JOBSHEET 6
INHERITANCE

Hammam Abdullah Saeed B.G
N.A : 12
Nim : 2341720203
Class : TI_I2

Github limk :
https://github.com/hemoabdullah/Semester-3/tree/main/week6/src

TRIAL 1 (extends)

ClassB.java

```java
public class ClassA {  1 usage  1 inheritor
    protected int x;  4 usages
    protected int y;  4 usages

    public void setX(int x) { this.x = x; }

    public void setY(int y) { this.y = y; }

    void getNilai() {  1 usage
        System.out.println("Nilai x = " + x);
        System.out.println("Nilai y = " + y);
    }
}
```

Percobaan1.java

```java
public class Percobaan1 {
    public static void main(String[] args) {
        ClassB hitung = new ClassB();
        hitung.x = 20;
        hitung.y = 30;
        hitung.setZ(5);
        hitung.getNilai();
        hitung.getNilaiZ();
        hitung.getJumlah();
    }
}
```

The program provided consists of a Percobaan1 class with a main method, which uses another class named ClassB. The purpose of this program is to instantiate an object from ClassB and perform some operations on it

B. QUESTIONS

1. In Experiment 1 above the program that was running error occurred, then fix so that the program can be run and not error!

```java
public class ClassB extends ClassA {  2 usages
    private int z;  4 usages

    public void getNilaiZ() { System.out.println("Nilai z = " + z); }

    void getJumlah() { System.out.println("Jumlah "+(x+y+z)); }

    public void setZ(int z) { this.z = z; }

    public int getZ() { return z; }
}
```

The error occurs because the child class does not properly inherit or implement the required methods or constructors from the superclass. You need to review the class definitions to ensure they follow the correct inheritance rules.

--

2. Explain what caused the program in experiment 1 when it ran an error!

The error might be related to access control or the absence of proper constructor calls from the superclass using super().

Result:

```
"C:\Program Files\Java\jdk-22\bin\java.exe"
Nilai x = 20
Nilai y = 30
Nilai z = 5
Jumlah 55
```

TRIAL 2 (Access Control)

Run the program above, then observe what happens!

B. QUESTIONS 1. In Experiment 2 above, the program that runs an error occurs, then fix it so 1.that the program can be run and not error!

2. Explain what caused the program in experiment 1 when it ran an error!

--

ClassB.java | Percobaan2.java | ClassA.java

```
1  public class Percobaan2 {
2      public static void main(String[] args) {
3          ClassB hitung = new ClassB();
4          hitung.setX(20);
5          hitung.setY(30);
6          hitung.setZ(5);
7          hitung.getNilai();
8          hitung.getNilaiZ();
9          hitung.getJumlah();
10     }
11 }
```

The error could occur due to incorrect access modifiers , we Ensure the fields and methods that should be inherited have protected or public access modifiers.

It was caused because in the first code the inherited methods was not protected and I fixed the void to run it .

Result:

```
Nilai x = 20
Nilai y = 30
Nilai z = 5
Jumlah 55
```

--

TRIAL 3 (Super)

```
baan2.java   Tabung.java ×   bangun.java   percobaan3.java
public class Tabung extends bangun {  2 usages
    protected int t;  2 usages

    void setSuperPhi(double phi) {  1 usage
        super.phi = phi;
    }

    void setSuperR(int r) {  1 usage
        super.r = r;
    }

    public void setT(int t) {  1 usage
        this.t = t;
    }

    void volume() {  1 usage
        System.out.println("Volume Tabung = " + (super.phi * super.r * super.r * this.t));

    }
}
```

Run the program above!

```
Volume Tabung = 942.0

Process finished with exit code 0
```

B. QUESTIONS 1

1.Explain the "super" function in the following program snippet in the Tube class

=

The super() call in the constructor of Tabung (Tube) refers to the constructor of its superclass Bangun. This ensures that when an object of Tabung is created, the constructor of Bangun runs first to initialize the parent class's fields.

2. Explain the "super" and "this" functions in the following program snippet in the Tube class 3. Explain why the Tube class does not declare the "phi" and "r" attributes, but the class can access these attributes!

=

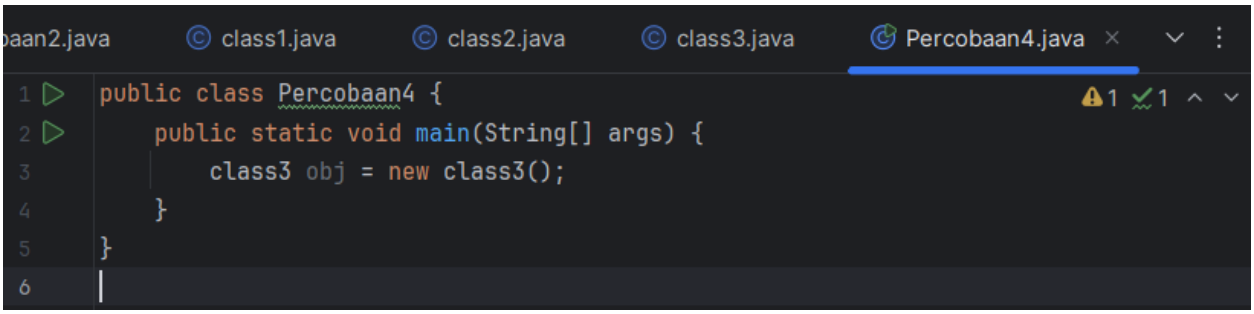
The super keyword is used to refer to members of the superclass, while this refers to the current class. Both are essential when a subclass has attributes or methods with the same names as the superclass.

3.

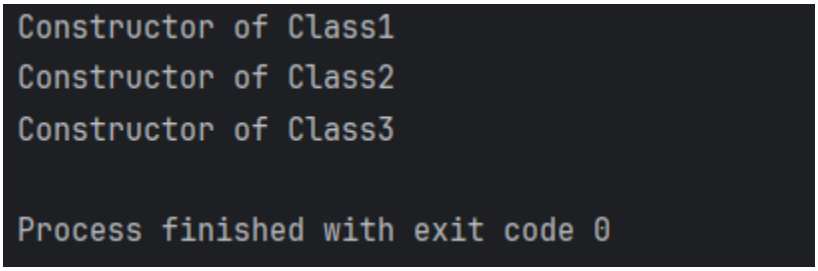
Why Tube can access phi and r: The Tabung class accesses phi and r from its superclass Bangun because these attributes are inherited. If these attributes are protected or public in Bangun, they are accessible in Tabung.

--

Trial 4: super constructor



Run the program then observe what happens!



--

1.Which class includes the superclass and subclass? Explain the reason.

Class1 is the superclass because it does not extend any other class.

Class2 is both a subclass (of Class1) and a superclass (to Class3).

2.What happens when the word super() is added in the constructor of Class3?

Adding super() to the first line of Class3's constructor explicitly calls the constructor of its immediate superclass, Class2. If you remove it, the call to super() is still made implicitly by Java. So adding it explicitly makes no difference in the output for this specific scenario.

Without super(), the constructor would still follow the same sequence of constructor calls.

What happens if super() is placed on the second line of the constructor?
Placing super() on the second line will cause a **compilation error**. The super() call **must** be the first line in the constructor, because the parent class's constructor must complete before the subclass's constructor can run its own initialization code.
You will see an error message like: Call to super() must be first statement in constructor.

Explain the constructor order when the test object is created in ClassC.
When an object of Class3 is created:

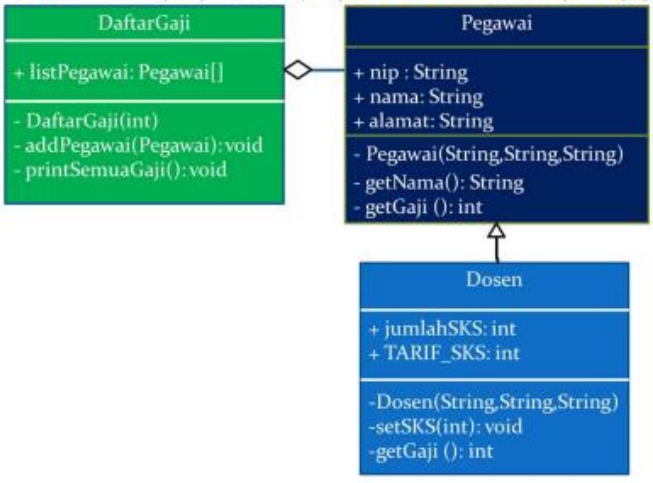
1. **Class1's constructor** runs first because it is the top-most superclass.
2. **Class2's constructor** runs next, after Class1 finishes.
3. **Class3's constructor** runs last, after `Class2` finishes.

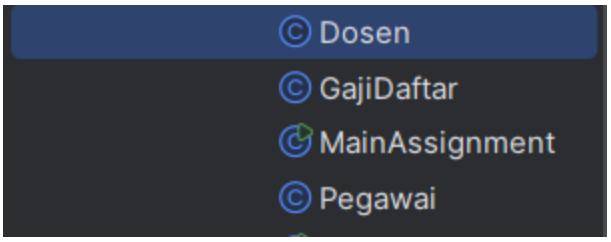
What is the super() function in ClassC's constructor?
it calls the constructor of its immediate superclass, Class2. This ensures that Class2's constructor is executed before Class3's constructor. In turn, Class2's constructor also calls Class1's constructor. This chain ensures that all constructors in the inheritance hierarchy are executed in the correct order.
--

. ASSIGNMENTS

10. ASSIGNMENTS

1. Make a program with the concept of inheritance as in the following class diagram. Then create an object instantiation to display the employee name and salary they get.






```java
public class Dosen extends Pegawai {  4 usages
    private int jumlahSKS;  2 usages
    private static final int TARIF_SKS = 50000;  1 usage

    public Dosen(String nip, String nama, String alamat) {  2 usages
        super(nip, nama, alamat);
    }

    public void setSKS(int SKS) {  2 usages
        this.jumlahSKS = SKS;
    }

    @Override  1 usage
    public int getGaji() {
        return this.jumlahSKS * TARIF_SKS;
    }
}
```

```java
public class Pegawai {  12 usages  1 inheritor
    protected String nip;  3 usages
    protected String nama;  3 usages
    protected String alamat;  2 usages

    public Pegawai(String nip, String nama, String alamat) {  3 usages
        this.nip = nip;
        this.nama = nama;
        this.alamat = alamat;
    }

    public String getNip() {  1 usage
        return nip;
    }

    public String getNama() {  no usages
        return nama;
    }

    public int getGaji() {  1 usage  1 override
        return 0;
    }

    public void displayInfo() {  2 usages
        System.out.println("NIP: " + nip + ", Nama: " + nama + ", Alamat: " + alamat + ", Gaji: Rp " + getGaji());
    }
```

```java
public class GajiDaftar {  2 usages
    private Pegawai[] listPegawai;  5 usages
    private int index;  4 usages

    public GajiDaftar(int jumlahPegawai) {  1 usage
        listPegawai = new Pegawai[jumlahPegawai];
        index = 0;
    }

    public void addPegawai(Pegawai pegawai) {  4 usages
        if (index < listPegawai.length) {
            listPegawai[index] = pegawai;
            index++;
        }
    }

    public Pegawai cariPegawai(String nip) {  1 usage
        for (Pegawai pegawai : listPegawai) {
            if (pegawai != null && pegawai.getNip().equals(nip)) {
                return pegawai;
            }
        }
        return null;
    }

    public void tampilkanSemuaPegawai() {  2 usages
        for (Pegawai pegawai : listPegawai) {
            if (pegawai != null) {
                pegawai.displayInfo();
            }
        }
    }
}
```

```java
import java.util.Scanner;

public class MainAssignment {
    public static void main(String[] args) {
        GajiDaftar daftarGaji = new GajiDaftar( jumlahPegawai: 5);

        Dosen dosen1 = new Dosen( nip: "5432",  nama: "Dr. Afif",  alamat: "Jl. Bandulan");
        dosen1.setSKS(12);
        daftarGaji.addPegawai(dosen1);

        Dosen dosen2 = new Dosen( nip: "2356",  nama: "Dr. Ameen",  alamat: "Jl. Suhat");
        dosen2.setSKS(14);
        daftarGaji.addPegawai(dosen2);

        Pegawai pegawai1 = new Pegawai( nip: "9879",  nama: "hammam",  alamat: "Jl.sukun");
        daftarGaji.addPegawai(pegawai1);

        Pegawai pegawai2 = new Pegawai( nip: "1010",  nama: "majid",  alamat: "Jl.batu");
        daftarGaji.addPegawai(pegawai2);

        Scanner scanner = new Scanner(System.in);
        boolean continueRunning = true;

        while (continueRunning) {
            System.out.println("\nMenu:");
            System.out.println("1. show All pegawai");
            System.out.println("2. search  by NIP");
            System.out.println("3. show the daftar pegawai");
            System.out.println("4. Exit");
            System.out.print("pick option: ");
            int choice = scanner.nextInt();
            scanner.nextLine();
```

Result :

```
Menu:
1. show All pegawai
2. search  by NIP
3. show the daftar pegawai
4. Exit
pick option:
```

```
pick option: 1
NIP: 5432, Nama: Dr. Afif, Alamat: Jl. Ba
NIP: 2356, Nama: Dr. Ameen, Alamat: Jl. S
NIP: 9879, Nama: hammam, Alamat: Jl.sukun
NIP: 1010, Nama: majid, Alamat: Jl.batu,
```

```
Pick option: 2
Enter the NIP you are looking for: 9879
NIP: 9879, Nama: Hammam, Alamat: Jl. Sukun,
```

```
Pick option: 3
Daftar Gaji Pegawai:
NIP: 5432, Nama: Dr. Afif, Alamat: Jl. Bandulan, Gaji: Rp 600000
NIP: 2356, Nama: Dr. Ameen, Alamat: Jl. Suhat, Gaji: Rp 700000
NIP: 9879, Nama: Hammam, Alamat: Jl. Sukun, Gaji: Rp 4800000
NIP: 1010, Nama: Majid, Alamat: Jl. Batu, Gaji: Rp 3600000
```

In summaty The program manages employee data using inheritance. Pegawai is the base class that holds common attributes like NIP, name, and work hours, while Dosen extends it with specific features such as SKS for salary calculation. The GajiDaftar class stores employees and provides functions to add, search by NIP, and display salary information. A menu-driven system allows users to view all employees, search by NIP, or exit. Salaries for Dosen are based on SKS, and for Pegawai, they depend on work hours. The program efficiently handles multiple employee types through inheritance and polymorphism.

Github link :
https://github.com/hemoabdullah/Semester-3/tree/main/week6/src