nilai

2. Lalu setelah di isi dan di enter, akan keluar konversi nilai dari celcius ke Fahrenheit dan kelvin



```
run:
Enter temperature in Celsius: 38
Celsius: 38.0
Fahrenheit: 3.333333333333335
Kelvin: -235.14999999999998
BUILD SUCCESSFUL (total time: 56 seconds)
```

Gambar 2. 8 Hasil konversi

BAB V Kesimpulan

Dalam pengembangan aplikasi konversi suhu dengan menggunakan Java dan NetBeans, beberapa konsep dasar pemrograman berorientasi objek (PBO) telah diterapkan. Berikut adalah beberapa poin kesimpulan:

1. Penggunaan Antarmuka (Interface):
 - a. Antarmuka `TemperatureConverter` dibuat untuk menetapkan kontrak metode yang harus diimplementasikan oleh kelas-kelas konversi suhu.
 - b. Antarmuka memungkinkan implementasi polimorfisme, di mana objek dari berbagai kelas dapat dianggap sebagai objek dari antarmuka, memungkinkan pemanggilan metode tanpa mengetahui implementasi sebenarnya.
2. Inheritance (Warisan):
 - a. Kelas konverter suhu (`CelsiusConverter`, `FahrenheitConverter`, dan `KelvinConverter`) menggunakan inheritance untuk mengambil sifat dan metode dari antarmuka `TemperatureConverter`.
 - b. Warisan memungkinkan kelas-kelas tersebut untuk menggunakan metode yang sama yang didefinisikan dalam antarmuka.
3. Encapsulation (Enkapsulasi):
 - a. Enkapsulasi diterapkan dengan mendeklarasikan variabel sebagai `private` di dalam kelas-kelas konverter suhu.
 - b. Setter dan getter digunakan untuk mengakses dan mengubah nilai variabel tersebut, memastikan kontrol akses yang baik terhadap data.
4. Getter dan Setter:
 - a. Setiap kelas konverter suhu memiliki metode getter dan setter untuk mengakses dan mengatur nilai suhu yang dienkapsulasi.
 - b. Getter dan setter memberikan kontrol akses terhadap data dan mencegah penggunaan langsung terhadap variabel `private`.
5. Polimorfisme:
 - a. Polimorfisme terlihat dalam penggunaan antarmuka, di mana objek dari

berbagai kelas dapat dianggap sebagai objek dari antarmuka yang sama.

- b. Hal ini memungkinkan pemanggilan metode `convert` dan `getUnit` dari berbagai kelas dengan cara yang seragam.

6. Pemisahan Kode (Code Separation):

- a. Kode aplikasi dibagi menjadi beberapa kelas dan antarmuka untuk meningkatkan keterbacaan, pemeliharaan, dan pengelolaan kode..

DAFTAR PUSTAKA

- (Czajkowski and Von Eicken, 1998; Nasser, Counsell and Shepperd, 2010; Perry, 2010)Czajkowski, G. and Von Eicken, T. (1998) 'JRes: A Resource Accounting Interface for Java', *SIGPLAN Notices (ACM Special Interest Group on Programming Languages)*, 33(10), pp. 21–35. Available at: <https://doi.org/10.1145/286942.286944>.
- Nasser, E., Counsell, S. and Shepperd, M. (2010) 'Class movement and re-location: An empirical study of Java inheritance evolution', *Journal of Systems and Software*, 83(2), pp. 303–315. Available at: <https://doi.org/10.1016/j.jss.2009.08.011>.
- Perry, J.S. (2010) *Table of content, IEEE Aerospace and Electronic Systems Magazine*. Available at: <https://doi.org/10.1109/maes.2010.5638796>.