TEB1043 Object Oriented Programming

January 2023 Semester

Department of Computer and Information Sciences

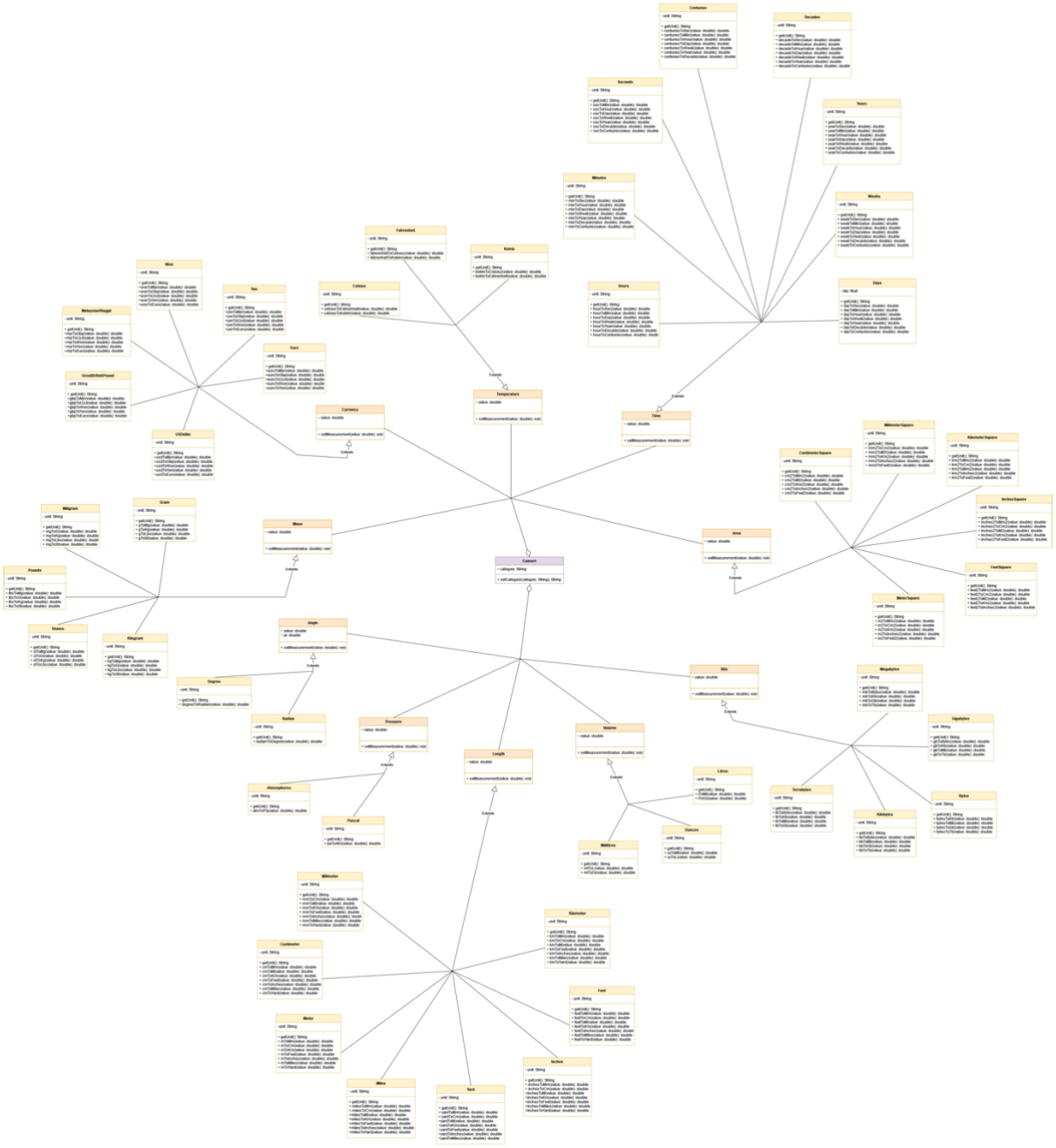
Prepared for:

Dr Mohamed Nordin bin Zakaria

Prepared by:

Name	Student ID	Program
Ahmad Zafri bin Mohd Ramli	21000556	Bachelor of Computer Science
Iman Kamil bin Suhaimi	21000175	Bachelor of Computer Science
Ammar Afif bin Mohd Khairie	21001232	Bachelor of Computer Science
Muhammad Aqil Fauzan bin Muhammad Hafidzi	21001113	Bachelor of Computer Science
Mohamad Izhar bin Ahmad Nordin	21001242	Bachelor of Computer Science
Danish Hariz bin Yusrizal	21001071	Bachelor of Computer Science
Imran Haqeem bin Mohd Khassan	21001366	Bachelor of Computer Science
Ikhwan Hanafi bin Irwan	21001245	Bachelor of Computer Science
Aiman Shuhaikal bin Shuhaimi	21001293	Bachelor of Computer Science
Muhammad Faris bin Mohd Rusli	21001346	Bachelor of Computer Science

Project UML



Members of each Class

Class	Assignee
Length	Imran Haqeem
Mass	Muhammad Faris
Time	Aiman Shuhaikal
Temperature	Muhammad Izhar
Currency	Iman Kamil
Angle	Ammar Afif
Pressure	Ikhwan Hanafi
Area	Danish Hariz
Volume	Ahmad Zafri
Bits	Aqil Fauzan

Source Codes

Temperature Class

TempConverter.java | Temperature.java

src > TempConverter.java > TempConverter > main(String[])

```java
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.*;

public class TempConverter{
    Run | Debug
    public static void main(String[] args) throws Exception {
        Scanner input = new Scanner(System.in);
        NumberFormat formatter = new DecimalFormat(pattern: "#0.0");

        System.out.print(s: "Enter temperature value: ");
        double tempValue = input.nextDouble();

        System.out.print(s: "Enter temperature unit (C/F/K): ");
        String fromUnit = input.next();


        System.out.print(s: "Enter unit to convert to (C/F/K): ");
        String toUnit = input.next();
        input.close();


        // Creates an instance of the class Temperature
        Temperature initialTemp;
        if (fromUnit.equalsIgnoreCase(anotherString: "C")){
            initialTemp = new Celcius(tempValue); // Instantiates a Celcius object if user entered "C"
        }else if (fromUnit.equalsIgnoreCase(anotherString: "F")){
            initialTemp = new Fahrenheit(tempValue); // Instantiates a Fahrenheit object if user entered "F"
        }else if (fromUnit.equalsIgnoreCase(anotherString: "K")){
            initialTemp = new Kelvin(tempValue); // Instantiates a Kelvin object if user entered "K"
        }else{
            System.out.print(s: "Invalid temperature unit!");
            return;
        }
```

TempConverter.java | Temperature.java

src > TempConverter.java > TempConverter > main(String[])

```java
        // Declare a new variable to store converted unit
        double convertedTemp;
        if (toUnit.equalsIgnoreCase(anotherString: "C")){
            convertedTemp = initialTemp.toCelcius();
            System.out.println("Temperature in Celcius: " + formatter.format(convertedTemp));
        }else if (toUnit.equalsIgnoreCase(anotherString: "F")){
            convertedTemp = initialTemp.toFahrenheit();
            System.out.println("Temperature in Fahrenheit: " + formatter.format(convertedTemp));
        }else if (toUnit.equalsIgnoreCase(anotherString: "K")){
            convertedTemp = initialTemp.toKelvin();
            System.out.println("Temperature in Kelvin: " + formatter.format(convertedTemp));
        }else{
            System.out.print(s: "Invalid temperature unit!");
            return;
        }

    }
}

// Inherited classes
class Celcius extends Temperature{
    public Celcius(double value){
        super(value);
    }

    public double toCelcius(){
        return value;
    }

    public double toFahrenheit(){
        return (value * 9 / 5) + 32;
    }

    public double toKelvin(){
        return (value + 273.15);
    }

}
```

TempConverter.java | Temperature.java

src > TempConverter.java > TempConverter > main(String[])

```java
}

class Fahrenheit extends Temperature{
    public Fahrenheit(double value){
        super(value);
    }

    public double toCelcius(){
        return (value - 32) * 5 / 9;
    }

    public double toFahrenheit(){
        return value;
    }

    public double toKelvin(){
        return toCelcius() + 273.15;
    }
}

class Kelvin extends Temperature{
    public Kelvin(double value){
        super(value);
    }

    public double toCelcius(){
        return (value - 273.15);
    }

    public double toFahrenheit(){
        return (value - 273.15) * 9 / 5 + 32;
    }

    public double toKelvin(){
        return value;
    }
}
```

TempConverter.java | Temperature.java

src > Temperature.java > Temperature > toCelcius()

```java
abstract class Temperature {
    protected double value;

    public Temperature(double value){
        this.value = value;
    }

    public abstract double toCelcius();

    public abstract double toFahrenheit();

    public abstract double toKelvin();

}

```

Angle Class

Main.java

```java
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        System.out.println("\n---Choose angle unit to convert from---");
        System.out.println("\tDegree (D)");
        System.out.println("\tRadian (R)");
        System.out.print("Choice (D/R): ");
        String unit = in.next().toUpperCase();

        System.out.print("\nEnter value of angle: ");
        double meas = in.nextDouble();

        if (unit.equalsIgnoreCase( anotherString: "D")){
            Degree deg = new Degree();
            System.out.println("\n" + meas + " " + deg.getUnit() + " in radian: " + String.format("%.4f", deg.degreeToRadian(meas)));
        } else if (unit.equalsIgnoreCase( anotherString: "R")) {
            Radian rad = new Radian();
            System.out.println("\n" + meas + " " + rad.getUnit() + " in degree: " + String.format("%.4f", rad.radianToDegree(meas)));
        } else{
            System.out.println("Invalid unit!");
            return;
        }

        in.close();

    }
}
```

Main.java | Angle.java

```java
2 inheritors
public class Angle {

    protected double measurement;
    protected final double pi = Math.PI;

    public void setMeasurement(double measurement) { this.measurement = measurement; }

}

class Degree extends Angle{
    private String unit = "Degree";

    public String getUnit() { return this.unit; }

    public double degreeToRadian(double measurement){
        double ans;

        ans = measurement * pi / 180;

        return ans;
    }

}

class Radian extends Angle{
    private String unit = "Radian";

    public String getUnit(){
        return this.unit;
    }

    public double radianToDegree(double measurement){
        double ans;

        ans = measurement * 180 / pi;

        return ans;
    }
}
```

Length Class

Main.java | Length.java

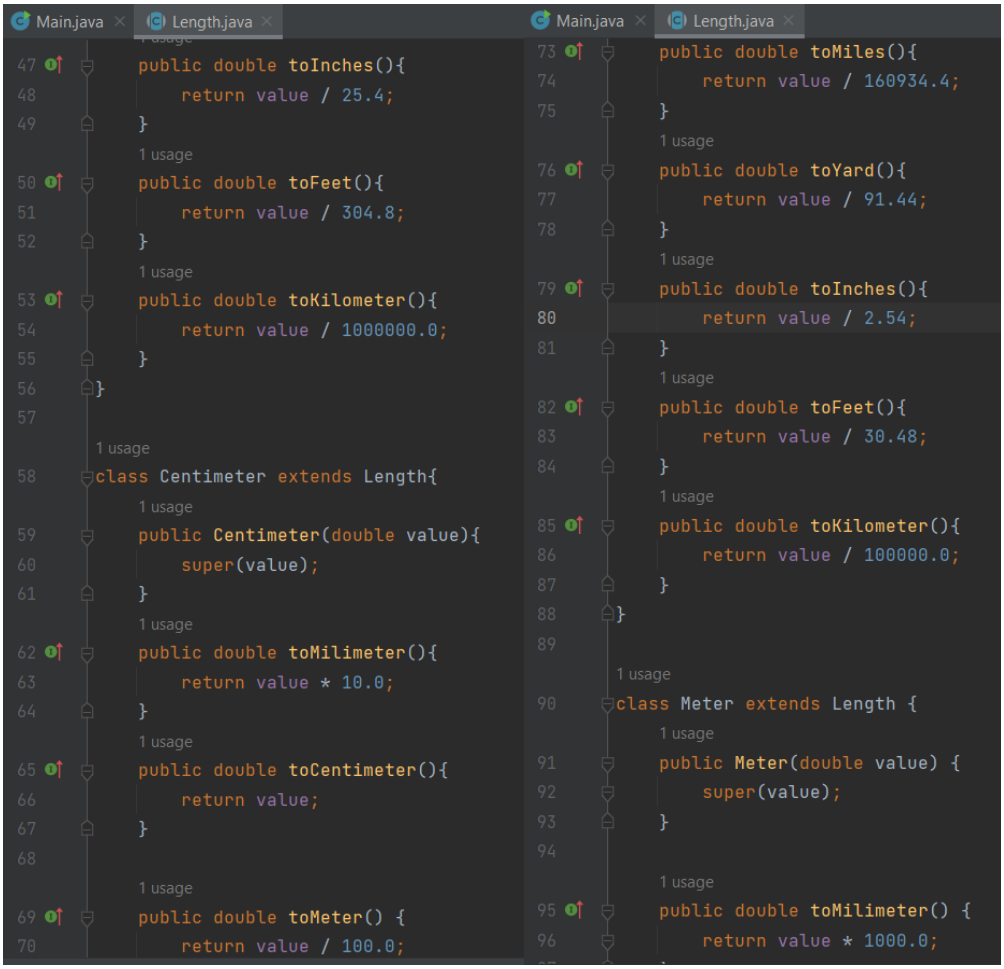
```java
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.*;

no usages
public class Main {
    no usages
    public static void main(String[] args)throws Exception{
        Scanner input = new Scanner(System.in);
        NumberFormat formatter = new DecimalFormat( pattern: "#0.0");

        System.out.print("Enter length value: ");
        double lengthValue = input.nextDouble();

        System.out.print("Enter length unit (MM/CM/M/MI/YD/IN/FT/KM): ");
        String unit = input.next();
        input.close();

        Length length;
        if (unit.equalsIgnoreCase( anotherString: "MM")){
            length = new Milimeter(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "CM")){
            length = new Centimeter(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "M")){
            length = new Meter(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "MI")){
            length = new Miles(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "YD")){
            length = new Yard(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "IN")){
            length = new Inches(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "FT")){
            length = new Feet(lengthValue);
        }else if (unit.equalsIgnoreCase( anotherString: "KM")){
            length = new Kilometer(lengthValue);
        }else{
            System.out.print("Invalid length unit");
            return;
        }

        System.out.println("Length in Milimeter: " + formatter.format(length.toMilimeter()));
        System.out.println("Length in Centimeter: " + formatter.format(length.toCentimeter()));
        System.out.println("Length in Meter: " + formatter.format(length.toMeter()));
        System.out.println("Length in Miles: " + formatter.format(length.toMiles()));
        System.out.println("Length in Yard: " + formatter.format(length.toYard()));
        System.out.println("Length in Inches: " + formatter.format(length.toInches()));
        System.out.println("Length in Feet: " + formatter.format(length.toFeet()));
        System.out.println("Length in Kilometer: " + formatter.format(length.toKilometer()));


    }
}
```

Main.java | Length.java

```java
9 usages  8 inheritors
abstract class Length {
    65 usages
    protected double value;

    8 usages
    public Length(double value) {
        this.value = value;
    }

    1 usage  8 implementations
    public abstract double toMilimeter();

    1 usage  8 implementations
    public abstract double toCentimeter();

    1 usage  8 implementations
    public abstract double toMeter();

    1 usage  8 implementations
    public abstract double toMiles();

    1 usage  8 implementations
    public abstract double toYard();

    1 usage  8 implementations
    public abstract double toInches();

    1 usage  8 implementations
    public abstract double toFeet();
```

Main.java | Length.java

```java
    1 usage  8 implementations
    public abstract double toKilometer();

}
1 usage
class Milimeter extends Length{
    1 usage
    public Milimeter(double value){
        super(value);
    }
    1 usage
    public double toMilimeter(){
        return value;
    }
    1 usage
    public double toCentimeter(){
        return value / 10.0;
    }

    1 usage
    public double toMeter() {
        return value / 1000.0;
    }

    1 usage
    public double toMiles(){
        return value / 1609344.0;
    }
    1 usage
    public double toYard(){
        return value / 914.4;
```



Main.java | Length.java

```
    public double toInches(){
        return value / 25.4;
    }
    1 usage
    public double toFeet(){
        return value / 304.8;
    }
    1 usage
    public double toKilometer(){
        return value / 1000000.0;
    }
}

1 usage
class Centimeter extends Length{
    1 usage
    public Centimeter(double value){
        super(value);
    }
    1 usage
    public double toMilimeter(){
        return value * 10.0;
    }
    1 usage
    public double toCentimeter(){
        return value;
    }

    1 usage
    public double toMeter() {
        return value / 100.0;
```

Main.java | Length.java

```
    public double toMiles(){
        return value / 160934.4;
    }
    1 usage
    public double toYard(){
        return value / 91.44;
    }
    1 usage
    public double toInches(){
        return value / 2.54;
    }
    1 usage
    public double toFeet(){
        return value / 30.48;
    }
    1 usage
    public double toKilometer(){
        return value / 100000.0;
    }
}

1 usage
class Meter extends Length {
    1 usage
    public Meter(double value) {
        super(value);
    }

    1 usage
    public double toMilimeter() {
        return value * 1000.0;
```


Main.java
Length.java
1 usage
public double toCentimeter() {
return value * 100.0;
}
1 usage
public double toMeter() {
return value;
}
1 usage
public double toMiles() {
return value / 1609.0;
}
1 usage
public double toYard() {
return value * 1.094;
}
1 usage
public double toInches() {
return value * 39.37;
}
1 usage
public double toFeet() {
return value * 3.281;
}
Main.java
Length.java
1 usage
public double toKilometer() {
return value / 1000.0;
}
}
1 usage
class Miles extends Length {
1 usage
public Miles(double value) {
super(value);
}
1 usage
public double toMilimeter() {
return value * 1609344;
}
1 usage
public double toCentimeter() {
return value * 160934.4;
}
1 usage
public double toMeter() {
return value * 1609.344;
}
1 usage
public double toMiles() {
return value;
}


Main.java
Length.java
1 usage
public double toYard() {
return value * 1760;
}
1 usage
public double toInches() {
return value * 63360;
}
1 usage
public double toFeet() {
return value * 5280;
}
1 usage
public double toKilometer() {
return value * 1.609344;
}
}
1 usage
class Yard extends Length {
1 usage
public Yard(double value) {
super(value);
}
1 usage
public double toMilimeter() {
return value * 914.4;
}
Main.java
Length.java
public double toCentimeter() {
return value * 91.44;
}
1 usage
public double toMeter() {
return value * 0.9144;
}
1 usage
public double toMiles() {
return value / 1760;
}
1 usage
public double toYard() {
return value;
}
1 usage
public double toInches() {
return value * 36;
}
1 usage
public double toFeet() {
return value * 3;
}
1 usage
public double toKilometer() {

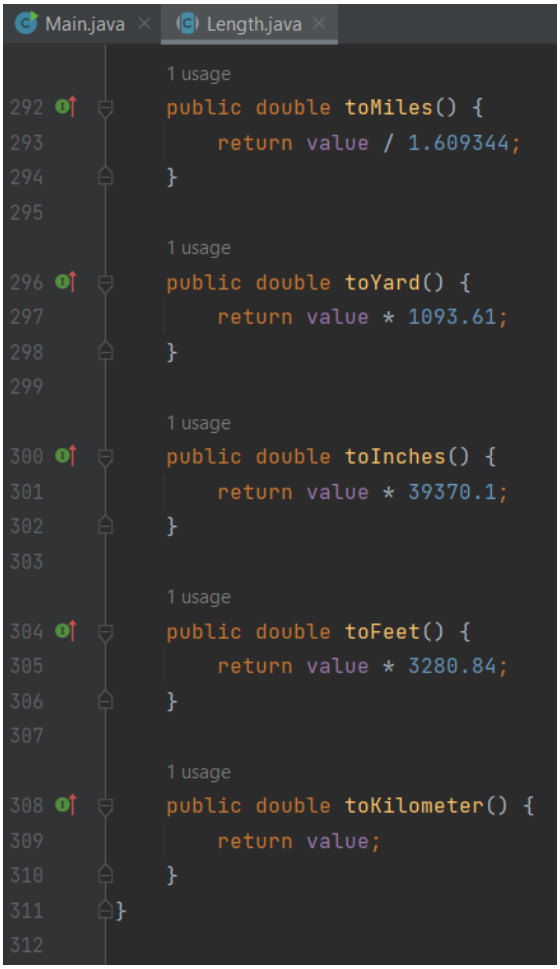

Main.java
Length.java
public double toKilometer() {
return value * 0.0009144;
}
}
1 usage
class Inches extends Length {
1 usage
public Inches(double value) {
super(value);
}
1 usage
public double toMilimeter() {
return value * 25.4;
}
1 usage
public double toCentimeter() {
return value * 2.54;
}
1 usage
public double toMeter() {
return value * 0.0254;
}
1 usage
public double toMiles() {
return value / 63360;
}
Main.java
Length.java
public double toYard() {
return value / 36;
}
1 usage
public double toInches() {
return value;
}
1 usage
public double toFeet() {
return value / 12;
}
1 usage
public double toKilometer() {
return value * 0.0000254;
}
}
1 usage
class Feet extends Length {
1 usage
public Feet(double value) {
super(value);
}
1 usage
public double toMilimeter() {
return value * 304.8;
}

Main.java Length.java

```java
    public double toCentimeter() {
        return value * 30.48;
    }

    1 usage
    public double toMeter() {
        return value * 0.3048;
    }

    1 usage
    public double toMiles() {
        return value / 5280;
    }

    1 usage
    public double toYard() {
        return value / 3;
    }

    1 usage
    public double toInches() {
        return value * 12;
    }

    1 usage
    public double toFeet() {
        return value;
    }

    1 usage
    public double toKilometer() {
```

Main.java Length.java

```java
    public double toKilometer() {
        return value / 3280.84;
    }
}

1 usage
class Kilometer extends Length {
    1 usage
    public Kilometer(double value) {
        super(value);
    }

    1 usage
    public double toMilimeter() {
        return value * 1000000;
    }

    1 usage
    public double toCentimeter() {
        return value * 100000;
    }

    1 usage
    public double toMeter() {
        return value * 1000;
    }

    1 usage
    public double toMiles() {
        return value / 1.609344;
    }
```


Main.java
Length.java
1 usage
public double toMiles() {
return value / 1.609344;
}
1 usage
public double toYard() {
return value * 1093.61;
}
1 usage
public double toInches() {
return value * 39370.1;
}
1 usage
public double toFeet() {
return value * 3280.84;
}
1 usage
public double toKilometer() {
return value;
}
}

Time Class

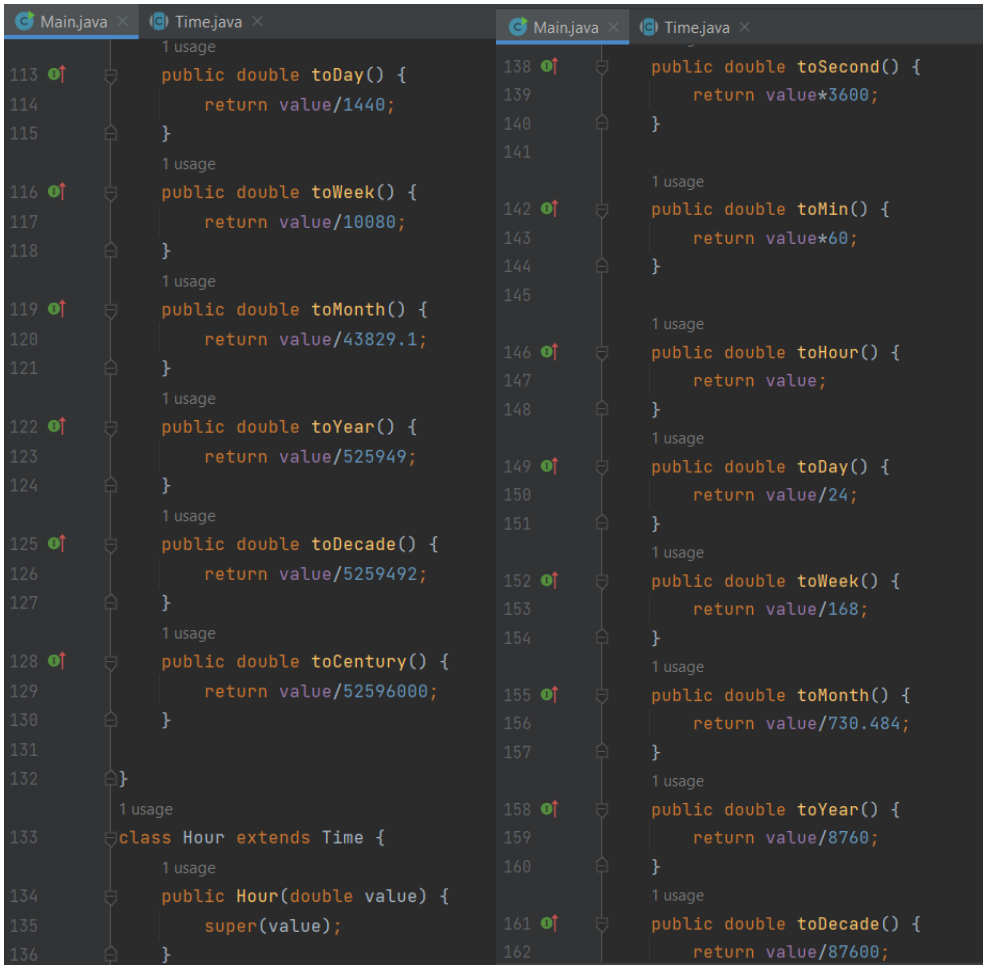
Main.java | Time.java

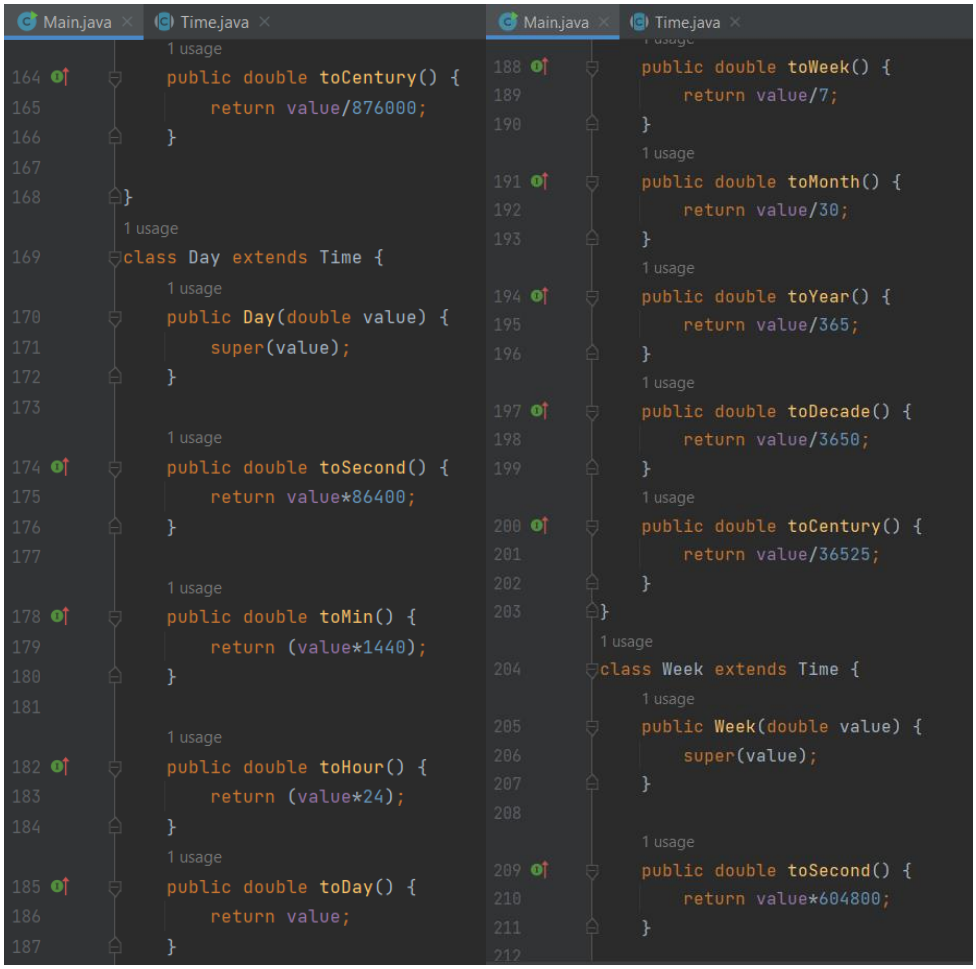
```java
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.*;
no usages
public class Main {

    no usages
    public static void main(String[] args) throws Exception {
        Scanner input = new Scanner(System.in);
        NumberFormat formatter = new DecimalFormat( pattern: "#0.0");

        System.out.print("Enter time value: ");
        double timeValue = input.nextDouble();

        System.out.print("Enter time unit (S/Min/H/D/W/M/Y/Dec/C): ");
        String unit = input.next();
        input.close();


        // Creates an instance of the class
        Time duration;
        if (unit.equalsIgnoreCase( anotherString: "S")) {
            duration = new Second(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "Min")){
            duration = new Minute(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "H")){
            duration = new Hour(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "D")){
            duration = new Day(timeValue);
```

Main.java | Time.java

```java
        }
        else if (unit.equalsIgnoreCase( anotherString: "W")){
            duration = new Week(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "M")){
            duration = new Month(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "Y")){
            duration = new Year(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "Dec")){
            duration = new Decade(timeValue);
        }
        else if (unit.equalsIgnoreCase( anotherString: "C")){
            duration = new Century(timeValue);
        }
        else{
            System.out.print("Invalid time unit!");
            return;
        }
        System.out.println("Time in Second: " + formatter.format(duration.toSecond()));
        System.out.println("Time in Minute: " + formatter.format(duration.toMin()));
        System.out.println("Time in Hour: " + formatter.format(duration.toHour()));
        System.out.println("Time in Day: " + formatter.format(duration.toDay()));
        System.out.println("Time in Week: " + formatter.format(duration.toWeek()));
        System.out.println("Time in Month: " + formatter.format(duration.toMonth()));
        System.out.println("Time in Year: " + formatter.format(duration.toYear()));
        System.out.println("Time in Decade: " + formatter.format(duration.toDecade()));
        System.out.println("Time in Century: " + formatter.format(duration.toCentury()));
    }
}
```

```java
class Second extends Time {
    1 usage
    public Second(double value) {
        super(value);
    }

    1 usage
    public double toSecond() {
        return value;
    }

    1 usage
    public double toMin() {
        return (value/60);
    }

    1 usage
    public double toHour() {
        return (value/3600);
    }
    1 usage
    public double toDay() {
        return (value/86400);
    }
    1 usage
    public double toWeek() {
        return value/604800;
    }
    1 usage
    public double toMonth() {
        return value/2628000;
    }
    1 usage
    public double toYear() {
        return value/31536000;
    }
    1 usage
    public double toDecade() {
        return value/315360000;
    }
    1 usage
    public double toCentury() {
        return value/315360000;
    }
}
1 usage
class Minute extends Time {
    1 usage
    public Minute(double value) {
        super(value);
    }

    1 usage
    public double toSecond() {
        return value*60;
    }

    1 usage
    public double toMin() {
        return value;
    }

    1 usage
    public double toHour() {
        return (value/60);
```


Main.java
Time.java
1 usage
public double toDay() {
return value/1440;
}
1 usage
public double toWeek() {
return value/10080;
}
1 usage
public double toMonth() {
return value/43829.1;
}
1 usage
public double toYear() {
return value/525949;
}
1 usage
public double toDecade() {
return value/5259492;
}
1 usage
public double toCentury() {
return value/52596000;
}
}
1 usage
class Hour extends Time {
1 usage
public Hour(double value) {
super(value);
}
Main.java
Time.java
public double toSecond() {
return value*3600;
}
1 usage
public double toMin() {
return value*60;
}
1 usage
public double toHour() {
return value;
}
1 usage
public double toDay() {
return value/24;
}
1 usage
public double toWeek() {
return value/168;
}
1 usage
public double toMonth() {
return value/730.484;
}
1 usage
public double toYear() {
return value/8760;
}
1 usage
public double toDecade() {
return value/87600;



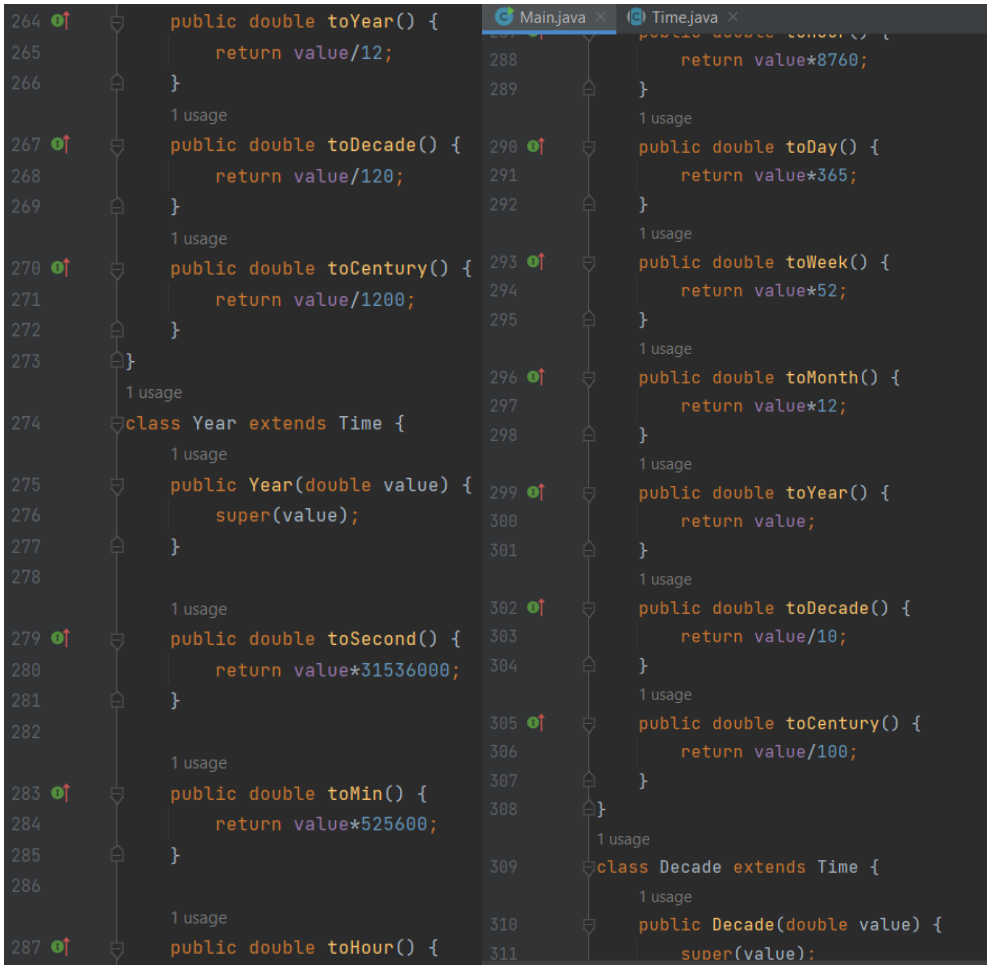
Main.java | Time.java

```
1 usage
    public double toCentury() {
        return value/876000;
    }

}
1 usage
class Day extends Time {
    1 usage
    public Day(double value) {
        super(value);
    }

    1 usage
    public double toSecond() {
        return value*86400;
    }

    1 usage
    public double toMin() {
        return (value*1440);
    }

    1 usage
    public double toHour() {
        return (value*24);
    }
    1 usage
    public double toDay() {
        return value;
    }
```

Main.java | Time.java

```
    public double toWeek() {
        return value/7;
    }
    1 usage
    public double toMonth() {
        return value/30;
    }
    1 usage
    public double toYear() {
        return value/365;
    }
    1 usage
    public double toDecade() {
        return value/3650;
    }
    1 usage
    public double toCentury() {
        return value/36525;
    }
}
1 usage
class Week extends Time {
    1 usage
    public Week(double value) {
        super(value);
    }

    1 usage
    public double toSecond() {
        return value*604800;
    }
```

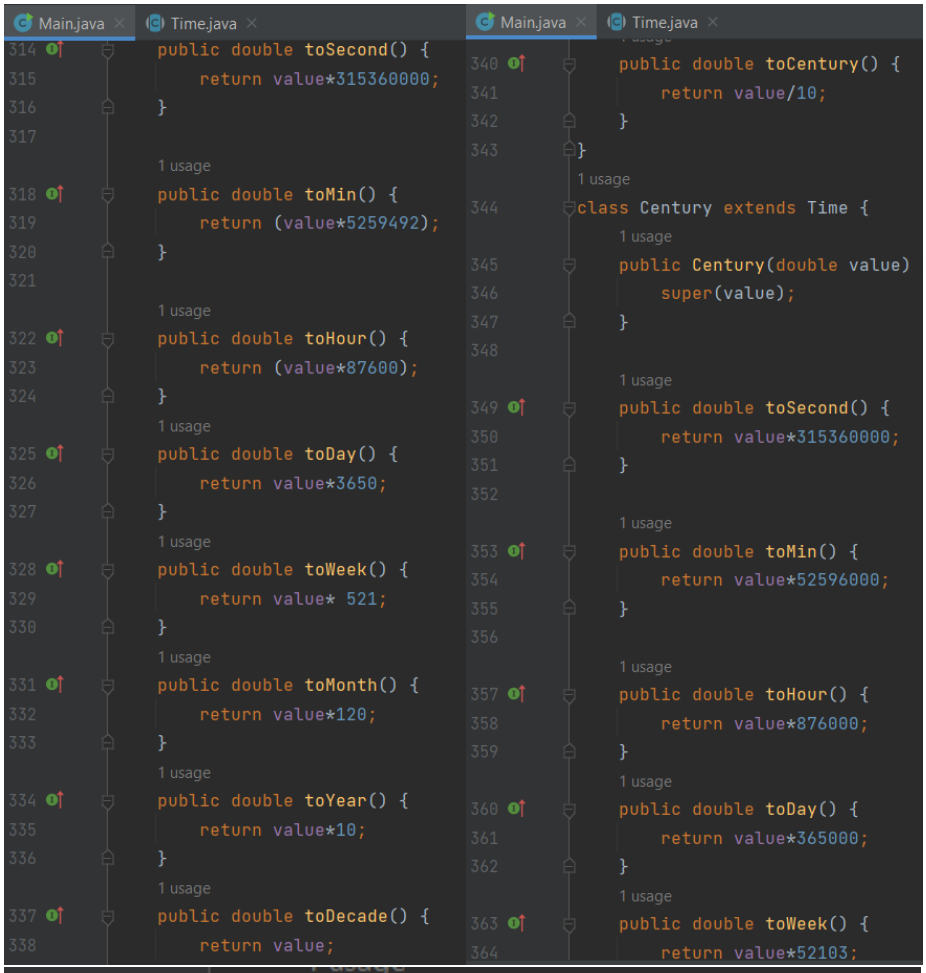

Main.java
Time.java
public double toMin() {
return value*10080;
}
1 usage
public double toHour() {
return value*168;
}
1 usage
public double toDay() {
return value*7;
}
1 usage
public double toWeek() {
return value;
}
1 usage
public double toMonth() {
return value/4;
}
1 usage
public double toYear() {
return value/52;
}
1 usage
public double toDecade() {
return value/520;
}
1 usage
public double toCentury() {
return value/5204;
}
class Month extends Time {
1 usage
public Month(double value) {
super(value);
}
1 usage
public double toSecond() {
return value*2592000;
}
1 usage
public double toMin() {
return value*43200;
}
1 usage
public double toHour() {
return value*720;
}
1 usage
public double toDay() {
return value*30;
}
1 usage
public double toWeek() {
return value*4;
}
1 usage
public double toMonth() {
return value;
}

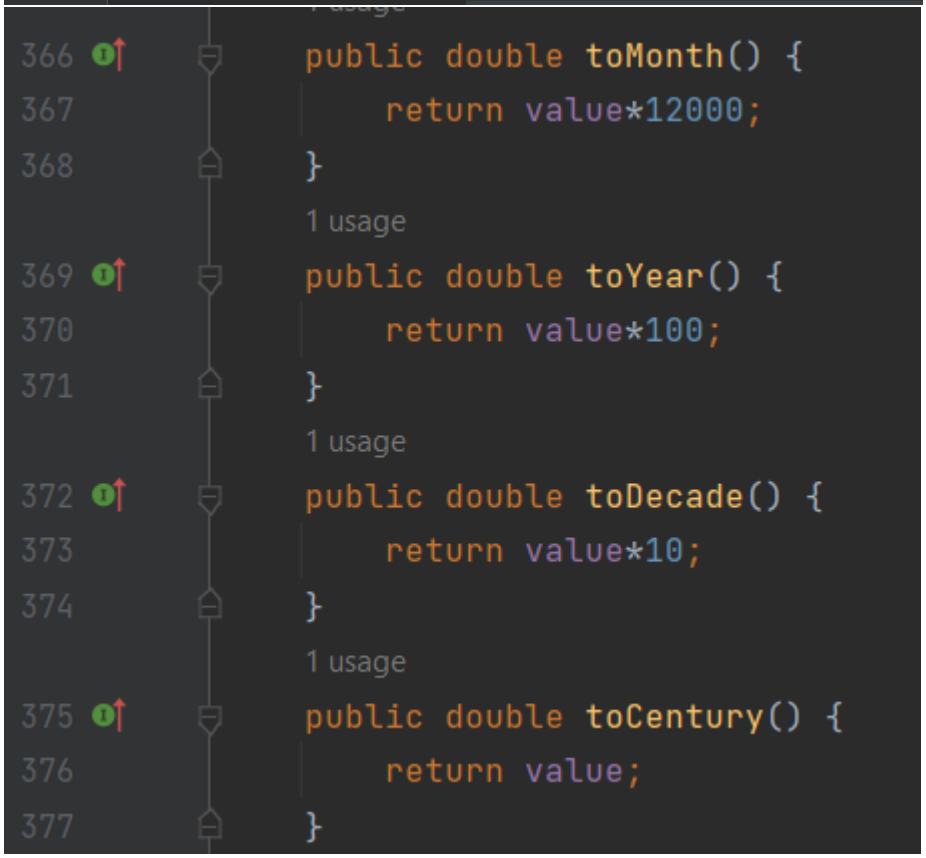


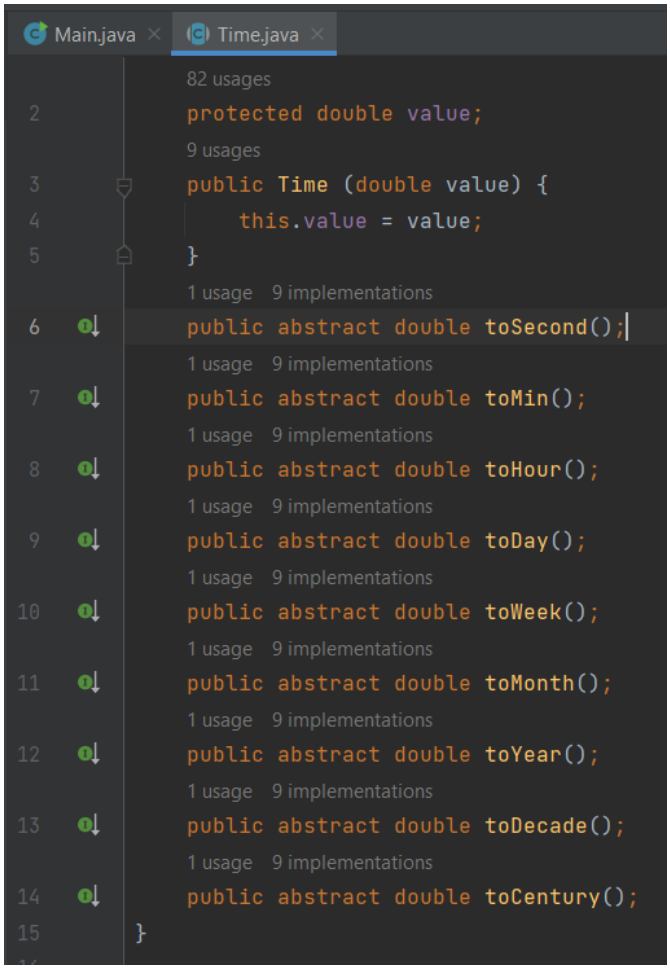
```
264     public double toYear() {
265         return value/12;
266     }
        1 usage
267     public double toDecade() {
268         return value/120;
269     }
        1 usage
270     public double toCentury() {
271         return value/1200;
272     }
273 }
    1 usage
274 class Year extends Time {
        1 usage
275     public Year(double value) {
276         super(value);
277     }
278
        1 usage
279     public double toSecond() {
280         return value*31536000;
281     }
282
        1 usage
283     public double toMin() {
284         return value*525600;
285     }
286
        1 usage
287     public double toHour() {
```

Main.java | Time.java

```
288         return value*8760;
289     }
        1 usage
290     public double toDay() {
291         return value*365;
292     }
        1 usage
293     public double toWeek() {
294         return value*52;
295     }
        1 usage
296     public double toMonth() {
297         return value*12;
298     }
        1 usage
299     public double toYear() {
300         return value;
301     }
        1 usage
302     public double toDecade() {
303         return value/10;
304     }
        1 usage
305     public double toCentury() {
306         return value/100;
307     }
308 }
    1 usage
309 class Decade extends Time {
        1 usage
310     public Decade(double value) {
311         super(value);
```


Main.java
Time.java
public double toSecond() {
return value*315360000;
}
1 usage
public double toMin() {
return (value*5259492);
}
1 usage
public double toHour() {
return (value*87600);
}
1 usage
public double toDay() {
return value*3650;
}
1 usage
public double toWeek() {
return value* 521;
}
1 usage
public double toMonth() {
return value*120;
}
1 usage
public double toYear() {
return value*10;
}
1 usage
public double toDecade() {
return value;
Main.java
Time.java
public double toCentury() {
return value/10;
}
}
1 usage
class Century extends Time {
1 usage
public Century(double value)
super(value);
}
1 usage
public double toSecond() {
return value*315360000;
}
1 usage
public double toMin() {
return value*52596000;
}
1 usage
public double toHour() {
return value*876000;
}
1 usage
public double toDay() {
return value*365000;
}
1 usage
public double toWeek() {
return value*52103;


public double toMonth() {
return value*12000;
}
1 usage
public double toYear() {
return value*100;
}
1 usage
public double toDecade() {
return value*10;
}
1 usage
public double toCentury() {
return value;
}


Main.java
Time.java
82 usages
protected double value;
9 usages
public Time (double value) {
this.value = value;
}
1 usage 9 implementations
public abstract double toSecond();
1 usage 9 implementations
public abstract double toMin();
1 usage 9 implementations
public abstract double toHour();
1 usage 9 implementations
public abstract double toDay();
1 usage 9 implementations
public abstract double toWeek();
1 usage 9 implementations
public abstract double toMonth();
1 usage 9 implementations
public abstract double toYear();
1 usage 9 implementations
public abstract double toDecade();
1 usage 9 implementations
public abstract double toCentury();
}

Currency Class

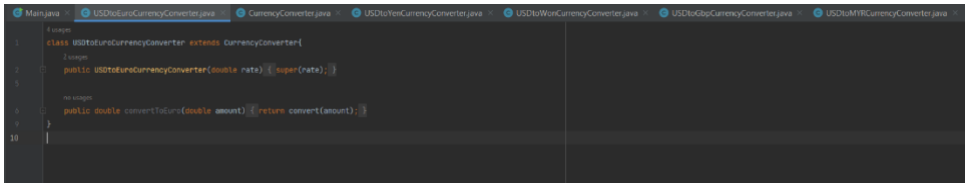
```java
// Main class
import java.util.Scanner;  // Import the Scanner class
no usages
public class Main {
    no usages
    public static void main(String[] args) {
        Scanner Input = new Scanner(System.in);         // Create instances of currency converters

        USDtoMYRCurrencyConverter usdtomyrConverter = new USDtoMYRCurrencyConverter( rate: 4.47629);
        USDtoGbpCurrencyConverter usdtogdpConverter = new USDtoGbpCurrencyConverter( rate: 0.843810);
        USDtoWonCurrencyConverter usdtowonConverter = new USDtoWonCurrencyConverter( rate: 1315.6556);
        USDtoYenCurrencyConverter usdtoyenConverter = new USDtoYenCurrencyConverter( rate: 136.76646);
        USDtoEuroCurrencyConverter usdtoeuroConverter = new USDtoEuroCurrencyConverter( rate: 0.94560212);
```

Main.java | USDtoEuroCurrencyConverter.java | CurrencyConverter.java | USDtoYenCurrencyConverter.java | USDtoWonCurrencyConverter.java | USDtoGbpCurrencyCo

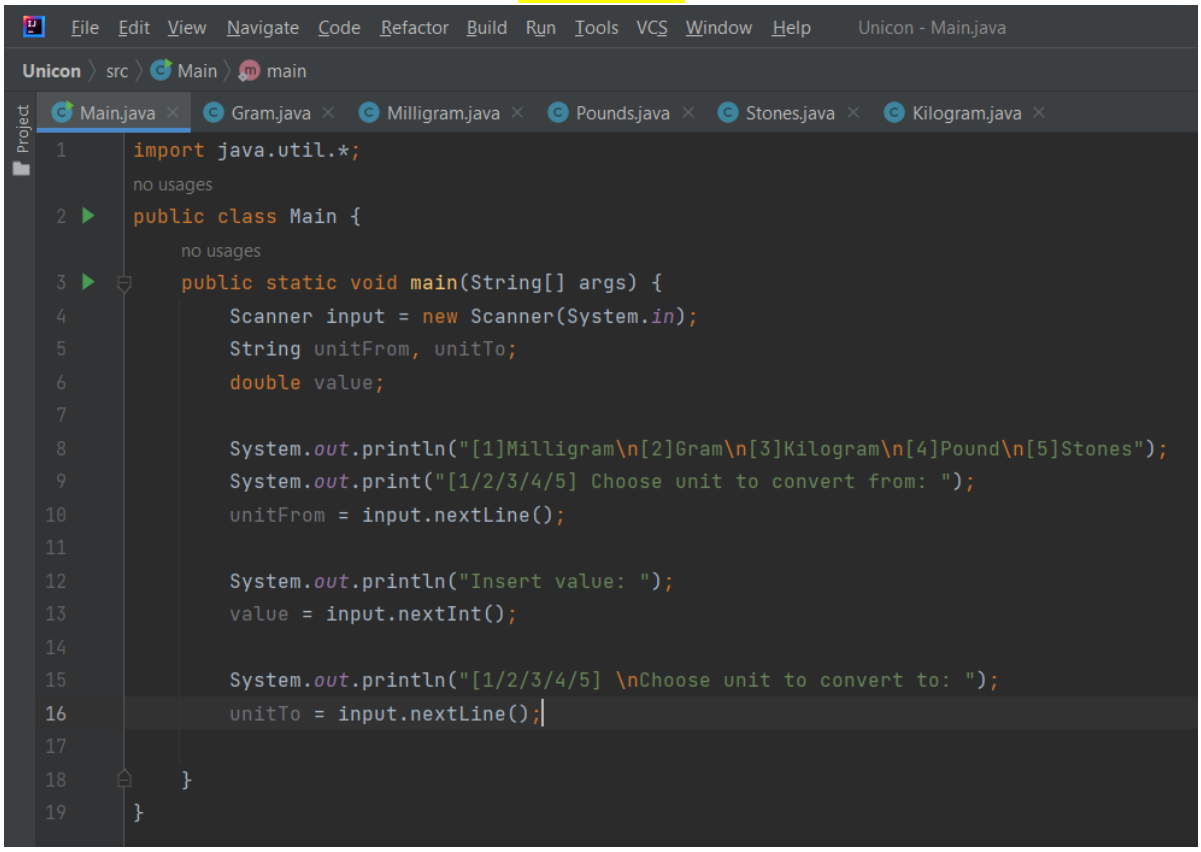
```java
        // Convert from USD to malaysian ringgit

        Integer choice;
        System.out.println("Please enter which converter (1 for USD, 2 for MYR, 3 for Euro,...): ");
        choice = Input.nextInt();

        if(choice == 1)
        {
            double usdAmount;
            System.out.println("Please enter the amount: ");
            usdAmount = Input.nextDouble();

            Integer choice2;
            System.out.println("Please enter which converter: ");
            System.out.println("\n 1 for GDB \n 2 for MYR \n 3 for Korean Won \n 4 for Japanese Yen" +
                    "\n 5 for Euro");
            choice2 = Input.nextInt();

            if(choice2 == 2){
                double MYRAmount = usdtomyrConverter.convert(usdAmount);
                System.out.println("$"  + usdAmount + " is equivalent to RM" + MYRAmount);
            } else if(choice2 == 1){
                double GbpAmount = usdtogdpConverter.convert(usdAmount);
                System.out.println("$"  + usdAmount + " is equivalent to " + GbpAmount + " GBP");
            } else if (choice2 == 3) {
                double WonAmount = usdtowonConverter.convert(usdAmount);
                System.out.println("$"  + usdAmount + " is equivalent to " + WonAmount + " Korean won");
            } else if(choice2 == 4){
                double YenAmount = usdtoyenConverter.convert(usdAmount);
                System.out.println("$"  + usdAmount + " is equivalent to " + YenAmount + " Japanese Yen");
            } else if(choice2 == 5){
                double EuroAmount = usdtoeuroConverter.convert(usdAmount);
                System.out.println("$"  + usdAmount + " is equivalent to " + EuroAmount + " Euros");
            }

        }
    }
```

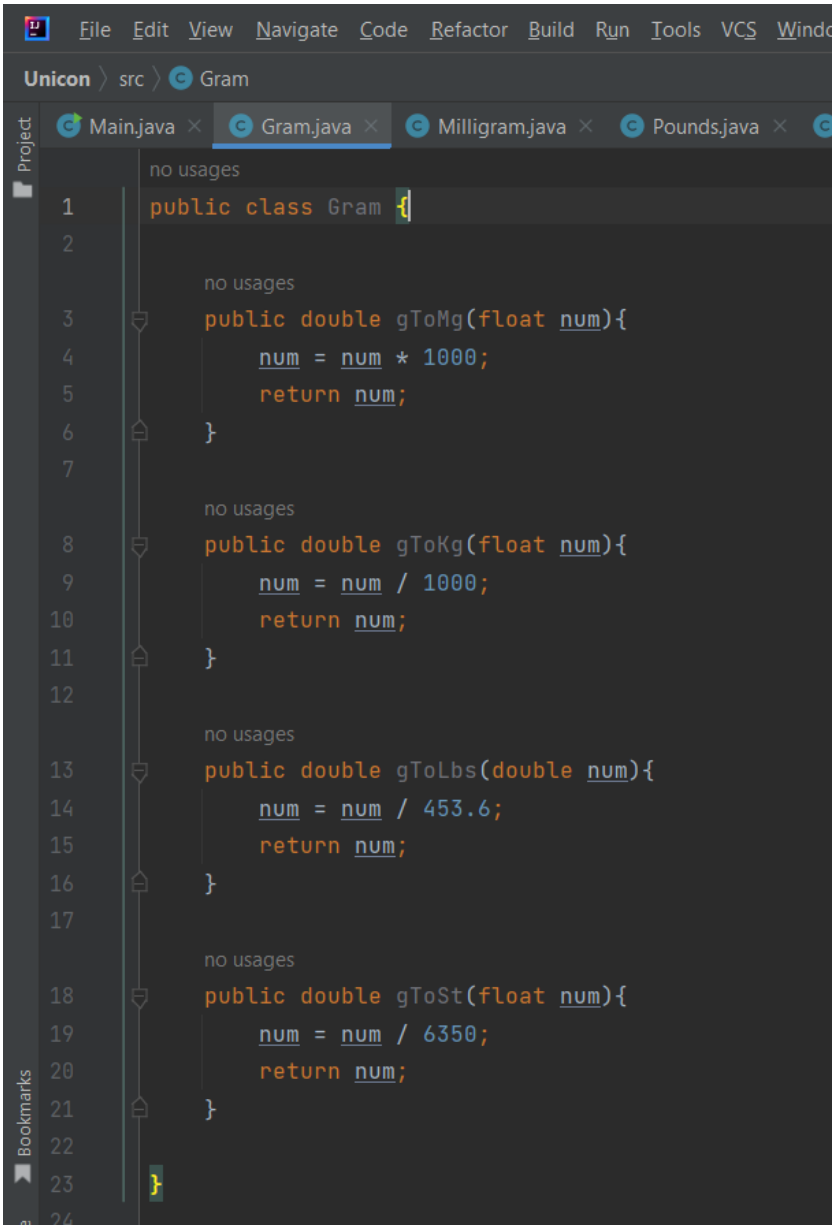
Main.java | USDtoEuroCurrencyConverter.java | CurrencyConverter.java | USDtoYenCurrencyConverter.java | USDtoWonCurrencyConverter.java | USDtoGbp

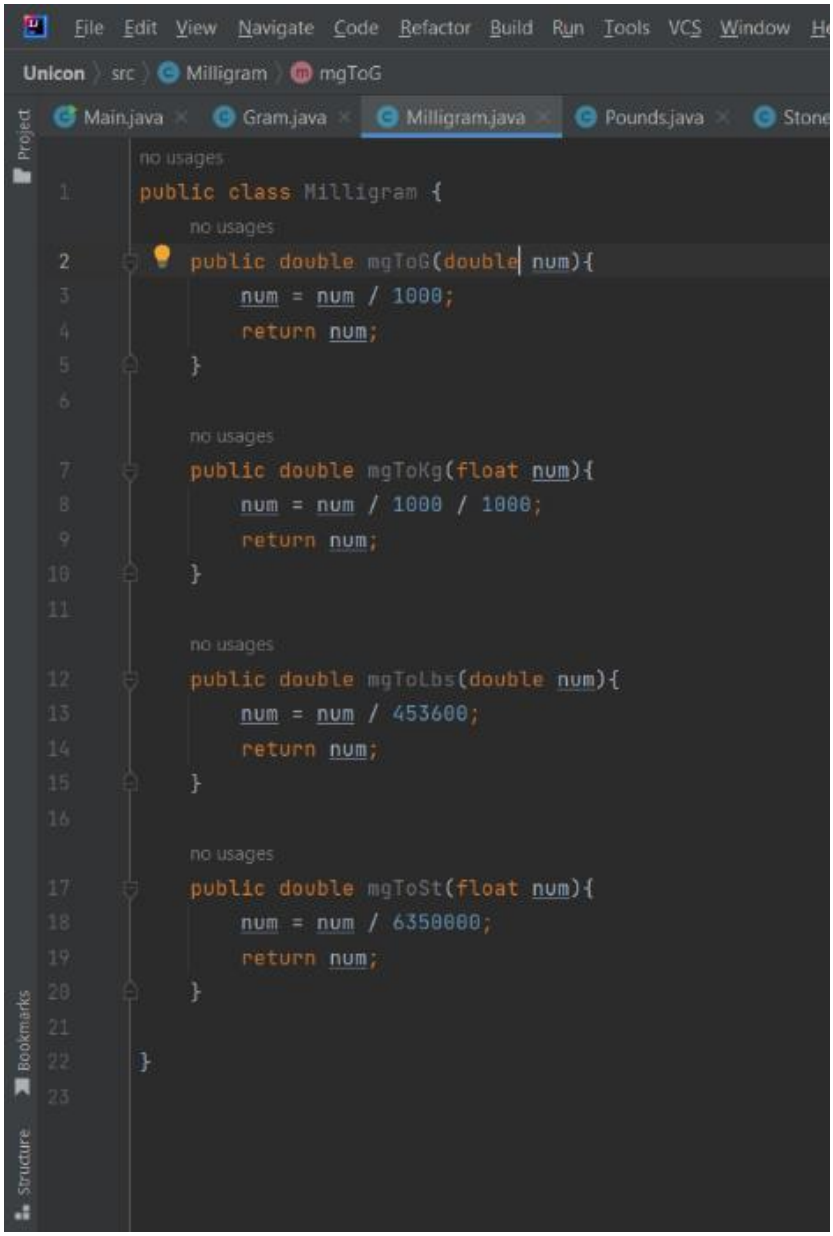
```java
5 usages  5 inheritors
class CurrencyConverter {
    2 usages
    protected double rate;

    5 usages
    public CurrencyConverter(double rate) {
        this.rate = rate;
    }

    15 usages
    public double convert(double amount) {
        return amount * rate;
    }
}
```

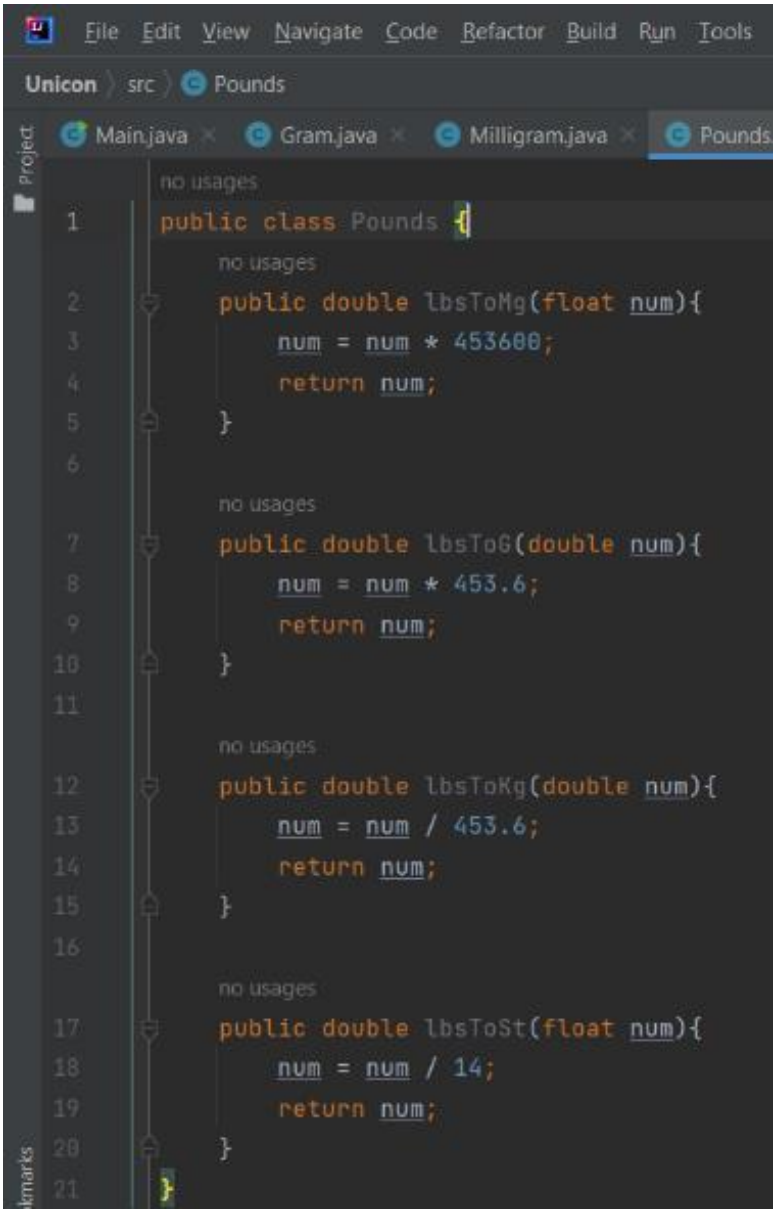

Main.java
USDtoEuroCurrencyConverter.java
CurrencyConverter.java
USDToYenCurrencyConverter.java
USDToWonCurrencyConverter.java
USDToGbpCurrencyConverter.java
USDToMYRCurrencyConverter.java
4 usages
class USDtoEuroCurrencyConverter extends CurrencyConverter{
2 usages
public USDtoEuroCurrencyConverter(double rate) { super(rate); }
no usages
public double convertToEuro(double amount) { return convert(amount); }
}

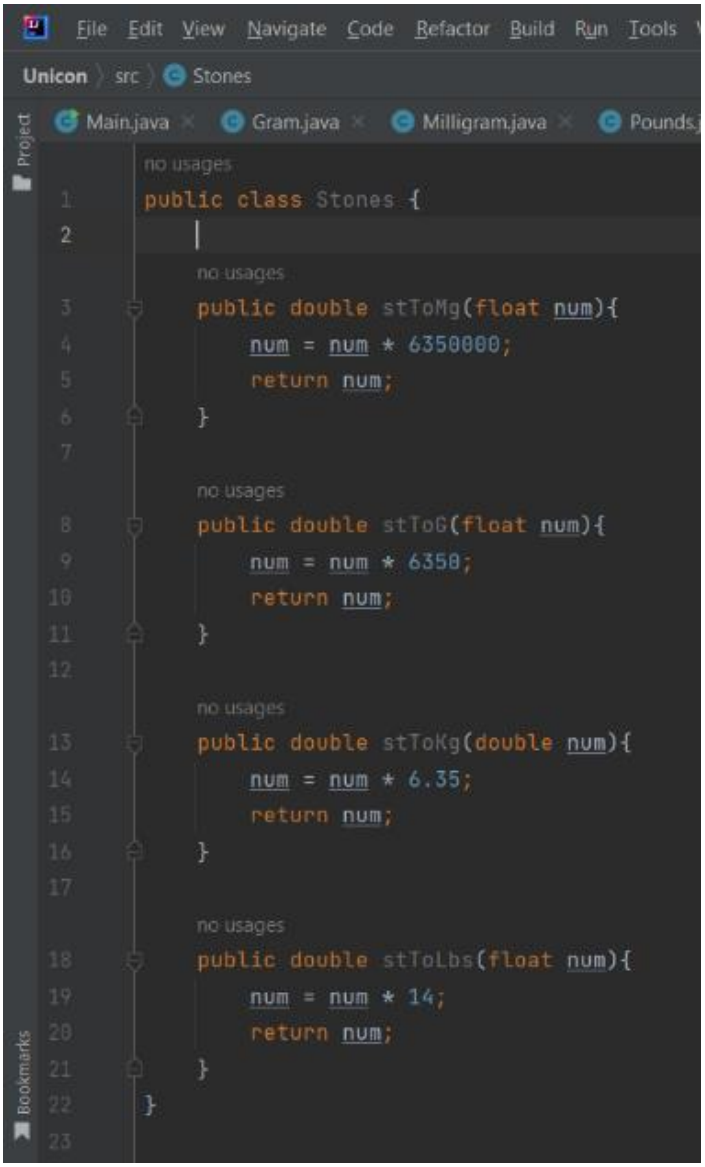
Mass Class

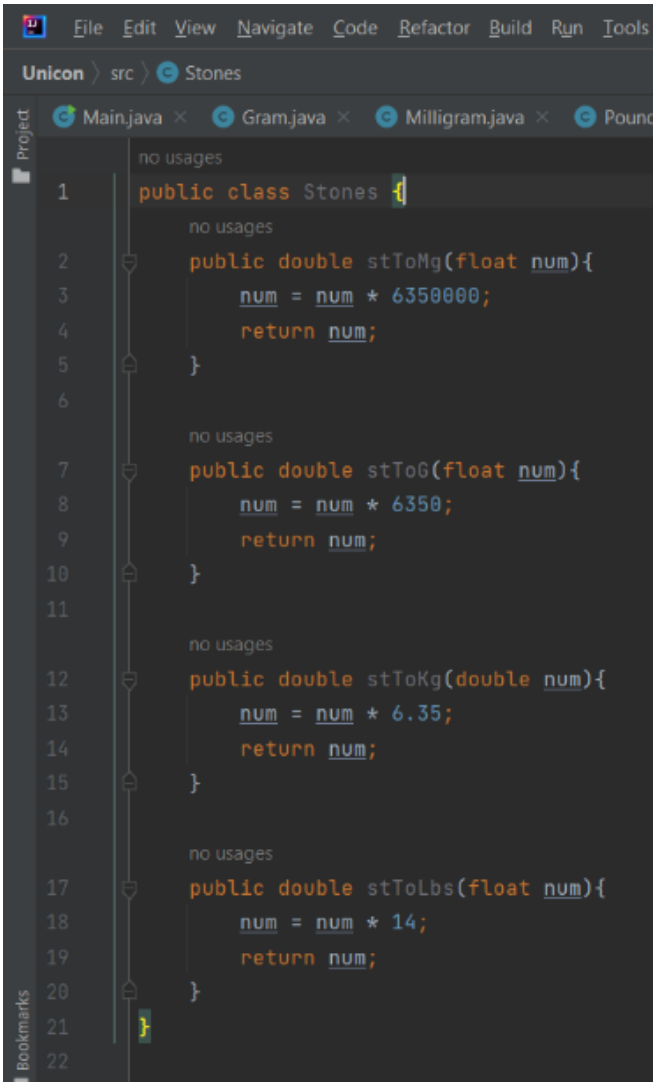

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Unicon - Main.java
Unicon src Main main
Project
Main.java
Gram.java
Milligram.java
Pounds.java
Stones.java
Kilogram.java
1 import java.util.*;
no usages
2 public class Main {
no usages
3 public static void main(String[] args) {
4 Scanner input = new Scanner(System.in);
5 String unitFrom, unitTo;
6 double value;
7
8 System.out.println("[1]Milligram\n[2]Gram\n[3]Kilogram\n[4]Pound\n[5]Stones");
9 System.out.print("[1/2/3/4/5] Choose unit to convert from: ");
10 unitFrom = input.nextLine();
11
12 System.out.println("Insert value: ");
13 value = input.nextInt();
14
15 System.out.println("[1/2/3/4/5] \nChoose unit to convert to: ");
16 unitTo = input.nextLine();
17
18 }
19 }

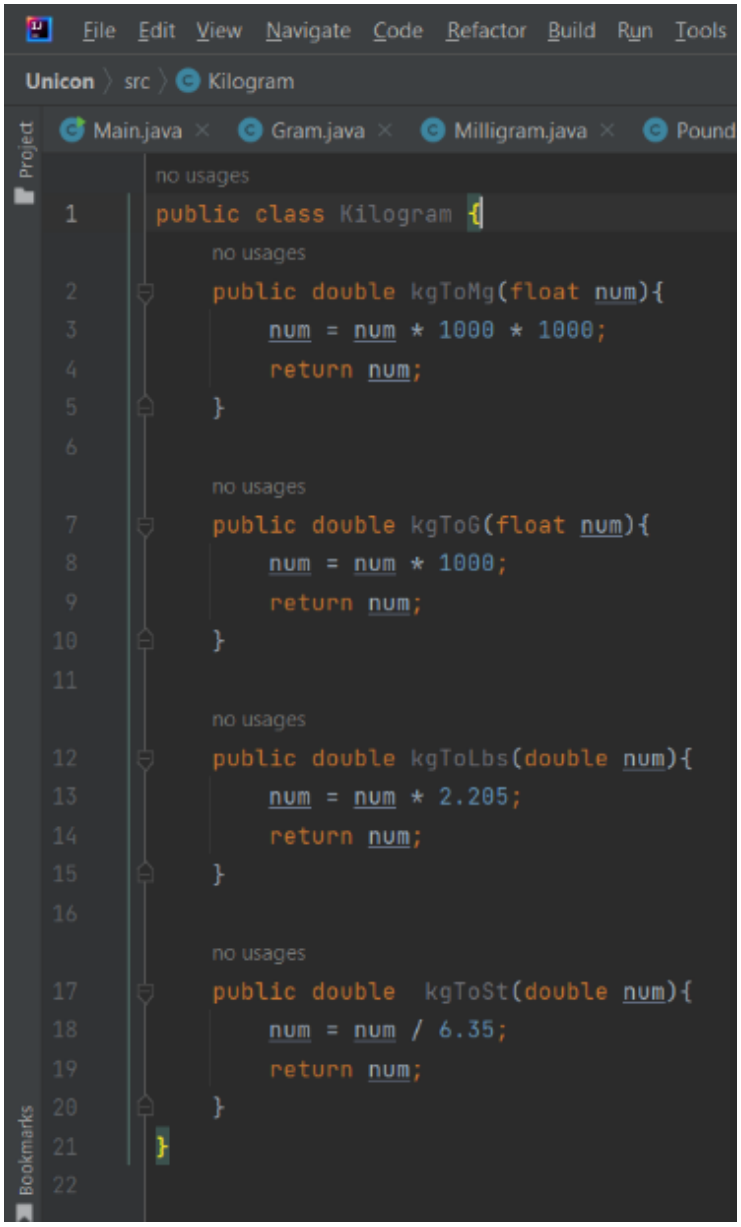

File Edit View Navigate Code Refactor Build Run Tools VCS Windo
Unicon src Gram
Main.java Gram.java Milligram.java Pounds.java
Project
Bookmarks
no usages
public class Gram {
no usages
public double gToMg(float num){
num = num * 1000;
return num;
}
no usages
public double gToKg(float num){
num = num / 1000;
return num;
}
no usages
public double gToLbs(double num){
num = num / 453.6;
return num;
}
no usages
public double gToSt(float num){
num = num / 6350;
return num;
}
}


File Edit View Navigate Code Refactor Build Run Tools VCS Window
Unicon src Milligram mgToG
Main.java Gram.java Milligram.java Pounds.java
Project
Bookmarks
Structure
no usages
public class Milligram {
no usages
public double mgToG(double num){
num = num / 1000;
return num;
}
no usages
public double mgToKg(float num){
num = num / 1000 / 1000;
return num;
}
no usages
public double mgToLbs(double num){
num = num / 453600;
return num;
}
no usages
public double mgToSt(float num){
num = num / 6350000;
return num;
}
}

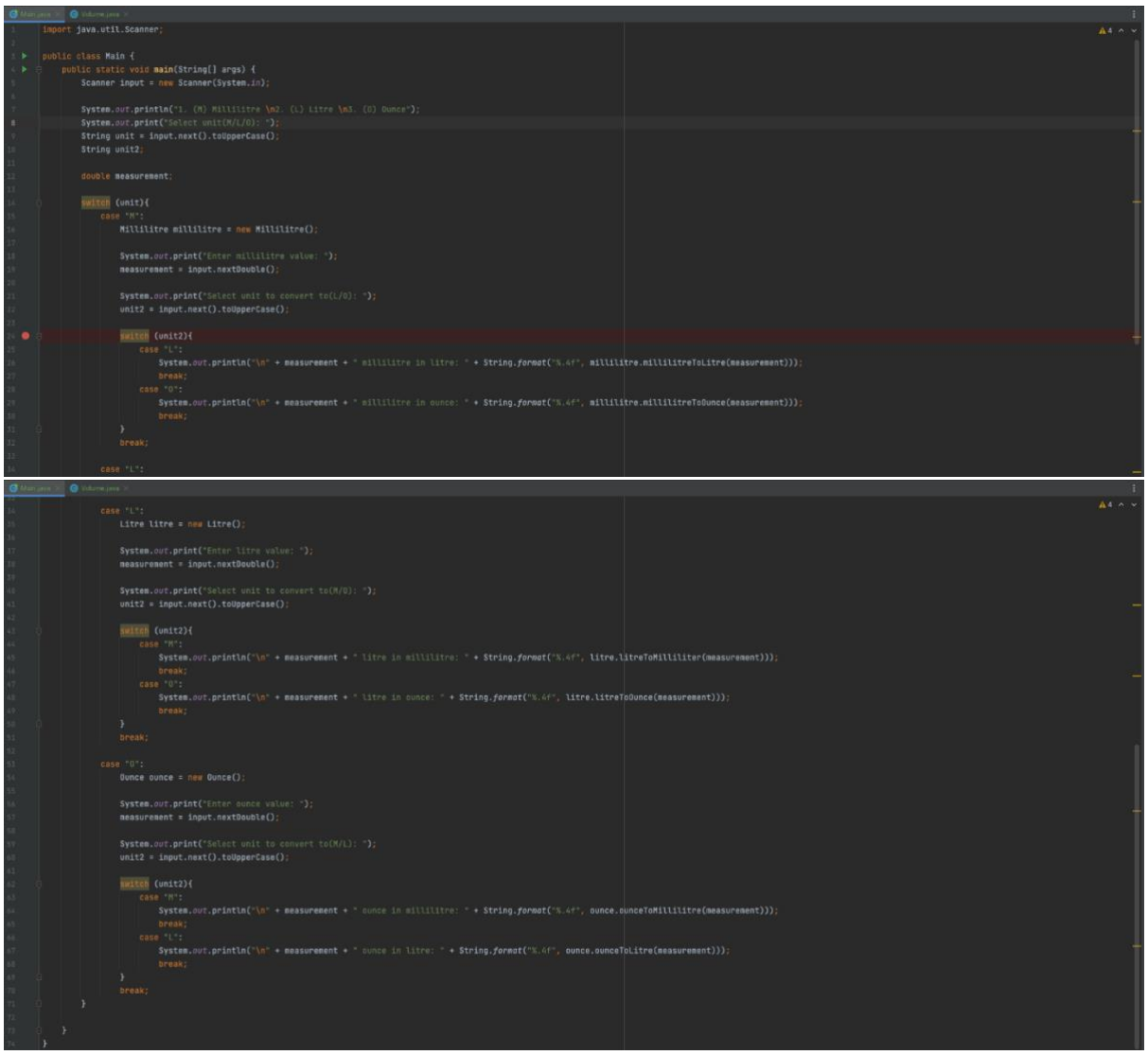

File Edit View Navigate Code Refactor Build Run Tools
Unicon src Pounds
Main.java Gram.java Milligram.java Pounds
project
no usages
public class Pounds {
no usages
public double lbsToMg(float num){
num = num * 453600;
return num;
}
no usages
public double lbsToG(double num){
num = num * 453.6;
return num;
}
no usages
public double lbsToKg(double num){
num = num / 453.6;
return num;
}
no usages
public double lbsToSt(float num){
num = num / 14;
return num;
}
}


File Edit View Navigate Code Refactor Build Run Tools
Unicon src Stones
Main.java Gram.java Milligram.java Pounds.
Project
Bookmarks
no usages
public class Stones {
no usages
public double stToMg(float num){
num = num * 6350000;
return num;
}
no usages
public double stToG(float num){
num = num * 6350;
return num;
}
no usages
public double stToKg(double num){
num = num * 6.35;
return num;
}
no usages
public double stToLbs(float num){
num = num * 14;
return num;
}
}


File Edit View Navigate Code Refactor Build Run Tools
Unicon src Stones
Main.java Gram.java Milligram.java Pound
Project
Bookmarks
no usages
public class Stones {
no usages
public double stToMg(float num){
num = num * 6350000;
return num;
}
no usages
public double stToG(float num){
num = num * 6350;
return num;
}
no usages
public double stToKg(double num){
num = num * 6.35;
return num;
}
no usages
public double stToLbs(float num){
num = num * 14;
return num;
}
}


File Edit View Navigate Code Refactor Build Run Tools
Unicon src Kilogram
Main.java Gram.java Milligram.java Pound
Project
Bookmarks
no usages
public class Kilogram {
no usages
public double kgToMg(float num){
num = num * 1000 * 1000;
return num;
}
no usages
public double kgToG(float num){
num = num * 1000;
return num;
}
no usages
public double kgToLbs(double num){
num = num * 2.205;
return num;
}
no usages
public double kgToSt(double num){
num = num / 6.35;
return num;
}
}

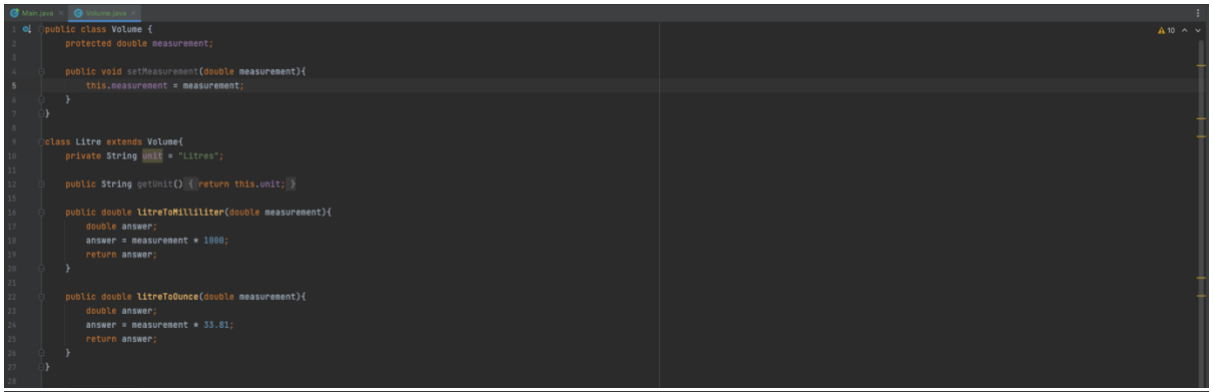
Volume Class



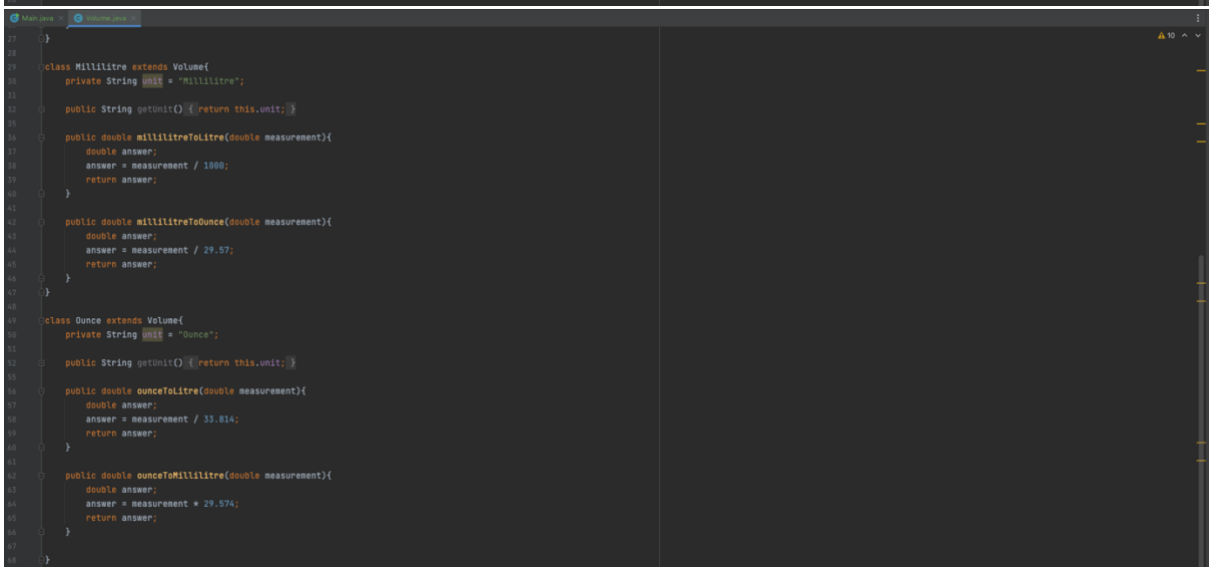
```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.println("1. (M) Millilitre \n2. (L) Litre \n3. (O) Ounce");
        System.out.print("Select unit(M/L/O): ");
        String unit = input.next().toUpperCase();
        String unit2;

        double measurement;

        switch (unit){
            case "M":
                Millilitre millilitre = new Millilitre();

                System.out.print("Enter millilitre value: ");
                measurement = input.nextDouble();

                System.out.print("Select unit to convert to(L/O): ");
                unit2 = input.next().toUpperCase();

                switch (unit2){
                    case "L":
                        System.out.println("\n" + measurement + " millilitre in litre: " + String.format("%.4f", millilitre.millilitreToLitre(measurement)));
                        break;
                    case "O":
                        System.out.println("\n" + measurement + " millilitre in ounce: " + String.format("%.4f", millilitre.millilitreToOunce(measurement)));
                        break;
                }
                break;

            case "L":
```



```java
            case "L":
                Litre litre = new Litre();

                System.out.print("Enter litre value: ");
                measurement = input.nextDouble();

                System.out.print("Select unit to convert to(M/O): ");
                unit2 = input.next().toUpperCase();

                switch (unit2){
                    case "M":
                        System.out.println("\n" + measurement + " litre in millilitre: " + String.format("%.4f", litre.litreToMillilitre(measurement)));
                        break;
                    case "O":
                        System.out.println("\n" + measurement + " litre in ounce: " + String.format("%.4f", litre.litreToOunce(measurement)));
                        break;
                }
                break;

            case "O":
                Ounce ounce = new Ounce();

                System.out.print("Enter ounce value: ");
                measurement = input.nextDouble();

                System.out.print("Select unit to convert to(M/L): ");
                unit2 = input.next().toUpperCase();

                switch (unit2){
                    case "M":
                        System.out.println("\n" + measurement + " ounce in millilitre: " + String.format("%.4f", ounce.ounceToMillilitre(measurement)));
                        break;
                    case "L":
                        System.out.println("\n" + measurement + " ounce in litre: " + String.format("%.4f", ounce.ounceToLitre(measurement)));
                        break;
                }
                break;
        }

    }
}
```



```java
public class Volume {
    protected double measurement;

    public void setMeasurement(double measurement){
        this.measurement = measurement;
    }
}

class Litre extends Volume{
    private String unit = "Litres";

    public String getUnit() { return this.unit; }

    public double litreToMilliliter(double measurement){
        double answer;
        answer = measurement * 1000;
        return answer;
    }

    public double litreToOunce(double measurement){
        double answer;
        answer = measurement * 33.81;
        return answer;
    }
}
```



```java
}

class Millilitre extends Volume{
    private String unit = "Millilitre";

    public String getUnit() { return this.unit; }

    public double millilitreToLitre(double measurement){
        double answer;
        answer = measurement / 1000;
        return answer;
    }

    public double millilitreToOunce(double measurement){
        double answer;
        answer = measurement / 29.57;
        return answer;
    }
}

class Ounce extends Volume{
    private String unit = "Ounce";

    public String getUnit() { return this.unit; }

    public double ounceToLitre(double measurement){
        double answer;
        answer = measurement / 33.814;
        return answer;
    }

    public double ounceToMillilitre(double measurement){
        double answer;
        answer = measurement * 29.574;
        return answer;
    }

}
```

Area Class

Main.java | AreaConverter.java | SquareMeterToSquareFootConverter.java | SquareFootToSquareMeterConverter.java | SquareCentimeterToSquareMilim

```java
import java.util.*;

public class Main {

    public static void main(String[] args) {
        double squareMeters = 10;
        double squareFeet = 100;

        AreaConverter squareMeterToSquareFootConverter = new SquareMeterToSquareFootConverter();
        AreaConverter squareFootToSquareMeterConverter = new SquareFootToSquareMeterConverter();

        double convertedSquareFeet = squareMeterToSquareFootConverter.convert(squareMeters);
        double convertedSquareMeters = squareFootToSquareMeterConverter.convert(squareFeet);

        System.out.println(squareMeters + " square meters is equal to " + convertedSquareFeet + " square feet");
        System.out.println(squareFeet + " square feet is equal to " + convertedSquareMeters + " square meters");
```

Main.java | AreaConverter.java | SquareMeterToSquareFootConverter.java | SquareFootToSquareMeterC

```java
10 usages  8 inheritors
abstract class AreaConverter {
    8 implementations
    public abstract double convert(double area);
}
```

in.java | AreaConverter.java | SquareMeterToSquareFootConverter.java | SquareFootToSquareMeterCo

```java
1 usage
public class SquareMeterToSquareFootConverter extends AreaConverter {
    1 usage
    private static final double CONVERSION_FACTOR = 10.76391;

    @Override
    public double convert(double area) { return area * CONVERSION_FACTOR; }
}
```

SquareFootToSquareMeterConverter.java | SquareCentimeterToSquareMilimeterConverter.java | Squar

```java
1 usage
public class SquareFootToSquareMeterConverter extends AreaConverter {
    1 usage
    private static final double CONVERSION_FACTOR = 0.09290304;

    @Override
    public double convert(double area) { return area * CONVERSION_FACTOR; }
}
```

SquareCentimeterToSquareMilimeterConverter.java | SquareCentimeterToSquareMeter.java | SquareMeterToSquareCent

```java
no usages
public class SquareCentimeterToSquareMilimeterConverter extends AreaConverter {
    1 usage
    private static final double CONVERSION_FACTOR = 100;

    @Override
    public double convert(double area) { return area * CONVERSION_FACTOR; }
}
```

SquareCentimeterToSquareMilimeterConverter.java | SquareCentimeterToSquareMeter.java | Square

```java
no usages
public class SquareCentimeterToSquareMeter extends AreaConverter {
    1 usage
    private static final double CONVERSION_FACTOR = 0.0001;

    @Override
    public double convert(double area) {
        return area * CONVERSION_FACTOR;
    }
```

va | SquareMeterToSquareCentimeterConverter.java | SquareMilimeterToSquareCentimeter.java | SquareKilome

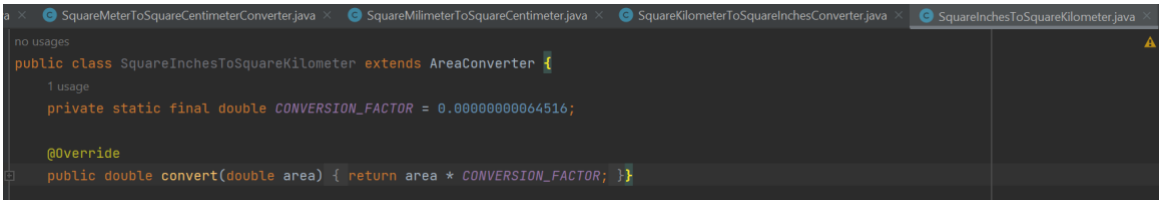
```java
no usages
public class SquareMeterToSquareCentimeterConverter extends AreaConverter {
    1 usage
    private static final double CONVERSION_FACTOR = 10000;

    @Override
    public double convert(double area) {
        return area * CONVERSION_FACTOR;
    }
}
```

SquareMeterToSquareCentimeterConverter.java | SquareMilimeterToSquareCentimeter.java | SquareK

```java
no usages
public class SquareMilimeterToSquareCentimeter extends AreaConverter {

    1 usage
    private static final double CONVERSION_FACTOR = 0.01;

    @Override
    public double convert(double area) { return area * CONVERSION_FACTOR; }
}
```

SquareMeterToSquareCentimeterConverter.java | SquareMilimeterToSquareCentimeter.java | SquareKilometerToSquareInchesConverter.java

```java
no usages
public class SquareKilometerToSquareInchesConverter extends AreaConverter {
    1 usage
    private static final double CONVERSION_FACTOR = 1550003100;

    @Override
    public double convert(double area) { return area * CONVERSION_FACTOR; }}
```



Memory Size Class

Main.java | Bytes.java

```java
public class Bytes {

    static long size;

    static long kilo = 1024;
    static long mega = kilo * kilo;
    static long giga = mega * kilo;
    static long tera = giga * kilo;


    public void bitsToByte(long size){
        this.size = size;

        double byteSize = (double)size / 8;

        System.out.println(byteSize + " byte");

        byteToKb(byteSize);
        byteToMb(byteSize);
        byteToGb(byteSize);
        byteToTb(byteSize);
    }

    public void byteToKb(double size){
        double kb = size / kilo;

        System.out.println(kb + " KB");
    }

    public void byteToMb(double size){
        double mb = size / mega;

        System.out.println(mb + " MB");

    }

    public void byteToGb(double size){
        double gb = size / giga;

        System.out.println(gb + " GB");
    }

    public void byteToTb(double size) {
        double tb = size / tera;

        System.out.println(tb + " TB");
    }
}
```

```java
import java.util.Scanner;

new *
public class Main {
    new *
    public static void main(String[] args) {
        Bytes bt = new Bytes();

        Scanner input = new Scanner(System.in);


        System.out.print("Enter size: ");
        double num = input.nextDouble();


        bt.bitsToByte((long) num);
    }
}
```

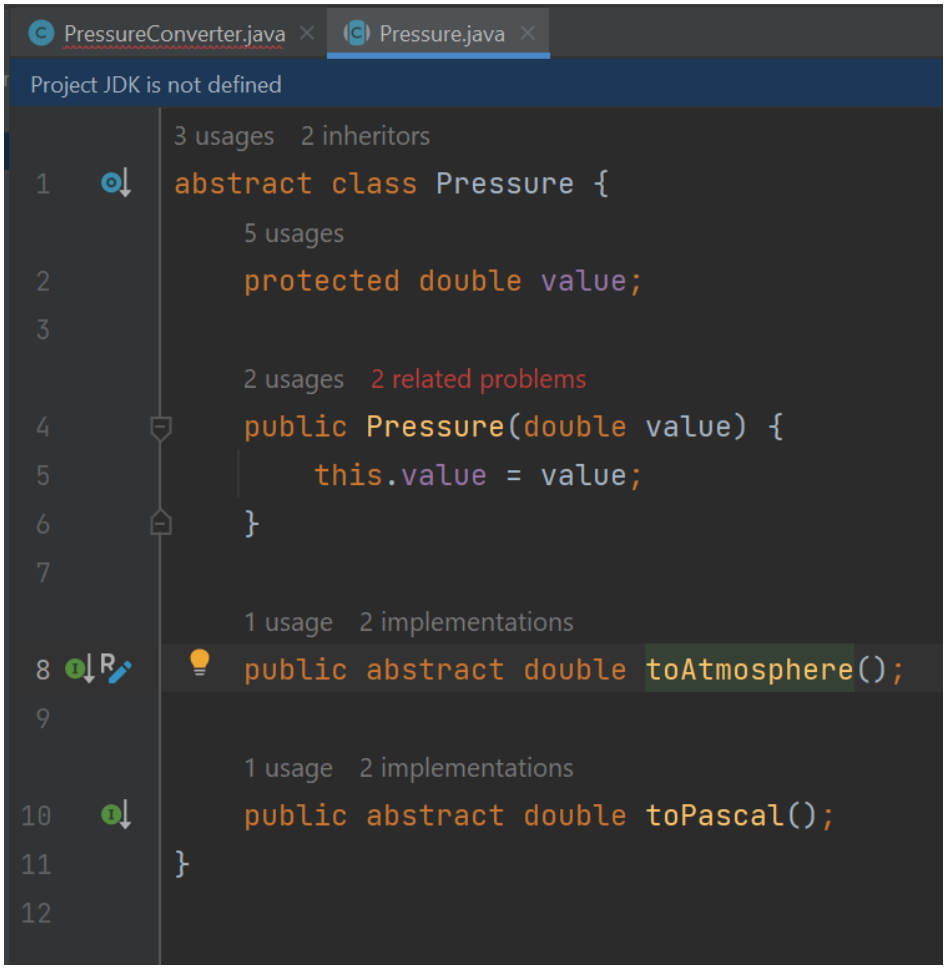
Pressure Class

PressureConverter.java | Pressure.java

Project JDK is not defined

```java
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.*;

no usages
public class PressureConverter {
    no usages
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        NumberFormat formatter = new DecimalFormat(pattern:"#0.00");

        System.out.print(s: "Enter Pressure Value: ");
        double pressureValue = input.nextDouble();

        System.out.print(s: "Enter Pressure Unit (A/P): ");
        String initialUnit = input.next();

        System.out.print(s: "Enter unit to convert to (A/P): ");
        String finalUnit = input.next();
        input.close;

        Pressure initialPressure;
        if (initialUnit.equalsIgnoreCase(anotherString: "A")) {
            initialPressure = new Atmosphere(pressureValue);
        }else if (initialUnit.equalsIgnoreCase(anotherString: "P")) {
            initialPressure = new Pascal(pressureValue);
        }else {
            System.out.print(s: "Invalid Pressure Unit.");
            return;
        }
        double convertedPressure;
        if (finalUnit.equalsIgnoreCase(anotherString: "A")) {
            convertedPressure = initialPressure.toAtmosphere();
            System.out.println("Pressure in Atmosphere(atm): " + formatter.format(convertedPressure));
        }else if (finalUnit.equalsIgnoreCase(anotherString: "P")) {
            convertedPressure = initialPressure.toPascal();
            System.out.println("Pressure in Pascal(Pa): " + formatter.format(convertedPressure));
        }else {
            System.out.print(s: "Invalid Pressure Unit.");
            return;
        }

    }
}
```

```java
class Atmospshere extends Pressure {
    no usages
    public Atmospshere(double value) {
        super(value);
    }

    1 usage
    public double toAtmosphere() {
        return value;
    }

    1 usage
    public double toPascal() {
        return (value * 101325);
    }
}
1 usage
class Pascal extends Pressure {
    1 usage
    public Pascal(double value) {
        super(value);
    }

    1 usage
    public double toAtmosphere() {
        return (value / 101325);
    }

    1 usage
    public double toPascal() {
        return (value);
    }
}
```


PressureConverter.java
Pressure.java
Project JDK is not defined
3 usages 2 inheritors
abstract class Pressure {
5 usages
protected double value;
2 usages 2 related problems
public Pressure(double value) {
this.value = value;
}
1 usage 2 implementations
public abstract double toAtmosphere();
1 usage 2 implementations
public abstract double toPascal();
}