LAPORAN RESMI

Praktikum 7 Mengelola Kelas

Mata Kuliah: Praktek Pemrograman Berbasis Objek

Disusun oleh:

M. Ainur Ramadhan (3122500047)

2 D3 Teknik Informatika B

Dosen Pengampu: Yanuar Risah Prayogi S.Kom., M.Kom.

PROGRAM STUDI D3 TEKNIK INFORMATIKA

POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2023/2024

A. TUGAS PENDAHULUAN

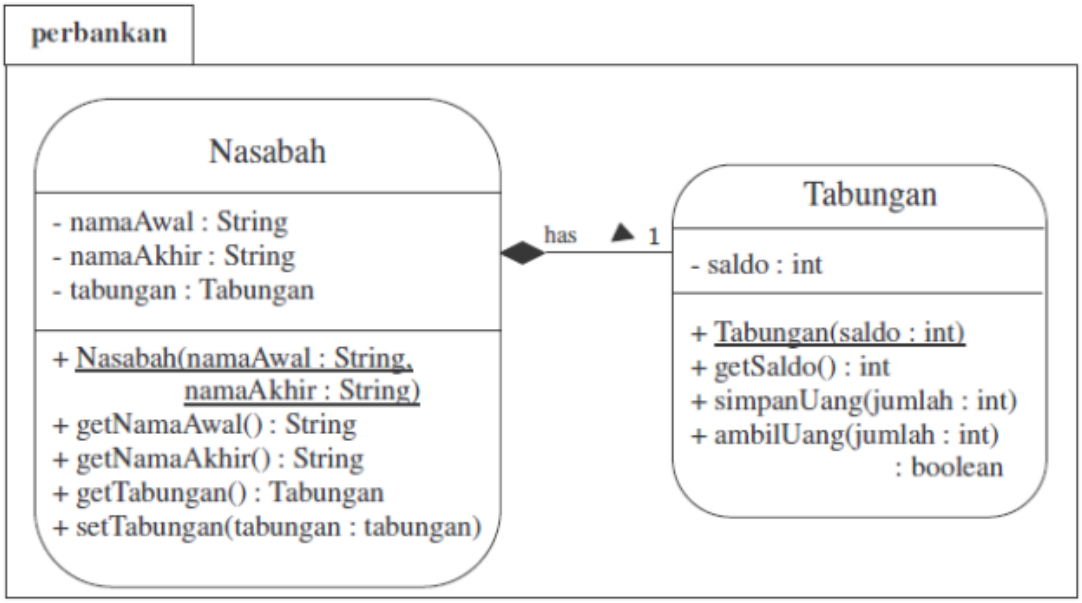
1. Apakah yang dimaksud dengan package?
   Package adalah cara untuk mengorganisasi kelas-kelas dalam bahasa pemrograman seperti Java. Package adalah direktori yang berisi kelas-kelas terkait yang dikelompokkan bersama. Dengan menggunakan package, Anda dapat mengorganisasi dan mengelompokkan kelas-kelas berdasarkan fungsionalitas atau tujuan tertentu.

2. Apakah kegunaan kata kunci import?
   Kata kunci **import** digunakan dalam banyak bahasa pemrograman berorientasi objek (seperti Java) untuk mengimpor kelas atau paket dari package lain agar dapat digunakan dalam program.
   Kegunaan: Import memungkinkan untuk mengakses dan menggunakan kelas atau paket yang terdefinisi di luar kelas saat ini. Ini membantu dalam menghindari penamaan kelas yang ambigu dan membuat kode lebih mudah dibaca. Dengan mengimpor kelas atau paket, maka dapat menggunakan fungsionalitas yang ada di dalamnya.

3. Apakah kegunaan kata kunci this?
   Kata kunci **this** digunakan dalam banyak bahasa pemrograman untuk merujuk ke objek saat ini di dalam konteks kelas. Dalam banyak kasus, ini digunakan untuk membedakan antara variabel kelas dan parameter atau variabel lokal dengan nama yang sama.
   Kegunaan: Kegunaan utama dari **this** adalah untuk mengakses variabel kelas atau metode kelas saat ada variabel dengan nama yang sama dalam lingkup yang lebih lokal (seperti metode atau konstruktor). Ini memungkinkan untuk menghindari konflik nama dan memungkinkan penggunaan yang benar dari variabel kelas. Selain itu, **this** juga digunakan untuk memanggil metode kelas lain dan konstruktor kelas lain saat ada konstruktor yang overloading.

B. LATIHAN

1. Mengimplementasikan UML class diagram dalam program untuk package perbankan



Kode Program ➔ class *Nasabah.java*

```java
package perbankan;

public class Nasabah {
    String namaAwal;
    String namaAkhir;
    Tabungan tabungan;

    public Nasabah(String namaAwal, String namaAkhir) {
        this.namaAwal = namaAwal;
        this.namaAkhir = namaAkhir;
    }

    public String getNamaAwal() {
        return namaAwal;
    }

    public String getNamaAkhir() {
        return namaAkhir;
    }

    public Tabungan getTabungan() {
        return tabungan;
    }

    public void setTabungan(Tabungan tabungan) {
        this.tabungan = tabungan;
    }
}
```

Analisa:
Kode tersebut digunakan untuk memodelkan nasabah dengan informasi nama dan rekening tabungan mereka. Kelas **Nasabah** ini bertindak sebagai entitas yang menyimpan informasi dasar tentang nasabah dan mengelola hubungan antara nasabah dan rekening tabungan mereka dengan menggunakan getter dan setter.

Kode Program ➔ *Tabungan.java*

```
package perbankan;

public class Tabungan {
    int saldo;

    public Tabungan(int saldo) {
        this.saldo = saldo;
    }

    public int getSaldo() {
        return saldo;
    }

    public void simpanUang(int jumlah) {
        saldo += jumlah;
    }

    public boolean ambilUang(int jumlah) {
        if (jumlah <= saldo) {
            saldo -= jumlah;
            return true;
        } else {
            return false;
        }
    }
}
```

Analisa:
Kode ini digunakan untuk mengimplementasikan rekening tabungan dengan kemampuan untuk menyimpan dan mengambil uang, namun dengan akses langsung ke saldo tanpa perlu menggunakan getter. Hal ini dapat memudahkan penggunaan saldo, tetapi juga dapat menimbulkan potensi masalah jika saldo dimodifikasi secara langsung tanpa validasi.

Berikut kode program yang akan dikompilasi:

```
import perbankan.*;

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

```
        int tmp;
        boolean status;
        Nasabah nasabah = new Nasabah("Agus", "Daryanto");
        System.out.println("Nasabah atas nama" + nasabah.getNamaAwal()
+ nasabah.getNamaAkhir());
        nasabah.setTabungan(new Tabungan(5000));
        tmp = nasabah.getTabungan().getSaldo();
        System.out.println("Saldo awal : " + tmp);
        nasabah.getTabungan().simpanUang(3000);
        System.out.println("Jumlah uang yang disimpan 3000");
        status = nasabah.getTabungan().ambilUang(6000);
        System.out.println("Jumlah uang yang diambil 6000");
        if (status)
            System.out.println(" OK");
        else
            System.out.println(" Gagal");
        nasabah.getTabungan().simpanUang(3500);
        System.out.println("Jumlah uang yang disimpan 3500");
        status = nasabah.getTabungan().ambilUang(4000);
        System.out.println("Jumlah uang yang diambil 4000");
        if (status)
            System.out.println(" OK");
        else
            System.out.println(" Gagal");
        status = nasabah.getTabungan().ambilUang(1600);
        System.out.println("Jumlah uang yang diambil 1600");
        if (status)
            System.out.println(" OK");
        else
            System.out.println(" Gagal");
        nasabah.getTabungan().simpanUang(2000);
        System.out.println("Jumlah uang yang disimpan 2000");
        tmp = nasabah.getTabungan().getSaldo();
        System.out.println("Saldo sekarang = " + tmp);
    }
}
```

Analisa:
Dalam metode main, program: Membuat objek Nasabah dengan nama "Agus" dan "Daryanto". Mengatur saldo awal rekening tabungan nasabah sebesar 5000. Melakukan berbagai operasi, seperti simpan uang dan ambil uang dari rekening tabungan. Mencetak hasil operasi-operasi tersebut, termasuk saldo akhir.

Kode ini menyediakan simulasi sederhana tentang bagaimana seorang nasabah dapat berinteraksi dengan rekening tabungannya, yaitu dengan melakukan simpanan dan penarikan uang. Status keberhasilan atau kegagalan dari operasi-operasi tersebut dicetak ke layar.

Output

```
C:\Users\USER\OOP\Praktikum\Praktikum 7\Latihan>javac *.java

C:\Users\USER\OOP\Praktikum\Praktikum 7\Latihan>java TesLatihan
Nasabah atas nama Agus Daryanto
Saldo awal : 5000
Jumlah uang yang disimpan: 3000
Jumlah uang yang diambil: 6000
OK
Jumlah uang yang disimpan 3500
Jumlah uang yang diambil: 4000
OK
Jumlah uang yang diambil: 1600
Gagal
Jumlah uang yang disimpan: 2000
Saldo sekarang = 3500
```

Pada Output tersebut terlihat sudah sama seperti yang diperintahkan di soal, namun untuk outputnya terdapat beberapa symbol seperti titik dua yang saya tambahkan.

C. TUGAS

1. Perhatikan program dibawah ini. Apa yang terjadi bila dikompile dan dijalankan? Jelaskan jawaban anda!

```java
public class Pegawai {
    int nip;
    String nama;

    public Pegawai(int nip_pegawai){
        this(nip_pegawai,"NoName");
    }

    public Pegawai(int nip_pegawai, String nama_pegawai) {
        this.nip = nip_pegawai;
        this.nama = nama_pegawai;
    }
}
```

Output:

```
C:\Users\USER\OOP\Praktikum\Praktikum 7\Tugas\Tugas1>javac *.java

C:\Users\USER\OOP\Praktikum\Praktikum 7\Tugas\Tugas1>java Pegawai
Error: Main method not found in class Pegawai, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

Penjelasan :
Kode program yang diberikan merupakan definisi dari sebuah kelas Java yang dinamakan `Pegawai`. Tujuan utama dari kelas ini adalah untuk merepresentasikan

data seorang pegawai. Dalam kelas ini, terdapat dua variabel anggota (instance variables) yang digunakan untuk menyimpan informasi pegawai, yaitu `nip` (Nomor Induk Pegawai) yang bertipe data integer dan `nama` yang bertipe data string.

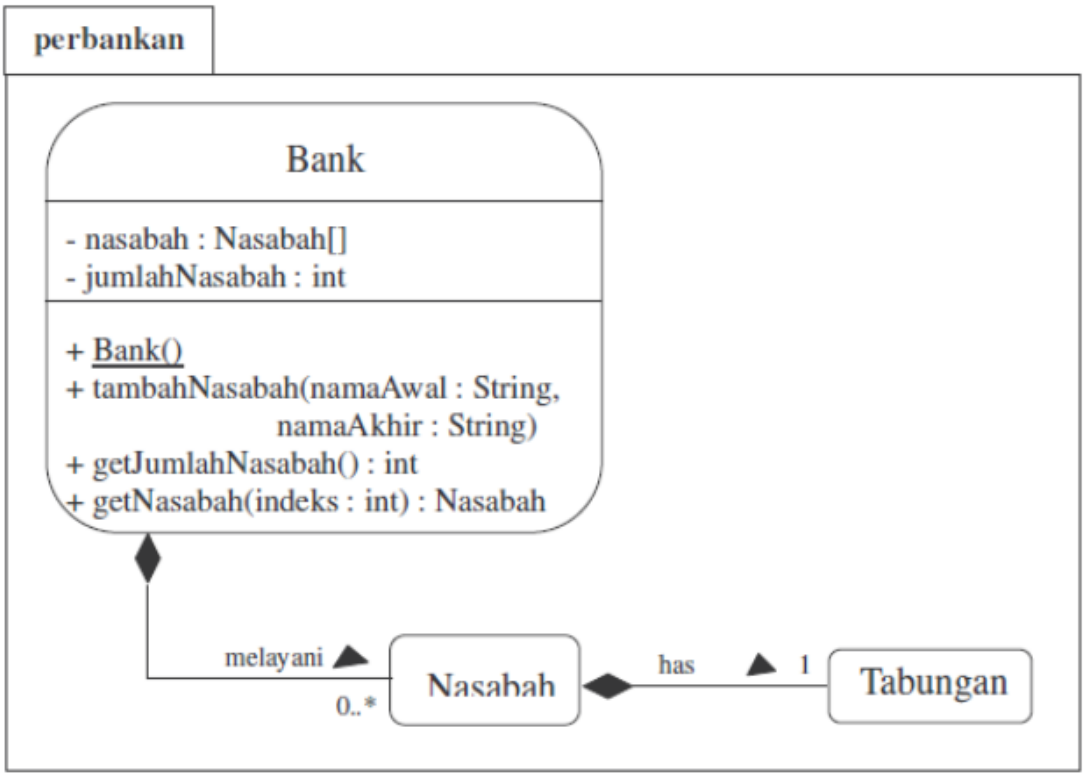
Kelas `Pegawai` ini juga memiliki dua konstruktor yang berbeda. Konstruktor pertama hanya menerima satu parameter, yaitu `nip_pegawai`, yang digunakan untuk menginisialisasi variabel `nip` pegawai. Namun, perhatikan bahwa konstruktor ini secara otomatis memanggil konstruktor kedua dengan mengirimkan `nip_pegawai` yang diberikan dan memberikan nilai default "NoName" sebagai nama pegawai. Dengan kata lain, jika kita membuat objek `Pegawai` dengan menggunakan konstruktor pertama, maka objek tersebut akan memiliki NIP yang telah ditentukan tetapi nama default "NoName".

Konstruktor kedua memungkinkan kita untuk menginisialisasi baik NIP maupun nama pegawai sesuai dengan nilai yang diberikan sebagai parameter. Dengan adanya overloading konstruktor ini, kita dapat membuat objek `Pegawai` dengan atau tanpa menentukan nama pegawai secara eksplisit.

Dengan demikian, kelas `Pegawai` ini menyediakan fleksibilitas dalam merepresentasikan pegawai dengan atribut NIP dan nama, serta memberikan nilai default jika nama tidak ditentukan saat membuat objek.

Namun setelah dikompile terdapat error, karena dalam program tersebut belum terdapat method main, yang mana method main ini merupakan titik awal eksekusi dalam program java saat dijalankan/dikompile.

2. Mengembangkan package perbankan dengan tambahan class Bank



Transformasikan class diagram diatas ke dalam bentuk program! Tulislah listing program berikut ini sebagai pengetesan.

Berikut Kode Program java untuk class Diagram Bank :

```
package perbankan;

public class Bank {
    private Nasabah[] nasabah;
    private int jumlahNasabah;

    public Bank() {
        nasabah = new Nasabah[10]; // Kapasitas awal array nasabah
(bisa disesuaikan)
        jumlahNasabah = 0;
    }

    public void tambahNasabah(String namaAwal, String namaAkhir) {
        if (jumlahNasabah < nasabah.length) {
            Nasabah newNasabah = new Nasabah(namaAwal, namaAkhir);
            nasabah[jumlahNasabah] = newNasabah;
            jumlahNasabah++;
        } else {
            System.out.println("Kapasitas nasabah penuh.");
        }
    }

    public int getJumlahNasabah() {
        return jumlahNasabah;
    }

    public Nasabah getNasabah(int indeks) {
        if (indeks >= 0 && indeks < jumlahNasabah) {
            return nasabah[indeks];
        } else {
            System.out.println("Indeks nasabah tidak valid.");
            return null;
        }
    }
}
```

Analisa:
kelas Bank tersebut memiliki atribut:

1. nasabah: Sebuah array dari objek Nasabah yang digunakan untuk menyimpan data nasabah.
2. jumlahNasabah: Sebuah integer yang digunakan untuk melacak jumlah nasabah yang ada dalam array.

Selain atribut juga terdapat metode-metode:

1. Konstruktor Bank: Digunakan untuk menginisialisasi array nasabah dengan kapasitas awal yang dapat disesuaikan.
2. tambahNasabah: Menggunakan parameter namaAwal dan namaAkhir untuk membuat objek Nasabah baru dan menambahkannya ke dalam array nasabah jika kapasitas masih cukup.
3. getJumlahNasabah: Mengembalikan jumlah nasabah yang ada dalam array.
4. getNasabah: Mengembalikan objek Nasabah berdasarkan indeks yang diberikan jika indeks valid, atau mengembalikan null jika indeks tidak valid.

Kode program *TesTugas.java*

```
import perbankan.*;

public class TesTugas {
    public static void main(String[] args) {
        Bank bank = new Bank();
        bank.tambahNasabah("Agus", "Daryanto");
        bank.getNasabah(0).setTabungan(new Tabungan(5000));
        bank.tambahNasabah("Tuti", "Irawan");
        bank.getNasabah(1).setTabungan(new Tabungan(7000));
        bank.tambahNasabah("Ani", "Ratna");
        bank.getNasabah(2).setTabungan(new Tabungan(4000));
        bank.tambahNasabah("Bambang", "Darmawan");
        bank.getNasabah(3).setTabungan(new Tabungan(6500));

        System.out.println("Jumlah Nasabah = " +
bank.getJumlahNasabah());

        for (int i = 0; i < bank.getJumlahNasabah(); i++) {
            System.out.println("Nasabah ke-" + (i + 1) + " : " +
                    bank.getNasabah(i).getNamaAwal() + " " +
                    bank.getNasabah(i).getNamaAkhir() + " ; Saldo = "
+
                    bank.getNasabah(i).getTabungan().getSaldo());
        }
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti diatas.

```
Jumlah nasabah = 4
Nasabah ke-1 : Agus Daryanto ; Saldo = 5000
Nasabah ke-2 : Tuti Irawan ; Saldo = 7000
Nasabah ke-3 : Ani Ratna ; Saldo = 4000
Nasabah ke-4 : Bambang Darwaman ; Saldo = 6500
```

Analisa kode program *TesTugas.java*:
Kode program `TesTugas` ini digunakan untuk menguji fungsionalitas dari kelas `Bank` dan objek-objek yang terkait dengannya. Pertama, sebuah objek `Bank` dengan nama `bank` dibuat. Kemudian, beberapa nasabah ditambahkan ke dalam bank dengan menggunaan metode `tambahNasabah`, dan saldo tabungan nasabah-nasabah tersebut diatur menggunakan metode `setTabungan`. Setelah semua nasabah dan saldo tabungan ditambahkan, jumlah nasabah dalam bank dicetak ke layar. Selanjutnya, program melakukan iterasi melalui nasabah-nasabah dalam bank menggunakan loop `for` dan mencetak informasi seperti nama awal, nama akhir, dan saldo tabungan nasabah-nasabah tersebut. Kode ini merupakan contoh penggunaan kelas-kelas terkait dalam suatu sistem perbankan dan mengilustrasikan bagaimana data nasabah dan tabungan dapat dikelola serta ditampilkan melalui objek-objek tersebut dalam program Java.

Output:

```
C:\Users\USER\OOP\Praktikum\Praktikum 7\Latihan>javac *.java

C:\Users\USER\OOP\Praktikum\Praktikum 7\Latihan>java TesTugas
Jumlah Nasabah = 4
Nasabah ke-1 : Agus Daryanto ; Saldo = 5000
Nasabah ke-2 : Tuti Irawan ; Saldo = 7000
Nasabah ke-3 : Ani Ratna ; Saldo = 4000
Nasabah ke-4 : Bambang Darmawan ; Saldo = 6500
```

D. KESIMPULAN

Mengelola kelas dengan baik dalam OOP adalah kunci untuk membangun perangkat lunak yang kuat, mudah dipahami, dan mudah dipelihara. Itu membantu dalam mengorganisir dan mengelola kompleksitas dalam pengembangan perangkat lunak, serta meningkatkan efisiensi dan kualitas kode.