LAPORAN PROJECT AKHIR

PEMROGRAMAN BERORIENTASI OBJEK

PROGRAM PERHITUNGAN TAGIHAN PEMBAYARAN LES

Disusun oleh:

NADIA MAHARANI ZULVIKA	**(L200220016)**
DIAN SASYA FITRIANI	**(L200220021)**
NASYAWA ADESTY RIYANNI	**(L200220023)**
ADINDA AULIA HAPSARI	**(L200220037)**

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS KOMUNIKASI DAN INFORMATIKA

UNIVERSITAS MUHAMMADIYAH SURAKARTA

TAHUN 2023/2024

BAB 1

PENDAHULUAN

1.1. Latar Belakang

Perkembangan zaman pada saat ini sudah sangat pesat dan modern. Banyak sekali tenaga kerja yang tergantikan dengan teknologi dan mesin cerdas. Hal ini tentu sangat membantu keberlangsungan hidup manusia di zaman ini. Teknologi – teknologi tersebut sudah pasti dirancang sedemikian rupa supaya dapat bekerja dengan maksimal dan dapat menguntungkan manusia dalam kehidupan sehari – harinya. Pada zaman ini tidak semua kegiatan harus dilakukan dengan bertatap muka. Dengan adanya teknologi, jaringan, dan sumber daya manusia semua bisa dilakukan tanpa adanya pertemuan antara kedua belah pihak. Banyaknya interaksi yang terjadi di dunia ini apalagi yang berkaitan dengan sebuah instansi atau perusahaan tentu membutuhkan sebuah hubungan yang terstruktur dan terorganisir dengan baik, seperti interaksi manusia yang terjadi pada sebuah badan pendidikan, badan kesehatan, organisasi, dan lain - lain. Berbagai instansi atau organisasi tersebut sudah banyak yang memanfaatkan kemajuan teknologi saat ini untuk membantu kinerjanya dan memudahkan mereka dalam berkomunikasi dengan para costumernya.

Oleh karena itu, berdasarkan realita yang terjadi saat ini kelompok kami bermaksud untuk membuat sebuah project pemrograman yang dapat memudahkan para costumer untuk mengetahui tagihan biaya pada sebuah tempat les untuk memudahkan terjalinnya suatu interaksi dan komunikasi secara efektif dan inovatif . Maksud dan tujuan dari adanya pembuatan project ini adalah untuk memenuhi tugas akhir dari Mata Kuliah Pemrograman Berorientasi Obyek dan kami berharap akan mendapatkan hasil yang maksimal melalui project ini.

Dengan adanya sistem tagihan pembayaran ini, peserta les akan dengan mudah mengetahui jumlah tagihan yang harus dibayarnya sesuai dengan kategori les musik yang diambilnya. Sistem tagihan pembayaran secara online ini akan menampilkan bar nomor anggota, nama anggota, kategori les music yang diambilnya, dan jumlah pertemuan yang diambil oleh peserta les. Setelah mengisi data – data tersebut, peserta akan dapat mengetahui berapa jumlah tagihan yang harus ia bayarkan berdasarkan jumlah pertemuan les dan kategori les yang ia ambil.

1.2. Tujuan

Berikut tujuan dari pembuatan program tagihan pembayaran :

1. Terpenuhinya nilai tugas akhir mata kuliah Pemrograman Berorientasi Objek (PBO).
2. Untuk mengimplementasikan materi perkuliahan PBO yang telah dipelajari.
3. Untuk mengetahui dan memahami proses berlangsungnya sistem PBO.

1.3. Manfaat

Berikut manfaat dari pembuatan program tagihan pembayaran les :

1. Tercapainya sistem tagihan pembayaran yang mudah diakses kapan saja dan dimana saja.
2. Dapat membuat program berbasis PBO dan mengembangkannya.
3. Meningkatkan pemahaman dan keterampilan mahasiswa mengenai OOP.
4. Mengetahui cara pembuatan suatu aplikasi dengan konsep OOP.
5. Mendapatkan pengalaman tentang pengimplementasian OOP.

1.4. Luaran

Luaran yang diharapkan dari pembuatan program tagihan pembayaran ini yaitu berupa program aplikasi tagihan pembayaran yang menggunakan bahasa pemrograman java yang telah dibuat menggunkan aplikasi Netbeans dan mencakup pokok pembahasan PBO yang telah disepakati, meliputi *class, object, package, variable, method, constructor, access modifier, inheritance, encapsulation, polymorphism,* dan sebagainya.

BAB 2

KONSEP

2.1. Konsep PBO yang Diimplementasikan

Konsep aplikasi system tagihan pembayaran yang kami buat menggunakan prinsip Pemrograman Berorientasi Objek (PBO) yang meliputi beberapa point yaitu *class, object, package, variable, method, constructor, access modifier, inheritance, encapsulation, polymorphism,* dan sebagainya. Tampilan program yang kami buat berisi beberapa bagoan, diantaranya :

a. Keterangan Nama Program

   Dalam program ini terdapat keterangan mengenai program yang dibuat.

b. Menu Data Diri

   Dalam tampilan data diri memuat informasi data diri yang harus diisi oleh user, meliputi nomor anggota dan nama anggota.

c. Menu Pilihan dan Jumlah Pertemuan

   Dalam tampilan menu terdapat opsi untuk memilih beberapa bimbingan yang telah disediakan, diantaranya biola, piano, gitar, harpa, biola, drum, dan Paduan suara. Selanjutnya, dalam menu ini terdapat kolom untuk mengisi jumlah pertemuan.

d. Tombol

   Program ini memiliki dua menu, yaitu menu total (untuk menghitung jumlah tagihan) dan menu reset (menghapus input sebelumnya).

e. Jendela Informasi

   Program ini akan menampilkan jendela informasi yang berisi data diri serta jumlah tagihan setelah user menekan menu total. Sedangkan, bila input nomor anggota yang dimasukkan oleh user salah, maka akan keluar jendela peringatan.

   Apabila user memasukkan jumlah pertemuan kurang dari 1, maka hasil yang ditampilkan pada jumlah tagihan adalah 0.

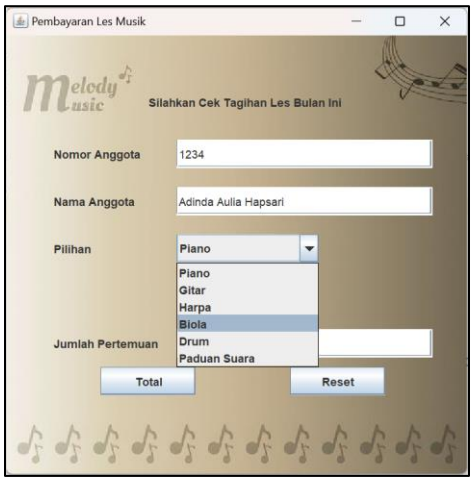
BAB 3

HASIL

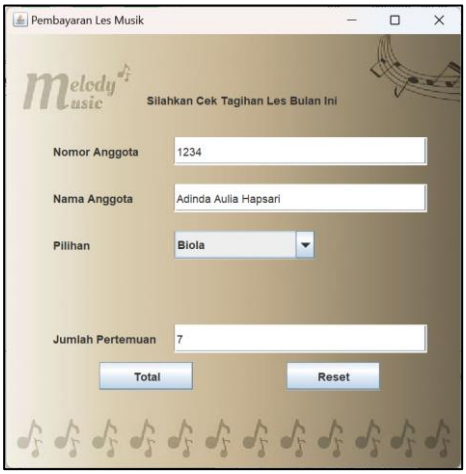
3.1. Tampilan Output Program

Berikut adalah output dari program ketika dijalankan.

Gambar 3.1.1 Tampilan Awal

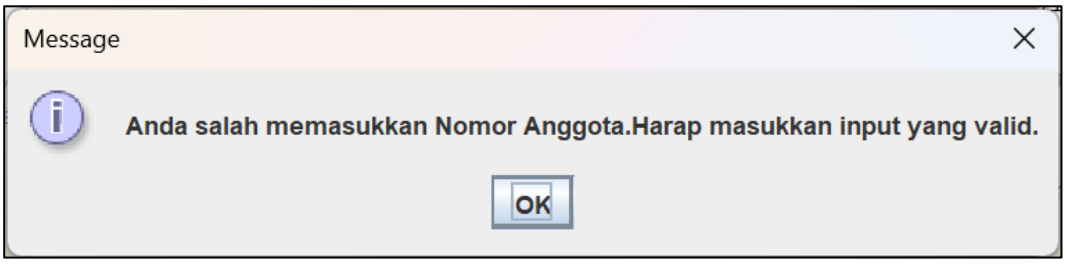


Gambar 3.1.2 Pengisian Data



Gambar 3.1.3 Isi Jumlah Pertemuan, Total

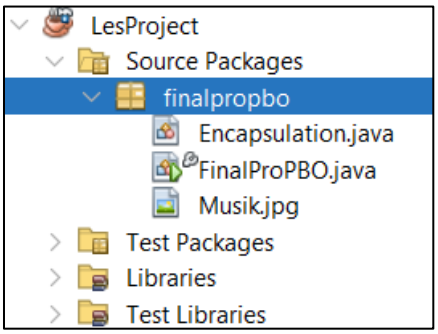
Gambar 3.1.4 Jendela Informasi



Gambar 3.1.5 Kotak Peringatan apabila salah memasukkan Nomor Anggota

Gambar 3.1.6 Jendela Informasi apabila mengisi Jumlah Pertemuan kurang dari 1

3.2. Kode Program



Gambar 3.2.1 Package dan Class

Project ini memiliki *package* finalpropbo yang memiliki *class* antara lain *Encapsulation,* FinalProPBO dan Musik.jpg. setiap kelas memiliki fungsi masing-masing.

a. Class FinalProPBO

```
//membuat class Container dengan hubungan inheritance atau extends dari javax
class Container extends javax.swing.JFrame{
    public Container(){             //constructor untuk memanggil container
        super(title: "Pembayaran Les Musik");
        //Local Variable
        //untuk mengatur size frame
        int sizeW = 500;
        int sizeH = 500;
        setSize(width: sizeW, height: sizeH);
        setDefaultCloseOperation(operation: EXIT_ON_CLOSE);
        setLocationRelativeTo(c: null);
        //Menambahkan gambar pada background program
        ImageIcon backgroundImage = new ImageIcon(location: getClass().getResource(name: "Musik.jpg"));
        JLabel background = new JLabel(image: backgroundImage);
        background.setLayout(new BorderLayout());
        setContentPane(contentPane:background);
    }
}
```

Gambar 3.2.2 Class FinalProPBO

Class

Class merupakan template atau cetakan untuk sekelompok objek yang memiliki properti yang sama. Kode program diatas memiliki class Bernama *Container* yang memiliki hubungan dengan javax.

Inheritance

Inheritance pada kode diatas berfungsi untuk membuat frame awal dari GUI. *Inheritance* adalah proses pewarisan data dan method dari suatu *class* yang telah ada kepada suatu *class* baru. Ukuran frame project yaitu 500x500. Didalam variable ini juga terdapat kode untuk menambahkan gambar pada background project.

Variable (Local Variable)

Variabel merupakan member dari class dimana setiap variabel harus memiliki tipe data tertentu. Variabel lokal merupakan variabel yang dideklarasikan di dalam sebuah method, konstruktor ataupun dalam suatu blok program dan hanya bisa digunakan oleh method yang mendeklarasikannya. Kode program diatas digunakan untuk mengatur ukuran frame project, yaitu 500x500.

Constructor

Pada kode program diatas terdapat variable *constructor* untuk menginisialisasi variabel- variabel instans yang akan dimiliki oleh objek. Ketentuan *constructor* sendiri harus memiliki nama yang sama dengan *class* nya.

- Kode super(title :) digunakan untuk mengisi judul pada tab atas project.
- setDefaultCloseOperation berfungsi apabila kita menutup aplikasi, maka program akan break.
- setLocationRelative digunakan untuk mengatur posisi program agar saat dijalankan berada di Tengah.

```
//membuat class abstrak masukan untuk membuat suatu objek agar memiliki banyak bentuk
abstract class Masukkan{        //polymorphism
    String NamaAnggota;
    protected void labelMasukkan(){
    }
    protected void labelMasukkan(String NamaAnggota){ //Overloading
        this.NamaAnggota = NamaAnggota;
    }
}

//membuat class NoAnggota sebagai sub class dari Masukkan
class NoAnggota extends Masukkan{
    @Override
    protected void labelMasukkan(){         //protected sebagai access modifier.
        //membuat variabel dengan JLabel dengan text "Nomor Anggota" dan penempatannya
        NoA = new JLabel(text: "Nomor Anggota");
        NoA.setBounds(x: 50, y: 50, width: 150, height: 150);
    }
}

//membuat class NamaAnggota sebagai sub class dari Masukkan
class NamaAnggota extends Masukkan{
    @Override
    protected void labelMasukkan(){         //protected sebagai access modifier.
        //membuat variabel dengan JLabel dengan text "Nama Anggota" dan penempatannya
        NA = new JLabel(text: "Nama Anggota");
        NA.setBounds(x: 50, y: 100, width: 150, height: 150);
    }
}
```

Gambar 3.2.3 Class FinalProPBO

Class

Class merupakan template atau cetakan untuk sekelompok objek yang memiliki properti yang sama.

Dalam kode program diatas, kami membuat class abstrak Masukkan untuk membuat suatu objek agar memiliki banyak bentuk. Class NoAnggota dan NamaAnggota berperan sebagai subclass dari class Masukkan.

Polymorphism

Polymorphism adalah suatu object dapat memiliki berbagai bentuk, sebagai object dari *class* sendiri atau *object* dari superclassnya.

Dalam kode diatas kami menggunakan *polymorphism overriding, Overriding* terjadi ketika deklarasi method subclass dengan nama dan parameter yang sama dengan method dari superclassnya.

Dalam kode diatas kami menggunakan *polymorphism overloading. Overloading* terjadi ketika deklarasi method subclass dengan nama yang sama tetapi parameter yang dimiliki berbeda-beda.

Access Modifier

Dalam *overriding* terdapat *access modifier protected* bernama Masukkan, *protected* sendiri berfungsi untuk mewariskan variabel yang ada di *superclass* terhadap *childclass*. *Access modifier* adalah *keyword* yang menyatakan hak akses berfungsi untuk membatasi kepada siapa saja suatu obyek dapat diakses.

```
ic class FinalProPBO extends Container{     //public juga sebagai access modifier

//variable static untuk memanggil JLabel, JtextField, JList, JButton, JComboBox
static JLabel NoA,NA,Pil,JP,Ju;
static JTextField tf1,tf2,tf3;
static JList pilihanList;
static JButton proses, reset;
static JComboBox<String> comboBox;
```

Gambar 3.2.4 Class FinalProPBO

Access Modifier

Access modifier adalah *keyword* yang menyatakan hak akses berfungsi untuk membatasi kepada siapa saja suatu obyek dapat diakses.

Public class FinalProPBO merupakan *access modifier.*

Variable (Static Variable)

Static variable merupakan jenis variabel yang mempertahankan nilainya pada setiap pemanggilan *function.* Pada kode diatas digunakan untuk memanggil :

- JLabel (Mengkonstruksi objek label dengan teks)
- JtextField (Mengkonstruksi obyek dengan teks kosong)
- JList (Menampilkan sekumpulan objek dan memungkinkan untuk memilih salah satunya)
- JButton (Mengkonstruksi obyek button tanpa teks dan ikon)
- JComboBox (Membuat data dalam bentuk kotak kombo drop-down)

```
public static void main(String[] args) {
    Container ct = new Container();          //Membuat objek baru dari Container dengan nama ct
    Encapsulation ec = new Encapsulation(); //Membuat objek baru dari Encapsulation dengan nama ec

    //Membuat objek baru dari NoAnggota dan NamaAnggota
    NoAnggota no = new NoAnggota();
    NamaAnggota nama = new NamaAnggota();

    //memanggil method labelMasukkan milik class Masukkan yang merupakan kelas Parents dari NoAnggo
    no.labelMasukkan();
    nama.labelMasukkan();

    //membuat variabel dengan memanggil JTextField
    tf1 = new JTextField(); //JTextField untuk Nomor Anggota
    tf2 = new JTextField(); //JTextField untuk Nama Anggota
    tf3 = new JTextField(); //JTextField untuk jumlah pertemuan
    //membuat JTextFeild dengan x yang sama yaitu 180 dan y yang disesuaikan
    tf1.setBounds(x: 180,y: 110,width: 270,height:30);
    tf2.setBounds(x: 180,y: 160,width: 270,height:30);
    tf3.setBounds(x: 180,y: 310,width: 270,height:30);


    Pil = new JLabel(text: "Pilihan");
    Pil.setBounds(x: 50,y: 150,width: 150,height:150);
    Ju = new JLabel(text: "Silahkan Cek Tagihan Les Bulan Ini");
    Ju.setBounds(x: 150, y: 20, width: 300, height:30);
```

Gambar 3.2.5 Class FinalProPBO

Object

Object merupakan representasi nyata dari sebuah class.

Class

Class merupakan template atau cetakan untuk sekelompok objek yang memiliki properti yang sama. *Class* bisa terdiri dari *Fields Template* dari keadaan (property) yang bisa membedakan objek yang satu dengan yang lainnya.

Kode diatas digunakan untuk membuat objek baru dari *container* dan *encapsulation,* dan nama anggota serta nomor anggota. Selanjutnya, akan dipangil method LabelMasukan milik *class* Masukan yang merupakan *class parents* dari objek yang telah dibuat.

- JTextField digunakan untuk mengkonstruksi obyek dengan teks kosong.
- JLabel digunakan untuk membuat label text pada program.

```
//membuat pilihan dengan menggunakan comboBox
String[] pilihan = {
    "Piano",
    "Gitar",
    "Harpa",
    "Biola",
    "Drum",
    "Paduan Suara"
};
comboBox = new JComboBox<>(items: pilihan);
comboBox.setBounds(x: 180, y: 210, width: 150, height: 30);

//membuat variabel untuk memanggil Jlabel dan penempatannya
JP = new JLabel(text: "Jumlah Pertemuan");
JP.setBounds(x: 50,y: 250,width: 150,height: 150);

//membuat variabel untuk memanggil JButton dan penempatannya
proses = new JButton(text: "Total");
proses.setBounds(x: 100,y: 350,width: 100,height: 30);
```

Gambar 3.2.6 Class FinalProPBO

Kode program ditas digunakan untuk membuat pilihan bimbingan belajar. ComboBox digunakan untuk membuat data dalam bentuk kotak kombo drop-down. JLabel digunakan untuk membuat teks variabel berupa Jumlah Pertemuan, sedangkan JButton digunakan untuk membuat tombol Total tagihan. setBounds digunakan untuk mengatur ukuran dan penempatan objek yang telah dibuat.

```
//membuat program proses yang dapat memproses sesuai yang diinginkan
proses.addActionListener(new ActionListener(){
    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            int nomorAnggota = Integer.parseInt(s: tf1.getText());
            ec.setNoAnggota(noAnggota: nomorAnggota);
            String NamaAnggota = tf2.getText();
            String pilihanLes = (String) comboBox.getSelectedItem();
            int hariMasuk = Integer.parseInt(s: tf3.getText());

            int tagihan = 0;
            if (pilihanLes.equals(anObject: "Piano")) {
                tagihan = 170000;
            } else if (pilihanLes.equals(anObject: "Gitar")) {
                tagihan = 100000;
            } else if (pilihanLes.equals(anObject: "Harpa")) {
                tagihan = 200000;
            } else if (pilihanLes.equals(anObject: "Biola")) {
                tagihan = 250000;
            } else if (pilihanLes.equals(anObject: "Drum")) {
                tagihan = 130000;
            } else if (pilihanLes.equals(anObject: "Paduan Suara")) {
                tagihan = 1500000;
            }
```

Gambar 3.2.7 Class FinalProPBO

Selanjutnya, kami membuat program untuk memproses sesuai dengan perintah yang sudah kami konsep dan kode untuk membuat setelan awal harga setiap bimbingan yang diambil.

- getText digunakan untuk mengambil teks yang diinput pada kolom nomorAnggota maupun NamaAnggota.
- ActionListener berfungsi agar button reset memiliki aksi, yaitu apabila ditekan maka inputan yang sudah diketikkan akan terhapus.
- ActionPerformed digunakan untuk memanggil method dari class parent.

```
            JOptionPane.showMessageDialog(parentComponent:ct,
                    "Nomor Anggota    : " + ec.getNoAnggota() + "\n" +
                            "Nama Anggota      : " + NamaAnggota + "\n" +
                            "Pilihan                    : " + pilihanLes + "\n" +
                            "Total                       : " + hitungTagihan(a: hariMasuk, b: tagihan));
        }
        // menampilkan message agar saat inputan No anggota dan hari masuk salah akan keluar message peringatan
        catch (NumberFormatException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(parentComponent:ct, "Anda salah memasukkan Nomor Anggota."
                    + "Harap masukkan input yang valid.");
            resetFrame();
        }
    }
});
```

Gambar 3.2.8 Class FinalProPBO

Selanjutnya, kami membuat *table massage* untuk memperlihatkan hasil akhir menggunakan JOptionPane, dan table message untuk menampilkan peringatan apabila inputan nomor anggota bukan integer dan jumlah pertemuan kurang dari 0.

- JOptionPane merupakan sebuah kelas yang menyediakan jendela dialog.

```
    //membuat variabel untuk memanggil JButton dan penempatannya
    reset = new JButton(text: "Reset");
    reset.setBounds(x: 300,y: 350,width: 100,height:30);

    //membuat program reset yang dapat mereset menjadi seperti tampilan awal
    reset.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            resetFrame();
        }
    });
    //memanggil semua variabel yang sudah dibuat dan menempatkannya sesuai dengan yang
    ct.add(comp:NoA);
    ct.add(comp:NA);
    ct.add(comp:Pil);
    ct.add(comp:JP);
    ct.add(comp:Ju);
    ct.add(comp:tf1);
    ct.add(comp:tf2);
    ct.add(comp:tf3);
    ct.add(comp:comboBox);
    ct.add(comp:proses);
    ct.add(comp:reset);
    ct.setLayout(manager: null);
    ct.setVisible(b: true);
}
```

Gambar 3.2.9 Class FinalProPBO

Pada kode program diatas, kami membuat variabel tombol Reset menggunakan JButton.

- ActionListener berfungsi agar button reset memiliki aksi, yaitu apabila ditekan maka inputan yang sudah diketikkan akan terhapus.
- ActionPerformed digunakan untuk memanggil method dari class parent.
- setLayout digunakan untuk menentukan jenis layout yang akan ditampilkan.
- setVisible digunakan untuk mengatur output agar dapat ditampilkan.

```
    public static int hitungTagihan(int a, int b) {
        int result = a * b;
        // membuat limit inputan untuk hari masuk,
        //jika memasukkan kurang dari 1 maka saat di run hasilnya akan tetap 0
        return Math.max(a: result, b: 0);
    }

    public static void resetFrame() {
        tf1.setText(t: "");
        tf2.setText(t: "");
        tf3.setText(t: "");
        comboBox.setSelectedIndex(anIndex:0);
        pilihanList.clearSelection();
    }
}
```

Gambar 3.2.10 Class FinalProPBO

Method (Non void)

Method non void merupakan method yang mengembalikan suatu nilai setelah dieksekus. Pada kode diatas, *method non void* berfungsi melakukan pengkalian harga sesuai bimbingan yang diambil dengan jumlah pertemuan.

Method (Void)

Method void adalah method yang tidak mengembalikan nilai apapun setelah dieksekusi, hanya melakukan sebuah proses. *Method void* pada kode tersebut berfungsi otomatis menghapus *text input* ketika button reset diklik.

Pada kode program diatas, kami membaut kode untuk limit input pada jumlah pertemuan. Jika user menginput kurang dari 1, maka hasilnya akan tetap 0.

b. *Class Encapsulation*

```
//mengubah noAnggota yang sebelumnya string menjadi bentuk int
public class Encapsulation {
    private int noAnggota; //variable instance, yaitu variable yang berada diluar kelas

    public int getNoAnggota(){       // Encapsulation (set dan get)
        return noAnggota;
    }
    public void setNoAnggota(int noAnggota){
        this.noAnggota = noAnggota;
    }
}
```

Gambar 3.2.11 Class Encapsulation

Encapsulation

Encapsulation merupakan teknik untuk melindungi atau membungkus data/variabel supaya tidak dapat diakses langsung dari class lain. Fungsi dari *getter* adalah untuk menampilkan private variabel, sedangkan *setter* digunakan untuk memodifikasi nilai *private variable.*

Variable (Instance Variable)

Instance variabel (variabel global) merupakan variabel milik dari suatu *class* dan berada diluar blok atau method tertentu, dan bisa diakses dari semua method yang menjadi member dari *class* sehingga semua method bisa memanipulasi nilai dari variabel global tersebut.

Pada kode program diatas, *return* noAnggota digunakan untuk mengakses nilai *field* NoAnggota pada *object*, sedangkan this.noAnggota digunakan untuk mengganti nilai *field* pada *object.*

BAB 4

KESIMPULAN

Sistem tagihan pembayaran merupakan program pembayaran yang nantinya akan digunakan di Bimbingan Melody Music. Program ini dibuat dengan maksud menjadi salah satu pengembang program tagihan pembayaran yang menginovasikan system yang lebih canggih, efektif, dan efisien serta memenuhi nilai tugas akhir mata kuliah Pemrograman Berorientasi Objek yang didalamnya meliputi penerapan beberapa point yaitu *class, object, package, variable, method, constructor, access modifier, inheritance, encapsulation, polymorphism,* dan sebagainya. Dengan adanya system tagihan pembayaran ini, kami berharap dapat membantu mempermudah proses tagihan dan pembayaran dan dapat menjadi terobosan baru dan menjadi program yang diminati dikalangan masyarakat.