

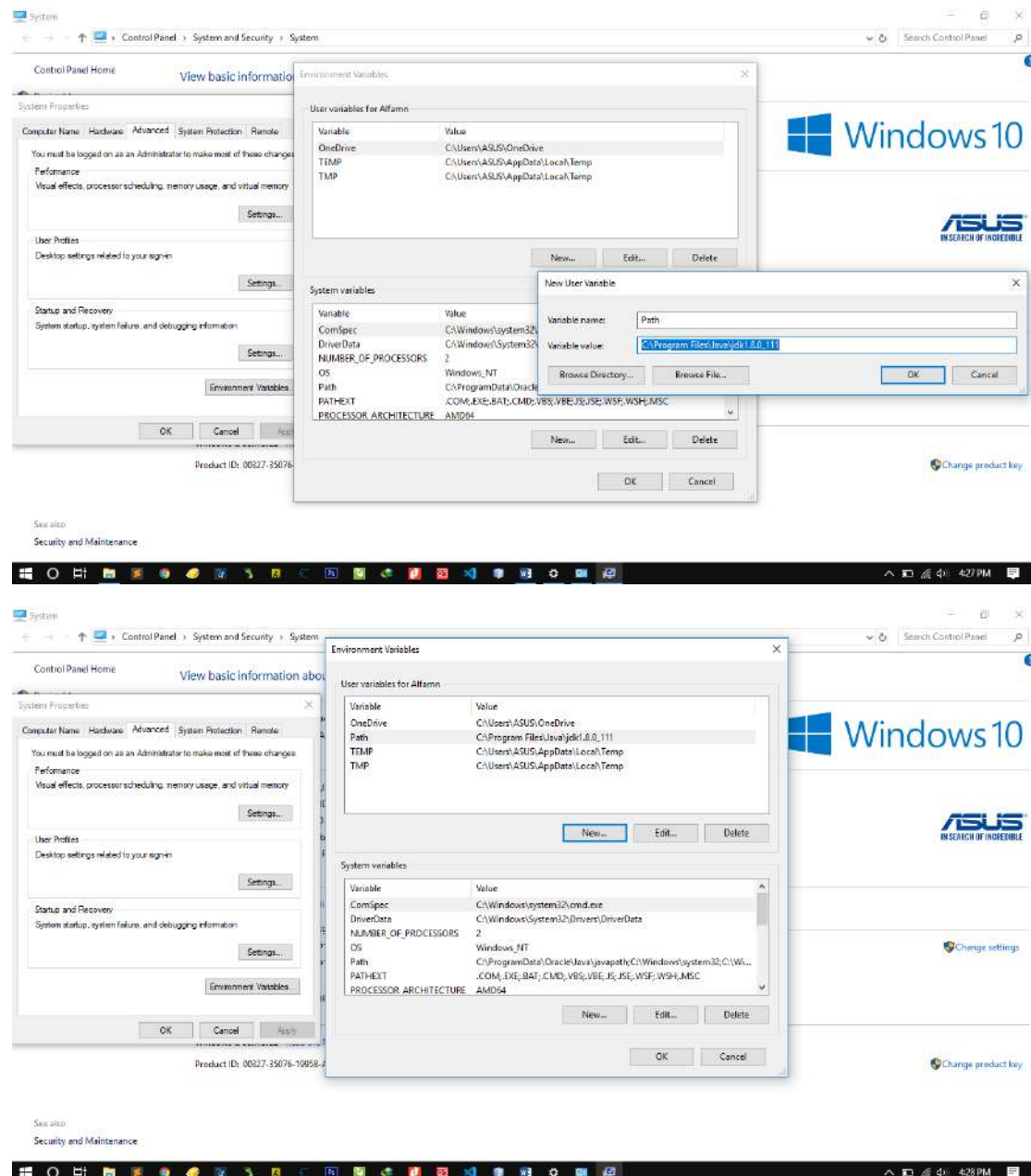
Nama : Alif Al Amin

NIM : L200180082

Kelas : B

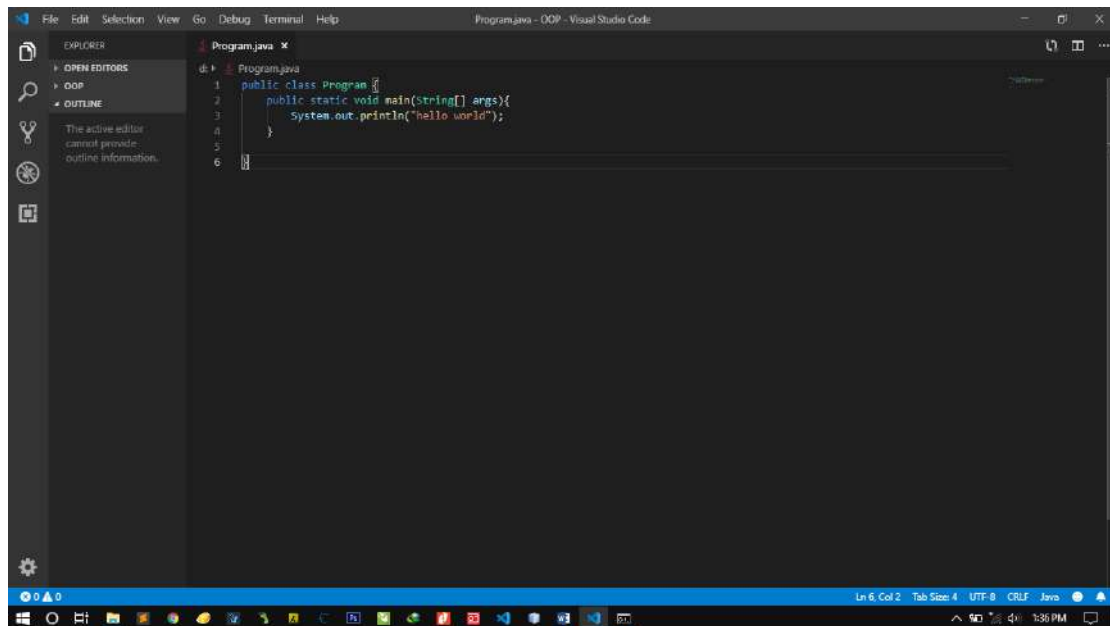
Laporan Modul 1

1. Menambahkan Path

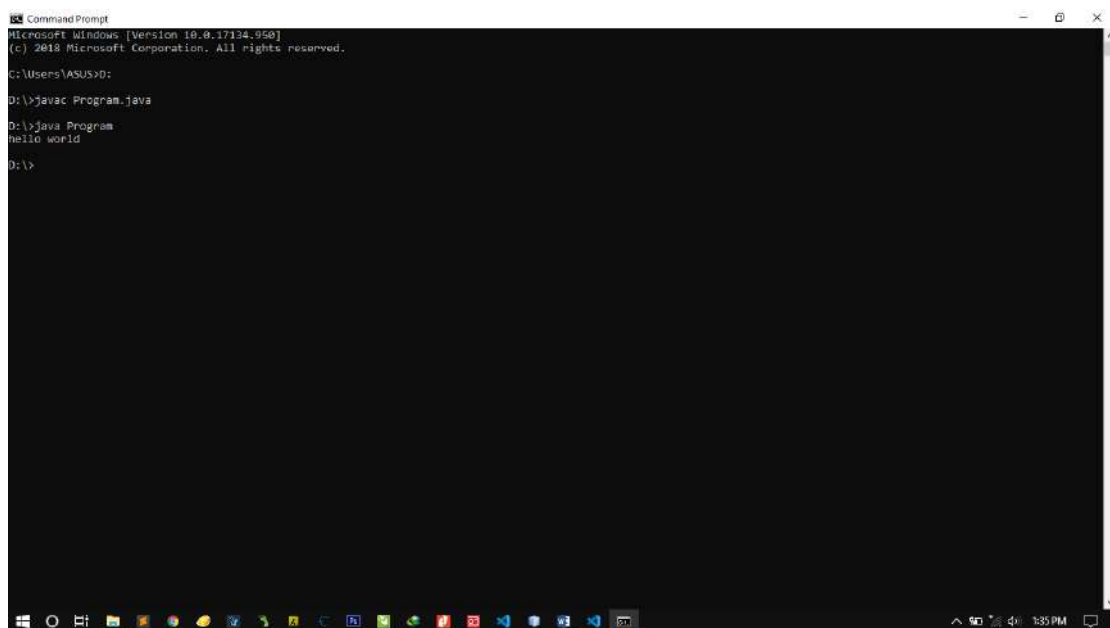


Tinggal klik 'OK'

2. Membuat class dan menjalankan lewat CMD



CMD:



3. Membuat class, membuat object dan dijalankan lewat CMD

Code :

The screenshot shows the Visual Studio Code editor with two Java files open. The left file, `Roti.java`, defines a `Roti` class with attributes `warna`, `rasa`, `berat`, and `harga`, and methods `beriWarna`, `beriRasa`, `timbangBerat`, `hargaJual`, and `infoRoti`. The right file, `RotiDemo.java`, contains a `main` method that creates a `Roti` object and calls its methods with specific values.

```
Roti.java
1 public class Roti{
2     String warna;
3     String rasa;
4     int berat;
5     double harga;
6
7     void beriWarna(String warnaRoti){
8         warna = warnaRoti;
9     }
10
11    void beriRasa(String rasaRoti){
12        rasa = rasaRoti;
13    }
14
15    void timbangBerat(int beratRoti){
16        berat = beratRoti;
17    }
18
19    void hargaJual(double hargaRoti){
20        harga = hargaRoti;
21    }
22
23    void infoRoti(){
24        System.out.println(
25            "Warna Roti : " + warna + "\n" +
26            "Rasa Roti : " + rasa + "\n" +
27            "Berat Roti : " + berat + "gr" + "\n" +
28            "Harga Roti : Rp " + harga);
29    }
30
31 }
32
33
```

```
RotiDemo.java
1 public class RotiDemo{
2     public static void main(String[] args){
3         Roti roti = new Roti();
4         roti.beriWarna("Hijau");
5         roti.beriRasa("Pandan");
6         roti.timbangBerat(30);
7         roti.hargaJual(6000);
8         roti.infoRoti();
9     }
10
11 }
12
13
14
15 }
```

CMD:

The screenshot shows a Windows Command Prompt window where the following commands are executed: `javac Roti.java`, `javac RotiDemo.java`, and `java RotiDemo`. The output of the `java RotiDemo` command displays the details of the `Roti` object: `Warna Roti : Hijau`, `Rasa Roti : Pandan`, `Berat Roti : 30gr`, and `Harga Roti : Rp 6000.0`.

```
Microsoft Windows [Version 10.0.17134.950]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ASUS>D:
D:\>javac Roti.java
D:\>javac RotiDemo.java
D:\>java RotiDemo
Warna Roti : Hijau
Rasa Roti : Pandan
Berat Roti : 30gr
Harga Roti : Rp 6000.0
D:\>
```

Nama : Alif Al Amin

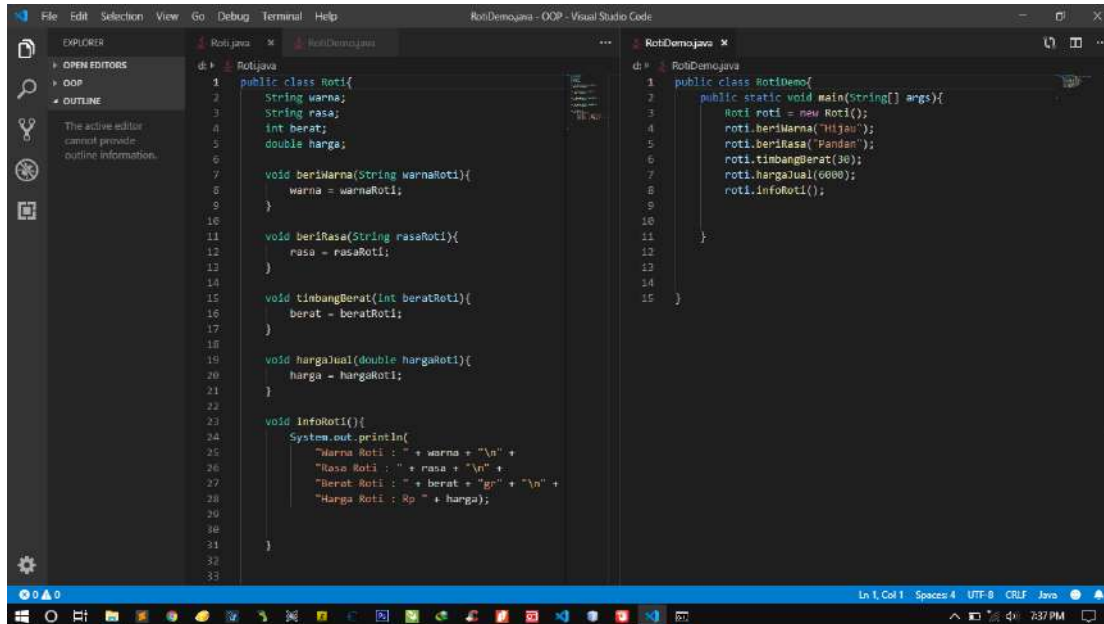
NIM : L200180082

Kelas : B

Modul 2

1. Membuat class Roti dan RotiDemo

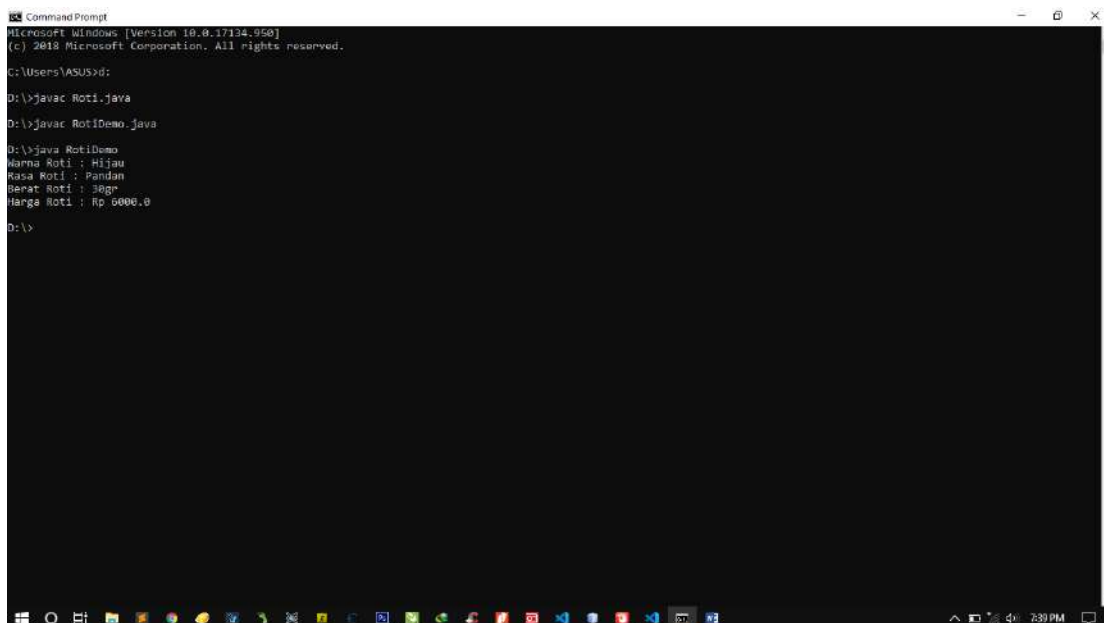
Code :



```
1 public class Roti{
2     String warna;
3     String rasa;
4     int berat;
5     double harga;
6
7     void beriWarna(String warnaRoti){
8         warna = warnaRoti;
9     }
10
11    void beriRasa(String rasaRoti){
12        rasa = rasaRoti;
13    }
14
15    void tinbangBerat(int beratRoti){
16        berat = beratRoti;
17    }
18
19    void hargaJual(double hargaRoti){
20        harga = hargaRoti;
21    }
22
23    void infoRoti(){
24        System.out.println(
25            "Warna Roti : " + warna + "\n" +
26            "Rasa Roti : " + rasa + "\n" +
27            "Berat Roti : " + berat + "gr" + "\n" +
28            "Harga Roti : Rp " + harga);
29    }
30
31 }
32
33 }
```

```
1 public class RotiDemo{
2     public static void main(String[] args){
3         Roti roti = new Roti();
4         roti.beriWarna("Hijau");
5         roti.beriRasa("Pandan");
6         roti.tinbangBerat(30);
7         roti.hargaJual(6000);
8         roti.infoRoti();
9     }
10
11 }
12
13
14
15 }
```

Hasil CMD :



```
Microsoft Windows [Version 10.0.17134.950]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ASUS>d:

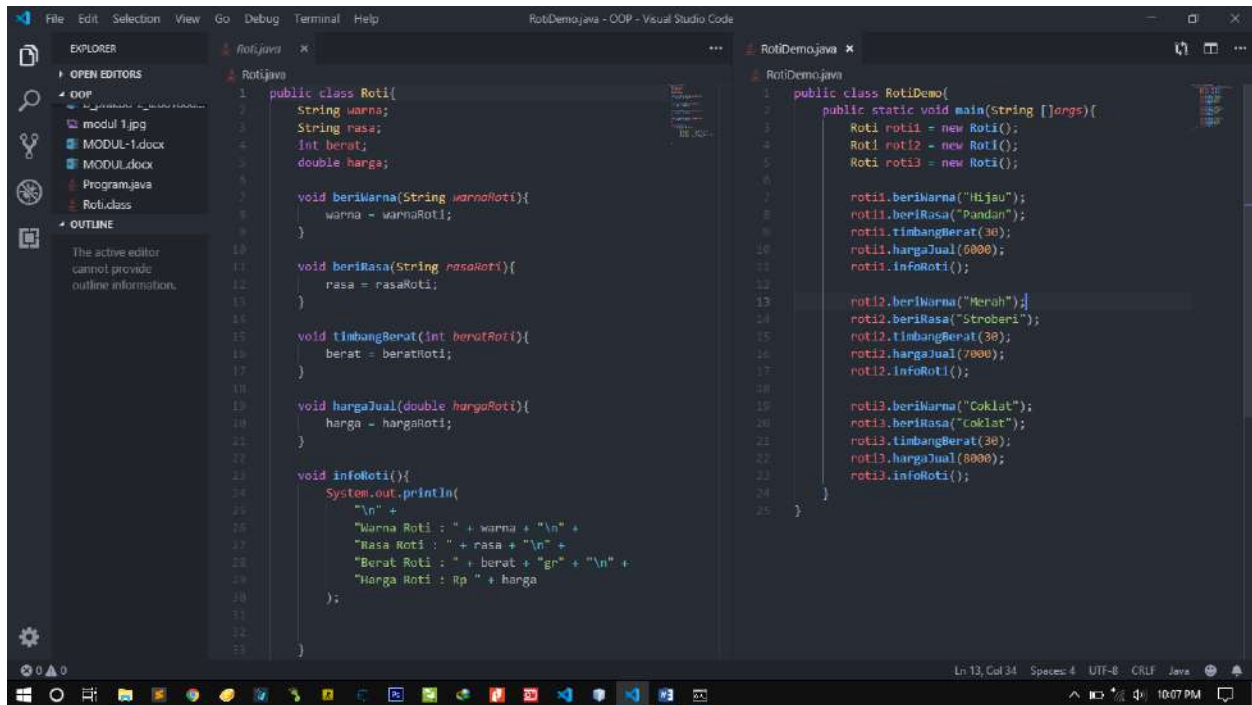
D:\>javac Roti.java

D:\>javac RotiDemo.java

D:\>java RotiDemo
Warna Roti : Hijau
Rasa Roti : Pandan
Berat Roti : 30gr
Harga Roti : Rp 6000.0

D:\>
```

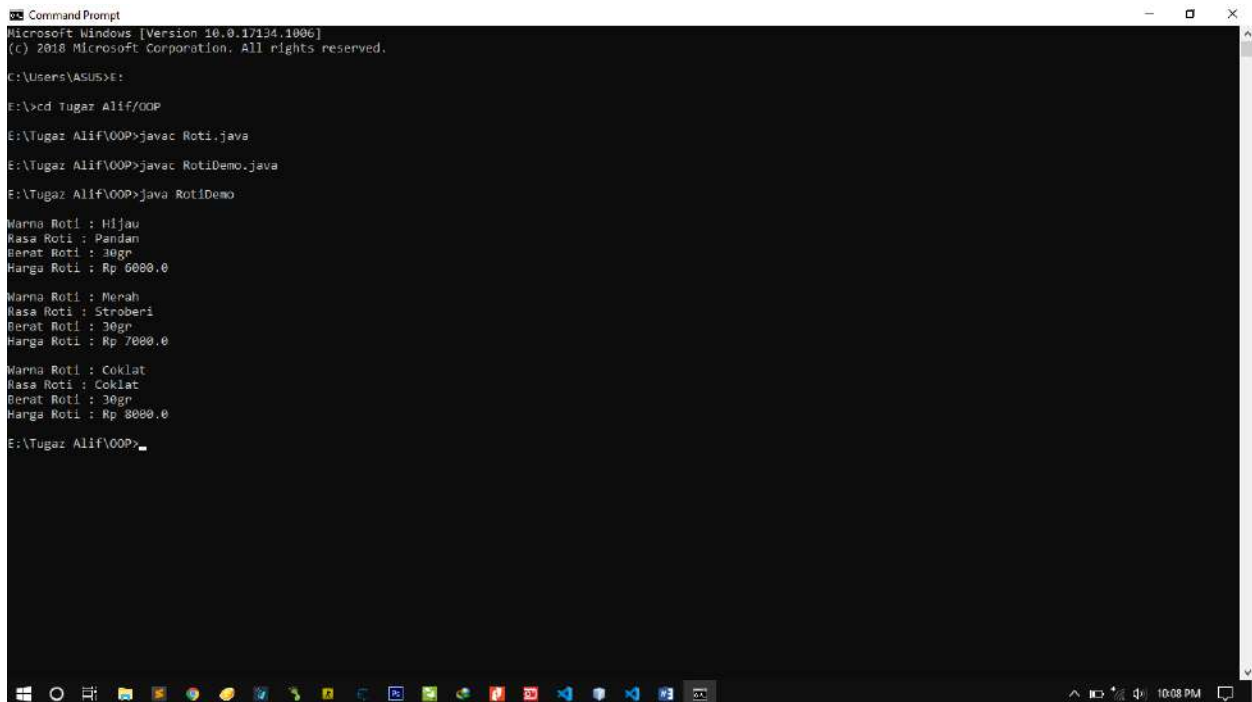
2. Memodifikasi class RotiDemo dan menambah 3 object



The screenshot shows the Visual Studio Code editor with two files open: `Roti.java` and `RotiDemo.java`. The `Roti.java` file contains a `Roti` class with attributes `warna`, `rasa`, `berat`, and `harga`, and methods `beriWarna`, `beriRasa`, `timbangBerat`, `hargaJual`, and `infoRoti`. The `RotiDemo.java` file shows the `main` method being updated to create three `Roti` objects (`roti1`, `roti2`, and `roti3`) and call their methods to set values and print information.

```
Roti.java
1 public class Roti{
2     String warna;
3     String rasa;
4     int berat;
5     double harga;
6
7     void beriWarna(String warnaRoti){
8         warna = warnaRoti;
9     }
10
11     void beriRasa(String rasaRoti){
12         rasa = rasaRoti;
13     }
14
15     void timbangBerat(int beratRoti){
16         berat = beratRoti;
17     }
18
19     void hargaJual(double hargaRoti){
20         harga = hargaRoti;
21     }
22
23     void infoRoti(){
24         System.out.println(
25             "\n" +
26             "Warna Roti : " + warna + "\n" +
27             "Rasa Roti : " + rasa + "\n" +
28             "Berat Roti : " + berat + "gr" + "\n" +
29             "Harga Roti : Rp " + harga
30         );
31     }
32 }

RotiDemo.java
1 public class RotiDemo{
2     public static void main(String [] args){
3         Roti roti1 = new Roti();
4         Roti roti2 = new Roti();
5         Roti roti3 = new Roti();
6
7         roti1.beriWarna("Hijau");
8         roti1.beriRasa("Pandan");
9         roti1.timbangBerat(30);
10        roti1.hargaJual(6000);
11        roti1.infoRoti();
12
13        roti2.beriWarna("Merah");
14        roti2.beriRasa("Stroberi");
15        roti2.timbangBerat(30);
16        roti2.hargaJual(7000);
17        roti2.infoRoti();
18
19        roti3.beriWarna("Coklat");
20        roti3.beriRasa("Coklat");
21        roti3.timbangBerat(30);
22        roti3.hargaJual(8000);
23        roti3.infoRoti();
24    }
25 }
```



The screenshot shows a Windows Command Prompt window where the `RotiDemo` program is executed. The output displays the details for three different roti objects: one with green color, pandan flavor, 30g weight, and 6000 price; another with red color, strawberry flavor, 30g weight, and 7000 price; and a third with chocolate color and flavor, 30g weight, and 8000 price.

```
Microsoft Windows [Version 10.0.17134.1006]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ASUS>E:

E:\>cd Tugaz Alif\OOP
E:\Tugaz Alif\OOP>javac Roti.java
E:\Tugaz Alif\OOP>javac RotiDemo.java
E:\Tugaz Alif\OOP>java RotiDemo

Warna Roti : Hijau
Rasa Roti : Pandan
Berat Roti : 30gr
Harga Roti : Rp 6000.0

Warna Roti : Merah
Rasa Roti : Stroberi
Berat Roti : 30gr
Harga Roti : Rp 7000.0

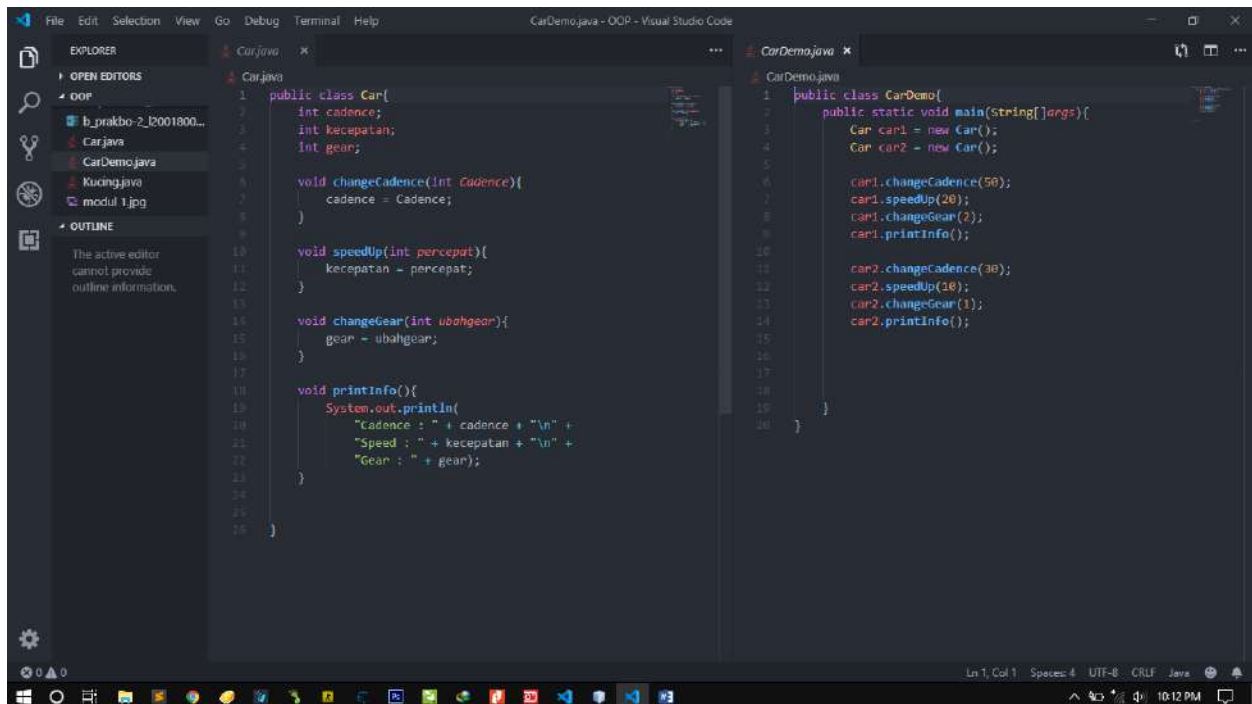
Warna Roti : Coklat
Rasa Roti : Coklat
Berat Roti : 30gr
Harga Roti : Rp 8000.0

E:\Tugaz Alif\OOP>
```

3. Gambar class diagram dari class RotiDemo

RotiDemo
#nama : String #rasa : String #berat : integer #harga : integer
#beriWarna() #beriRasa() #timbangBerat() #hargaJual()

4. Membuat satu class baru yang bisa digunakan sebagai template/blueprint dari class CarDemo, tidak memiliki fungsi main



```
Car.java
1 public class Car{
2     int cadence;
3     int kecepatan;
4     int gear;
5
6     void changeCadence(int cadence){
7         cadence = cadence;
8     }
9
10    void speedUp(int percepat){
11        kecepatan = percepat;
12    }
13
14    void changeGear(int ubahgear){
15        gear = ubahgear;
16    }
17
18    void printInfo(){
19        System.out.println(
20            "Cadence : " + cadence + "\n" +
21            "Speed : " + kecepatan + "\n" +
22            "Gear : " + gear);
23    }
24
25 }
26 }
```

```
CarDemo.java
1 public class CarDemo{
2     public static void main(String[] args){
3         Car car1 = new Car();
4         Car car2 = new Car();
5
6         car1.changeCadence(50);
7         car1.speedUp(20);
8         car1.changeGear(2);
9         car1.printInfo();
10
11        car2.changeCadence(30);
12        car2.speedUp(10);
13        car2.changeGear(1);
14        car2.printInfo();
15    }
16 }
```

```
Command Prompt
Microsoft Windows [Version 10.0.17134.1006]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ASUS>E:

E:\>cd Tugaz Alif\OOP

E:\Tugaz Alif\OOP>javac Car.java

E:\Tugaz Alif\OOP>javac CarDemo.java

E:\Tugaz Alif\OOP>java CarDemo
Cadence : 50
Speed : 20
Gear : 2
Cadence : 30
Speed : 10
Gear : 1

E:\Tugaz Alif\OOP>
```

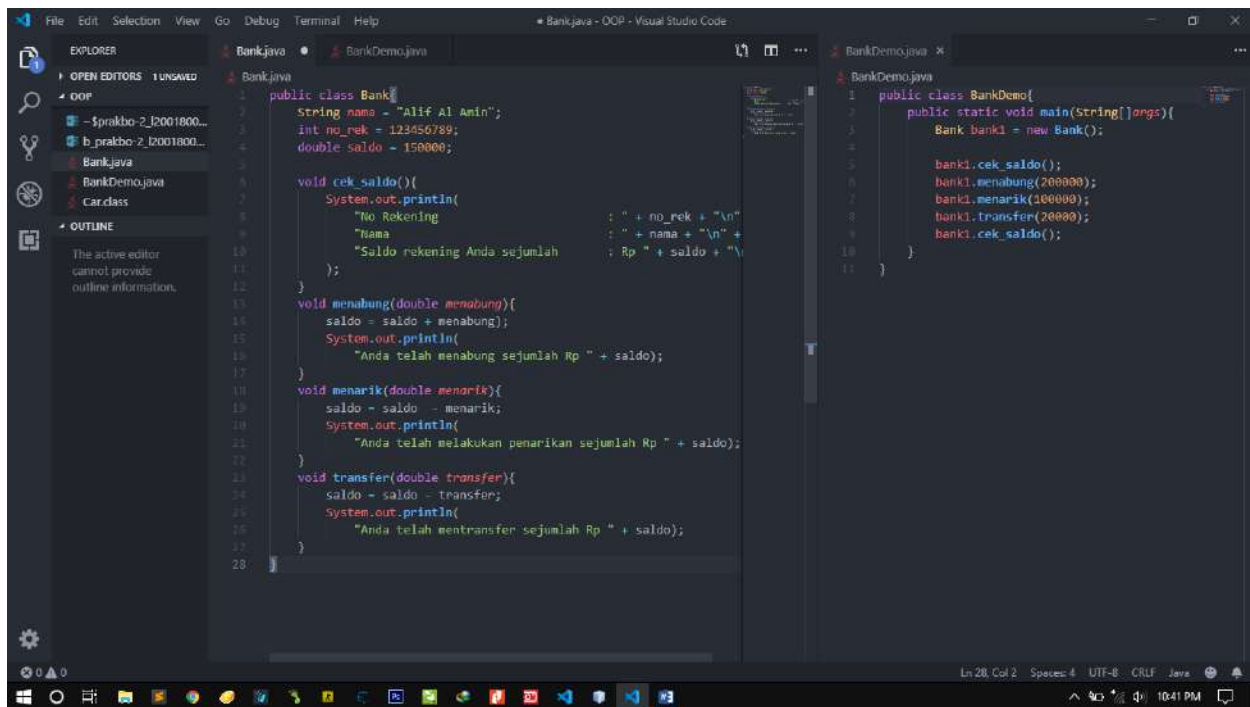
5. Membuat suatu class yang dapat merepresentasikan sifat-sifat dari object Kucing

```
File Edit Selection View Go Debug Terminal Help
Kucing.java - OOP - Visual Studio Code

EXPLORER
OPEN EDITORS 1 UNSAVED
OOP
  CarDemo.java
  Kucing.java
  modul 1.jpg
  MODUL-1.docx
  MODUL.docx
  Program.java
OUTLINE
The active editor cannot provide outline information.

Kucing.java
1 public class Kucing{
2     String warnabulu;
3     int umur;
4
5     Kucing Kucing1 = new Kucing();
6     void warnabulu(String warnabulu){
7         warnabulu = warnabulu;
8     }
9     void Umur(int usia){
10        umur = usia;
11    }
12
13 }
```

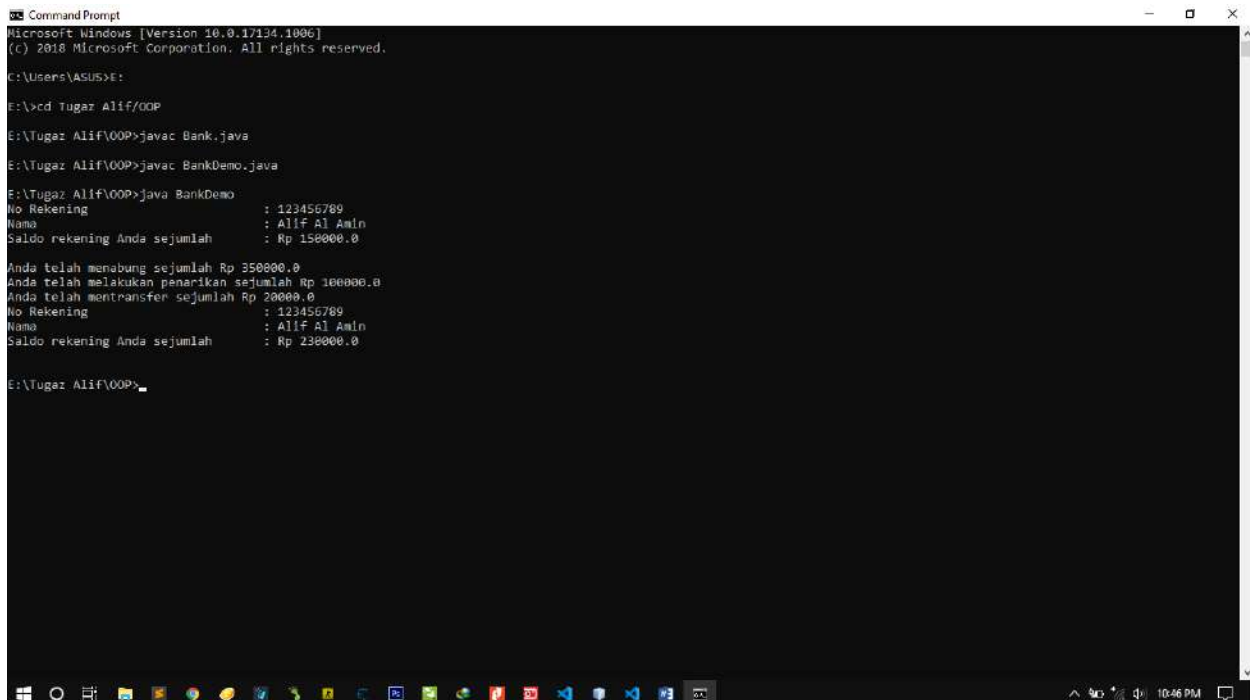
6. Bank dan BankDemo



The screenshot shows the Visual Studio Code editor with two files open: `Bank.java` and `BankDemo.java`. The `Bank.java` file contains a `Bank` class with attributes `nama`, `no_rek`, and `saldo`, and methods `cek_saldo`, `menabung`, `menarik`, and `transfer`. The `BankDemo.java` file contains a `BankDemo` class with a `main` method that creates a `Bank` object and performs a series of transactions: checking balance, depositing, withdrawing, and transferring.

```
Bank.java
1 public class Bank {
2     String nama = "Alif Al Amin";
3     int no_rek = 123456789;
4     double saldo = 150000;
5
6     void cek_saldo(){
7         System.out.println(
8             "No Rekening          : " + no_rek + "\n"
9             "Nama                  : " + nama + "\n"
10            "Saldo rekening Anda sejumlah : Rp " + saldo + "\n"
11        );
12    }
13    void menabung(double menabung){
14        saldo = saldo + menabung;
15        System.out.println(
16            "Anda telah menabung sejumlah Rp " + saldo;
17        );
18    }
19    void menarik(double menarik){
20        saldo = saldo - menarik;
21        System.out.println(
22            "Anda telah melakukan penarikan sejumlah Rp " + saldo;
23        );
24    }
25    void transfer(double transfer){
26        saldo = saldo - transfer;
27        System.out.println(
28            "Anda telah mentransfer sejumlah Rp " + saldo;
29        );
30    }
31 }
```

```
BankDemo.java
1 public class BankDemo {
2     public static void main(String[] args){
3         Bank bank1 = new Bank();
4
5         bank1.cek_saldo();
6         bank1.menabung(200000);
7         bank1.menarik(100000);
8         bank1.transfer(20000);
9         bank1.cek_saldo();
10    }
11 }
```



The screenshot shows a Windows Command Prompt window where the Java program is being executed. The user navigates to the directory `E:\Tugaz Alif\OOP` and runs `javac Bank.java` and `javac BankDemo.java`. Then, they run `java BankDemo`, which produces the following output:

```
E:\Tugaz Alif\OOP>javac Bank.java
E:\Tugaz Alif\OOP>javac BankDemo.java
E:\Tugaz Alif\OOP>java BankDemo
No Rekening          : 123456789
Nama                  : Alif Al Amin
Saldo rekening Anda sejumlah : Rp 150000.0

Anda telah menabung sejumlah Rp 350000.0
Anda telah melakukan penarikan sejumlah Rp 100000.0
Anda telah mentransfer sejumlah Rp 20000.0
No Rekening          : 123456789
Nama                  : Alif Al Amin
Saldo rekening Anda sejumlah : Rp 230000.0

E:\Tugaz Alif\OOP>
```

7. Daftar variable dan fungsi/method yang dimiliki oleh Class String

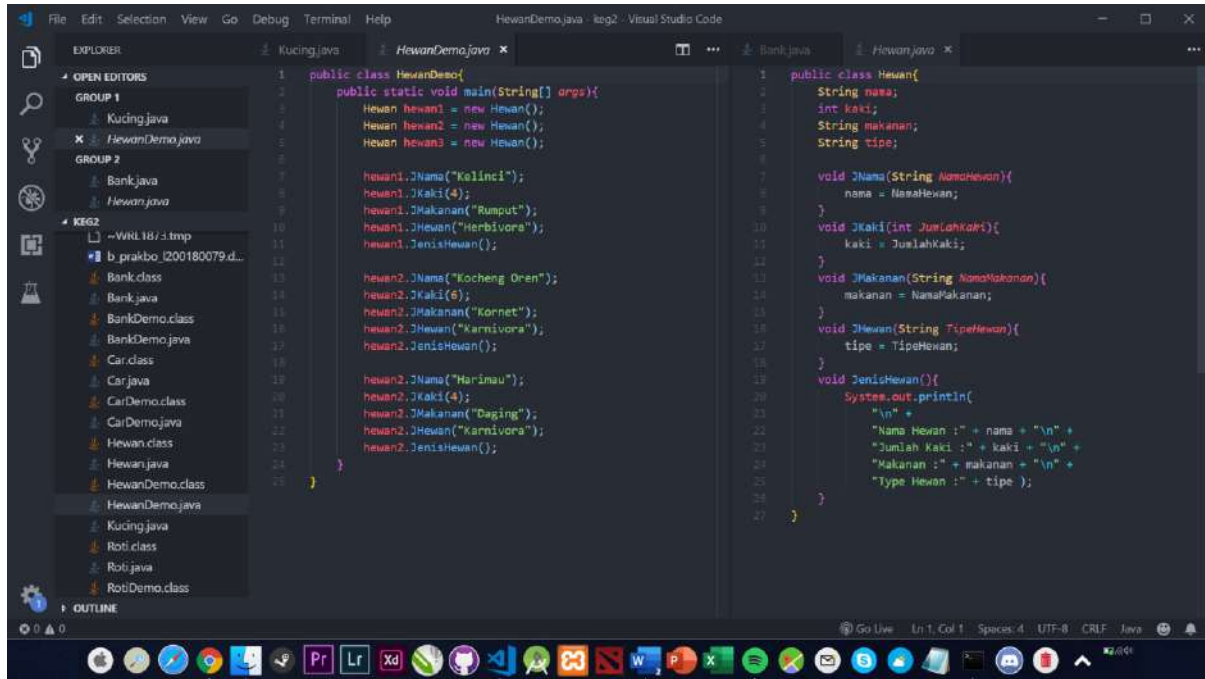
- `String dataString = "Data didalam String"`
- `codePointAt(int)` = mengembalikan code ASCII dari sebuah char yang di ambil dari String yang bersangkutan
- `charAt(int)` = mengembalikan Karakter yang ada di sebuah string bersangkutan sesuai dengan index

yang di masukkan.

d. `codePointBefore(int)` = mengembalikan code ASCII dari karakter yang di ambil dari sebuah String

Tambahan

Pekerjaan Rumah



The screenshot shows the Visual Studio Code editor with two files open: `HewanDemo.java` and `Hewan.java`. The `EXPLORER` sidebar on the left shows a project structure with various Java files. The `OUTPUT` window at the bottom shows the execution results of the program.

```
public class HewanDemo {
    public static void main(String[] args) {
        Hewan hewan1 = new Hewan();
        Hewan hewan2 = new Hewan();
        Hewan hewan3 = new Hewan();

        hewan1.setName("Kelinci");
        hewan1.setKaki(4);
        hewan1.setMakanan("Rumput");
        hewan1.setJenisHewan();

        hewan2.setName("Kucing Oren");
        hewan2.setKaki(4);
        hewan2.setMakanan("Kornet");
        hewan2.setJenisHewan();

        hewan3.setName("Harimau");
        hewan3.setKaki(4);
        hewan3.setMakanan("Daging");
        hewan3.setJenisHewan();
    }
}
```

```
public class Hewan {
    String nama;
    int kaki;
    String makanan;
    String tipe;

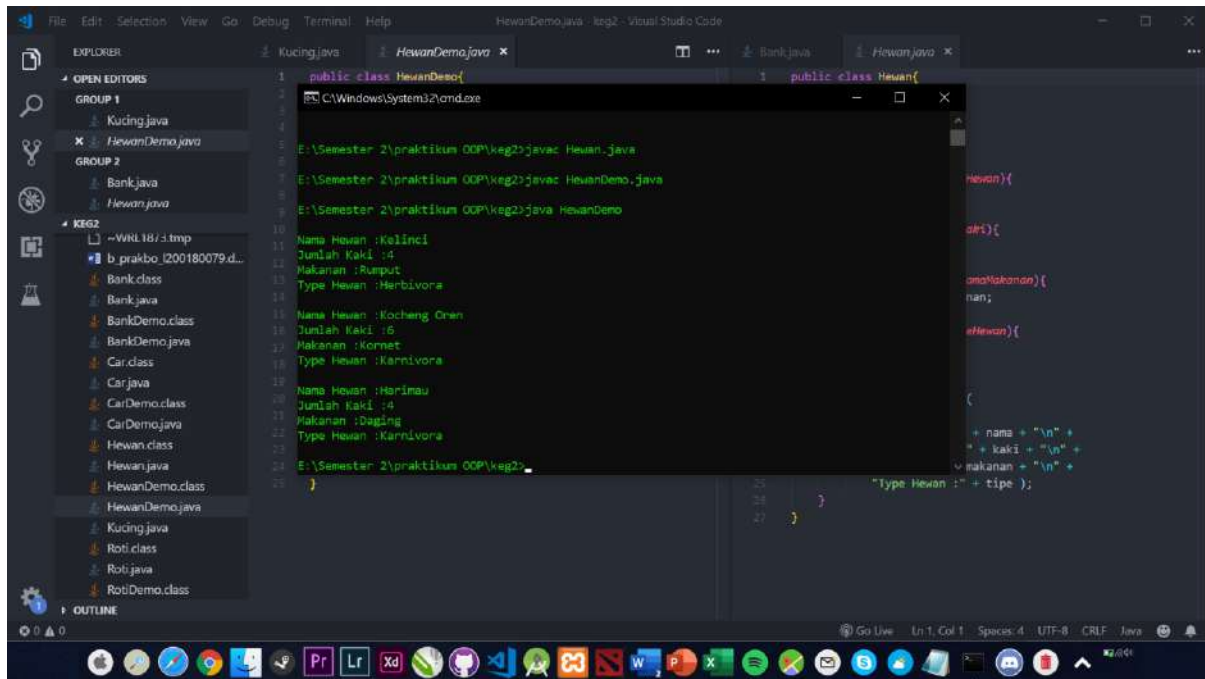
    void setName(String NamaHewan) {
        nama = NamaHewan;
    }

    void setKaki(int JumlahKaki) {
        kaki = JumlahKaki;
    }

    void setMakanan(String NamaMakanan) {
        makanan = NamaMakanan;
    }

    void setJenisHewan(String TipeHewan) {
        tipe = TipeHewan;
    }

    void JenisHewan() {
        System.out.println(
            "\n" +
            "Nama Hewan : " + nama + "\n" +
            "Jumlah Kaki : " + kaki + "\n" +
            "Makanan : " + makanan + "\n" +
            "Type Hewan : " + tipe );
    }
}
```



The screenshot shows the Visual Studio Code editor with the `OUTPUT` window open, displaying the execution results of the program. The output shows the details for three animals: a rabbit, a cat, and a tiger.

```

Nama Hewan : Kelinci
Jumlah Kaki : 4
Makanan : Rumput
Type Hewan : Herbivora

Nama Hewan : Kucing Oren
Jumlah Kaki : 4
Makanan : Kornet
Type Hewan : Karnivora

Nama Hewan : Harimau
Jumlah Kaki : 4
Makanan : Daging
Type Hewan : Karnivora
```

This screenshot shows the Visual Studio Code editor with two Java files open. The left pane shows the Explorer view with a file tree containing various Java files. The main editor area displays the code for `Dosen.java` and `Mahasiswa.java`.

```
public class Dosen{
    String nama;
    int nik;
    String pendidikan;
    Date tgllahir;

    void tampilkanNama(String nome){
        nama = nome;
    }

    void tampilkanNik(int nik){
        this.nik = nik;
    }

    void tampilkanTglahir(Date tgl){
        tgllahir = tgl;
    }
}

public class Mahasiswa{
    String nama;
    String alamat;
    int nim;
    int semester;

    void tampilkanNama(String nome){
        nama = nome;
    }

    void tampilkanAlamat(String almt){
        alamat = almt;
    }

    void tampilkanNim(int Nim){
        nim = Nim;
    }

    void tampilkanSemester(int smtr){
        semester = smtr;
    }
}
```

This screenshot shows the Visual Studio Code editor with the `Karyawan.java` file open. The Explorer view on the left shows the file tree. The main editor area displays the code for `Karyawan.java`.

```
public class Karyawan{
    String nama;
    String alamat;
    String jabatan;
    int gaji;

    void tampilkanNama(String nome){
        nama = nome;
    }

    void tampilkanAlamat(String almt){
        alamat = almt;
    }

    void tampilkanJabatan(String jbtn){
        jabatan = jbtn;
    }

    void tampilkanGaji(int gji){
        gaji = gji;
    }
}
```

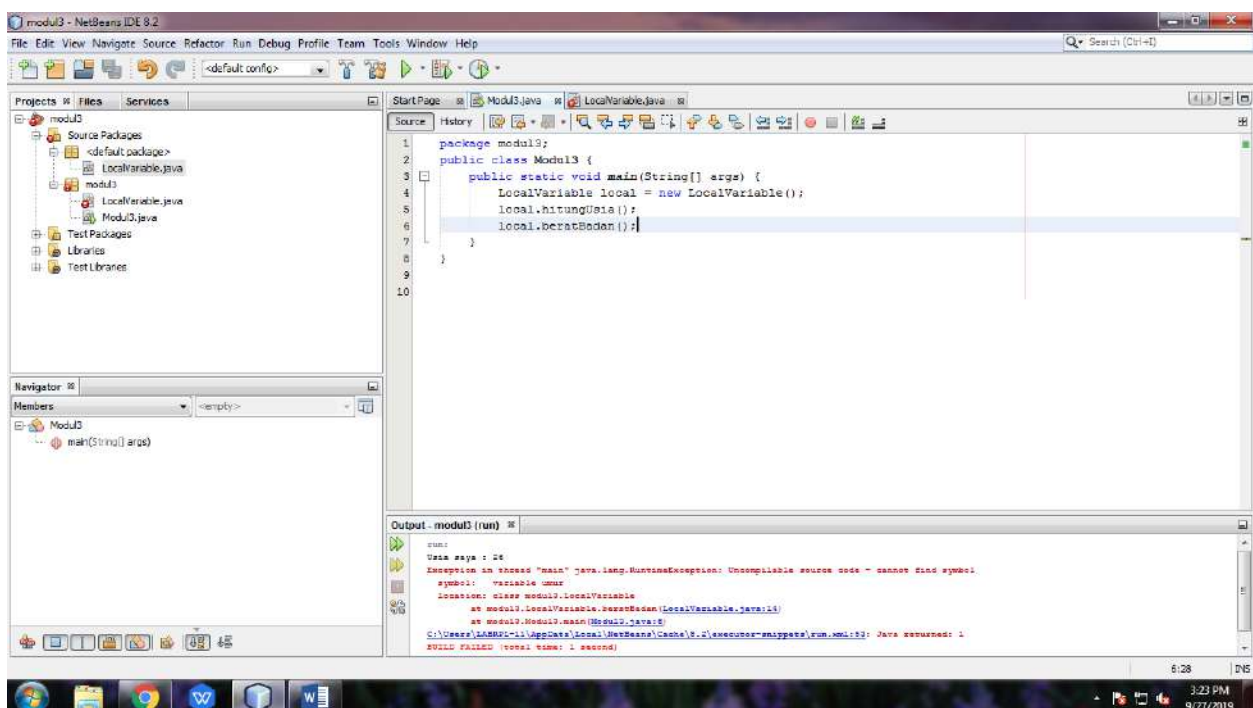
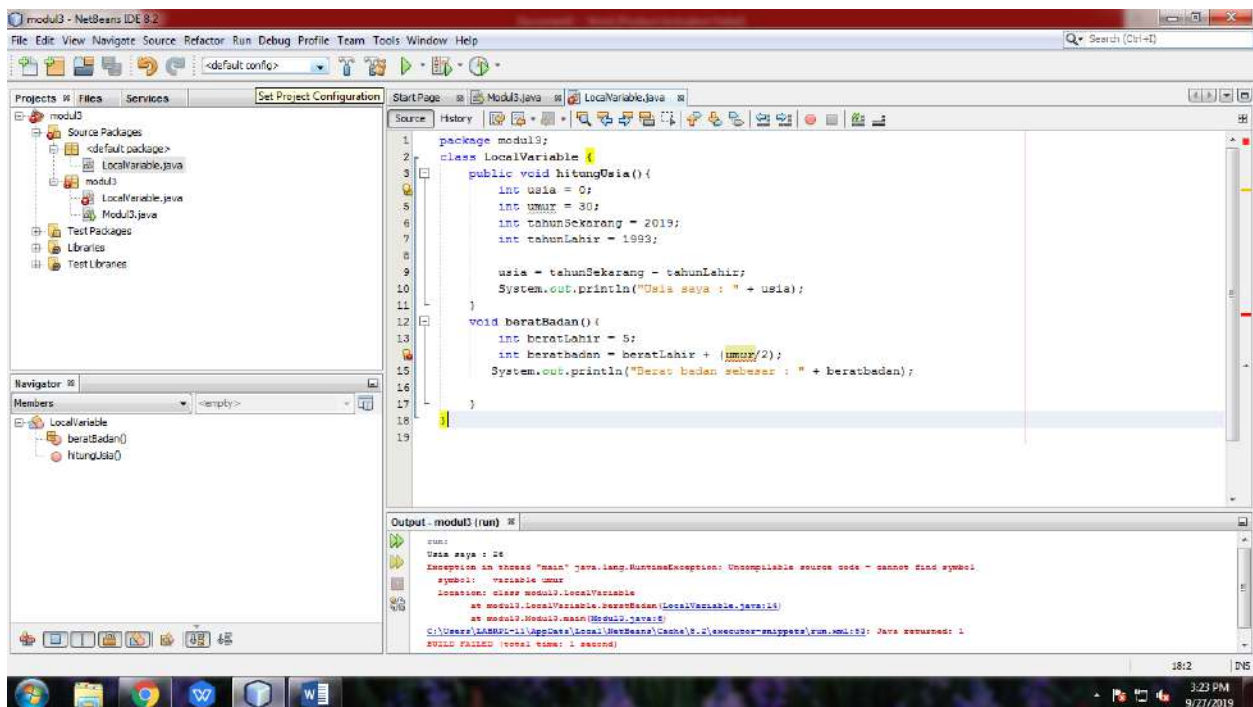

Nama : Alif Al Amin

NIM : L200180082

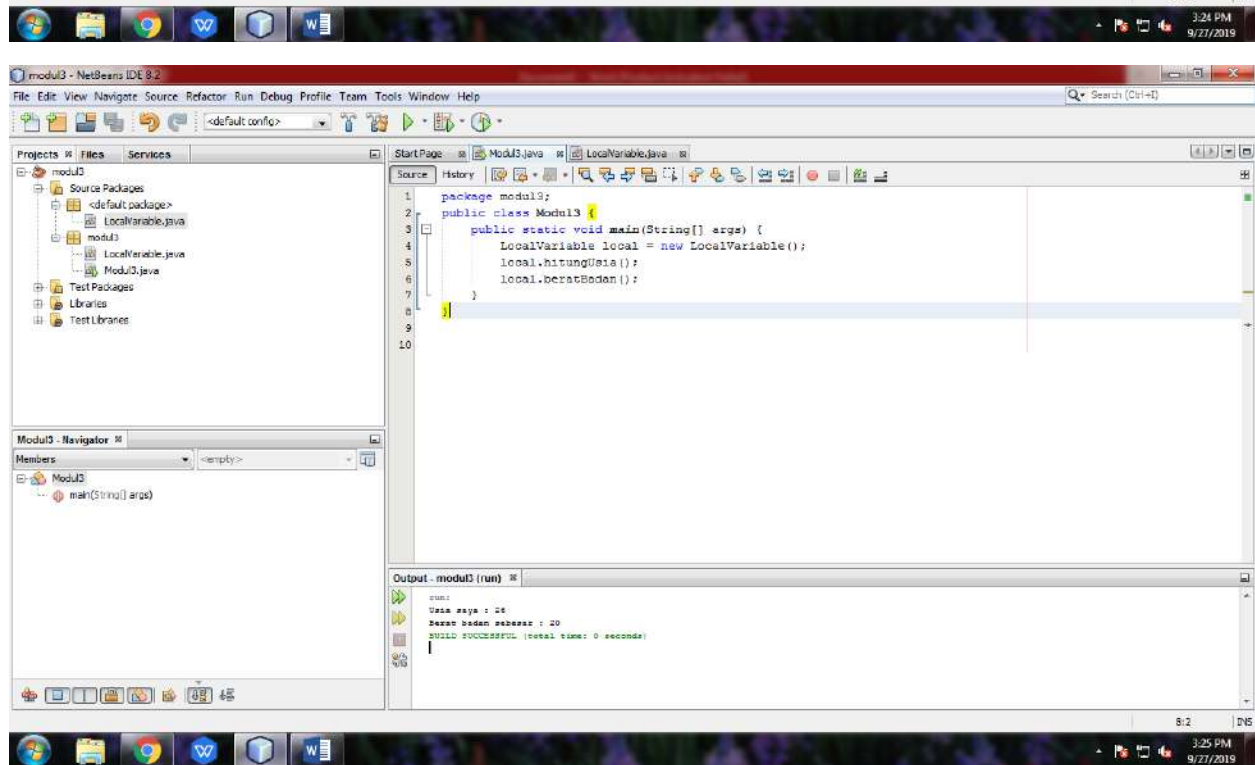
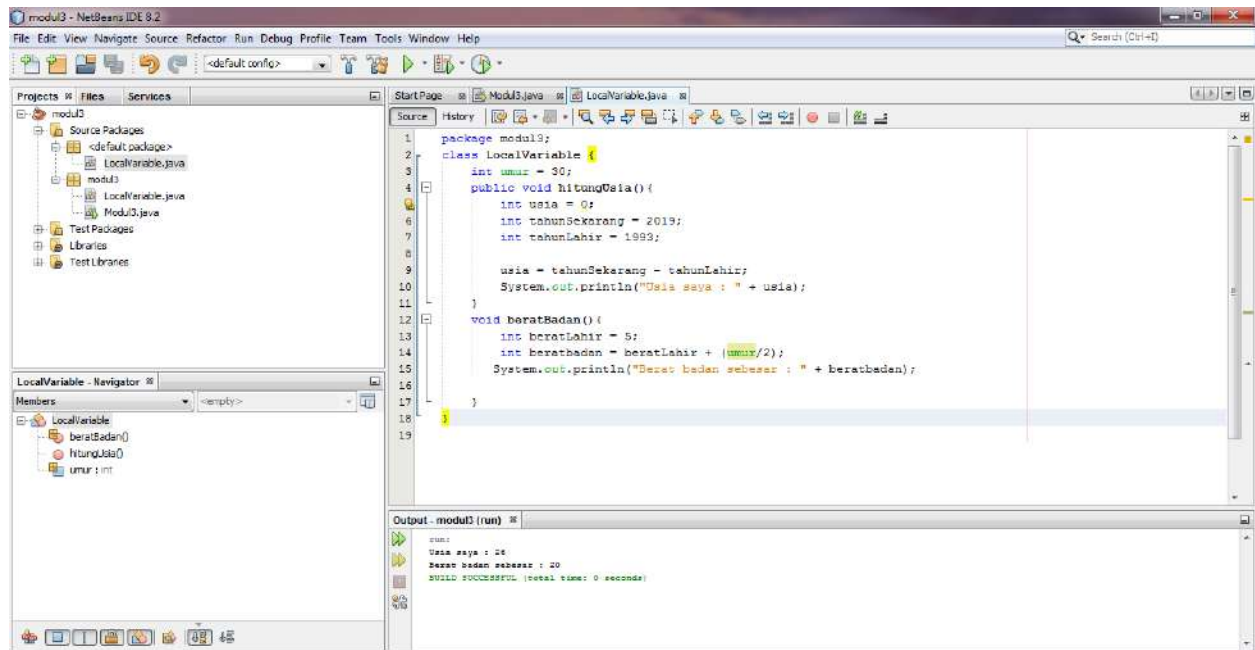
Kelas : B

Laporan Modul 3

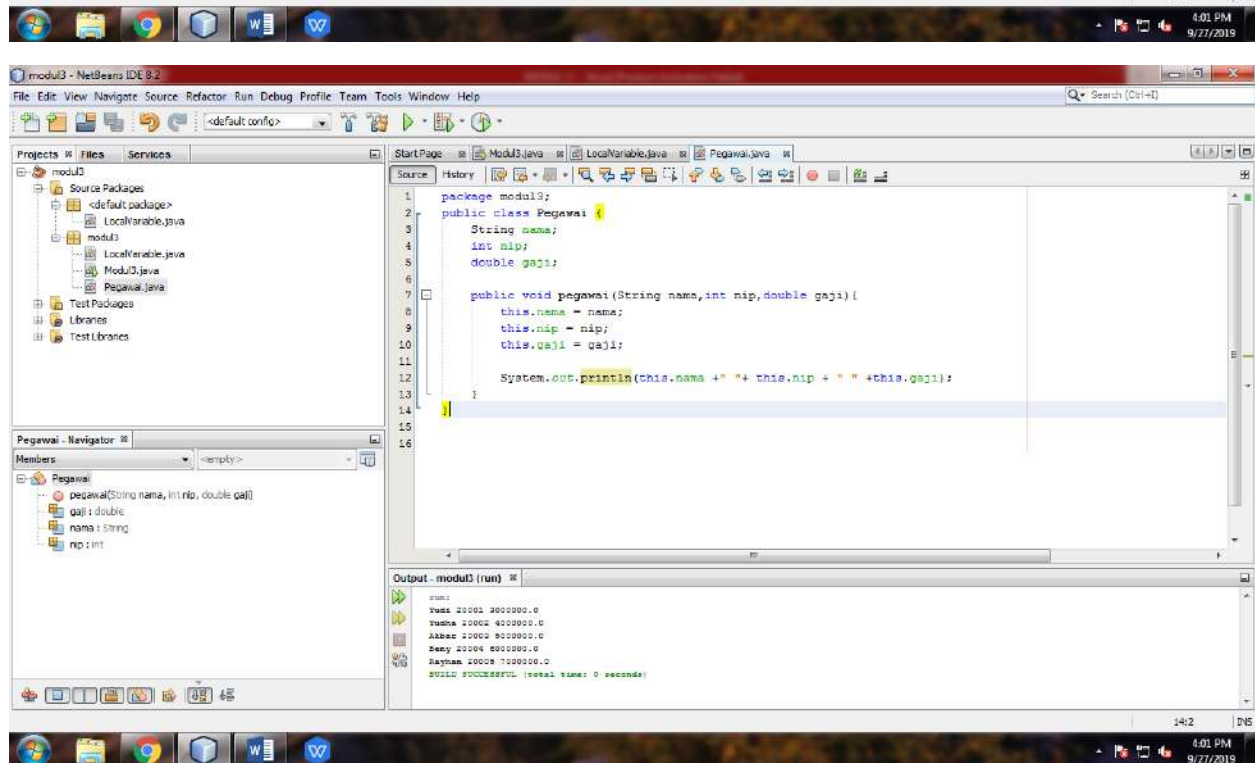
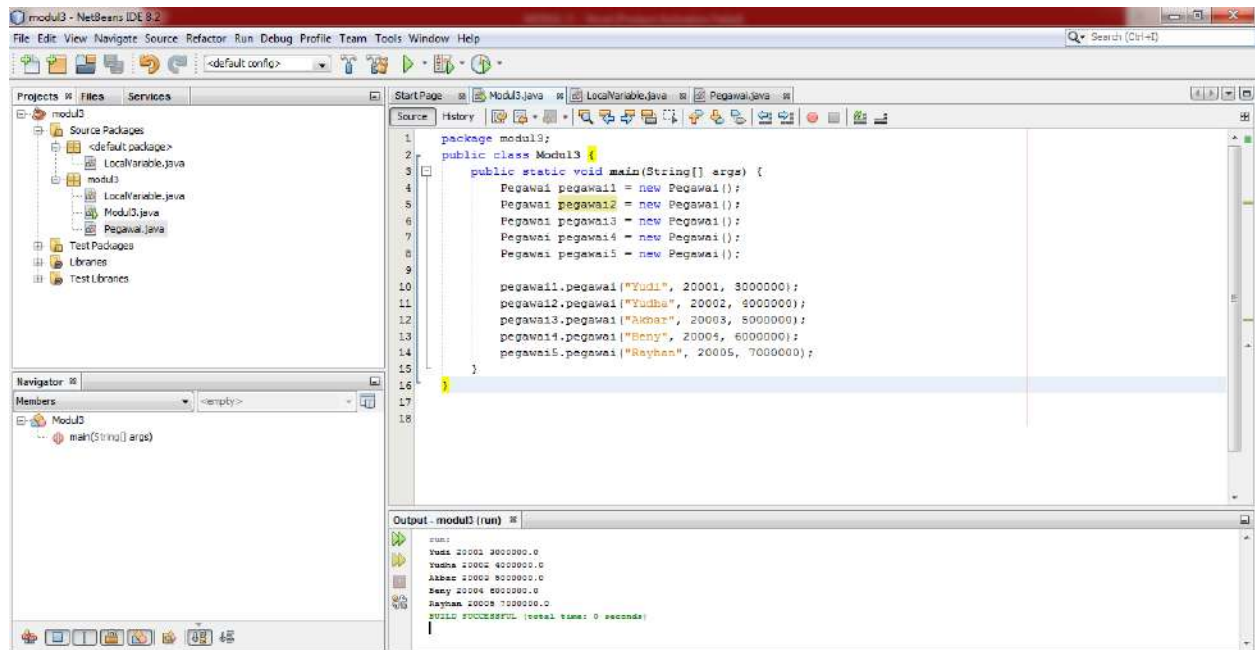
1. Latihan 1



2. Latihan 2

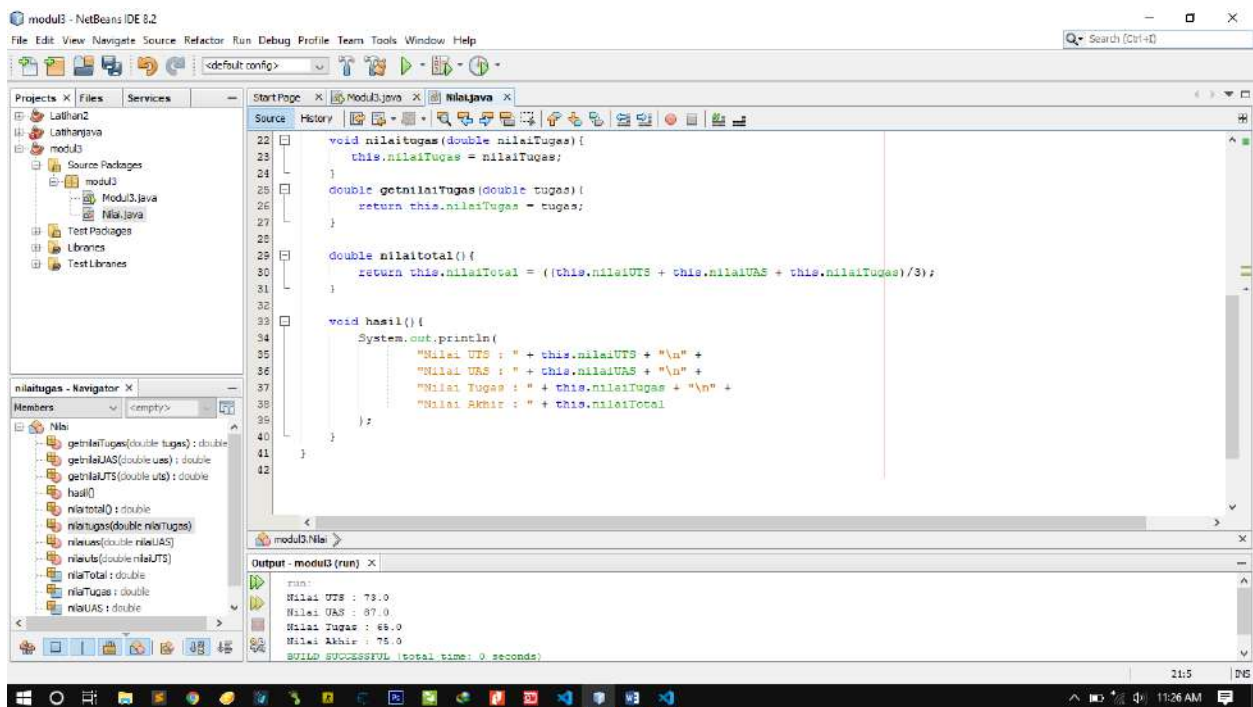
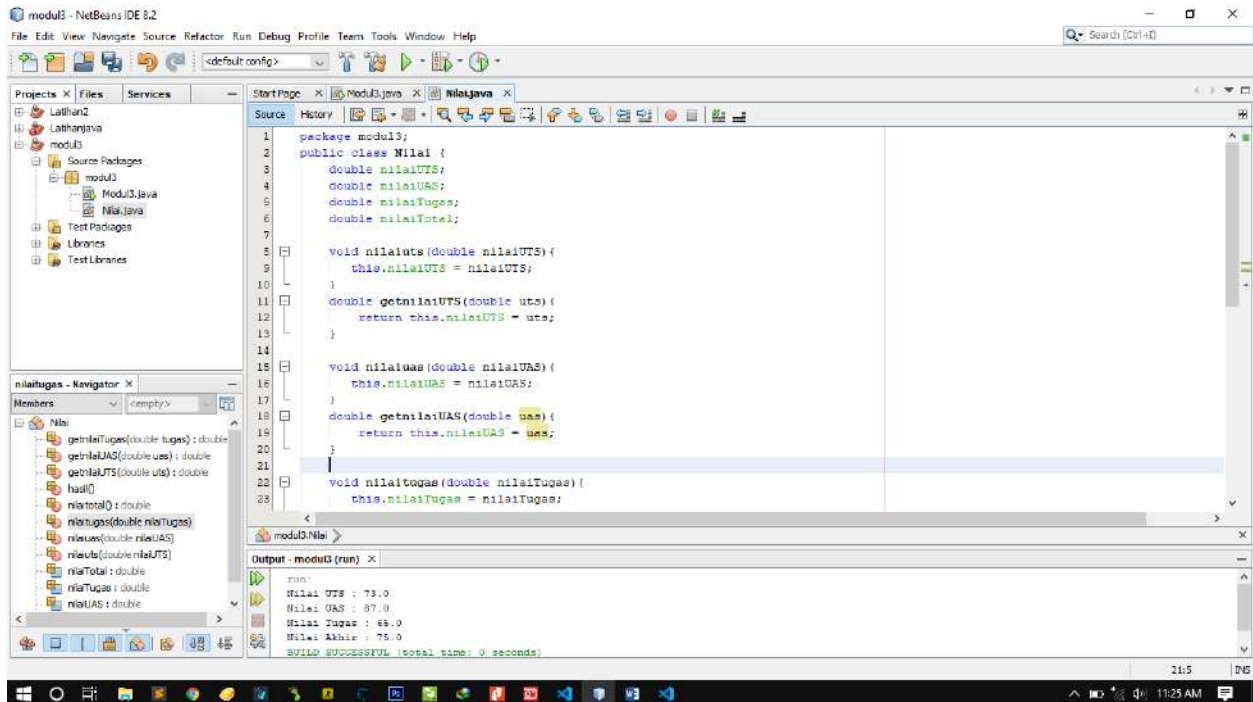


3. Latihan 3

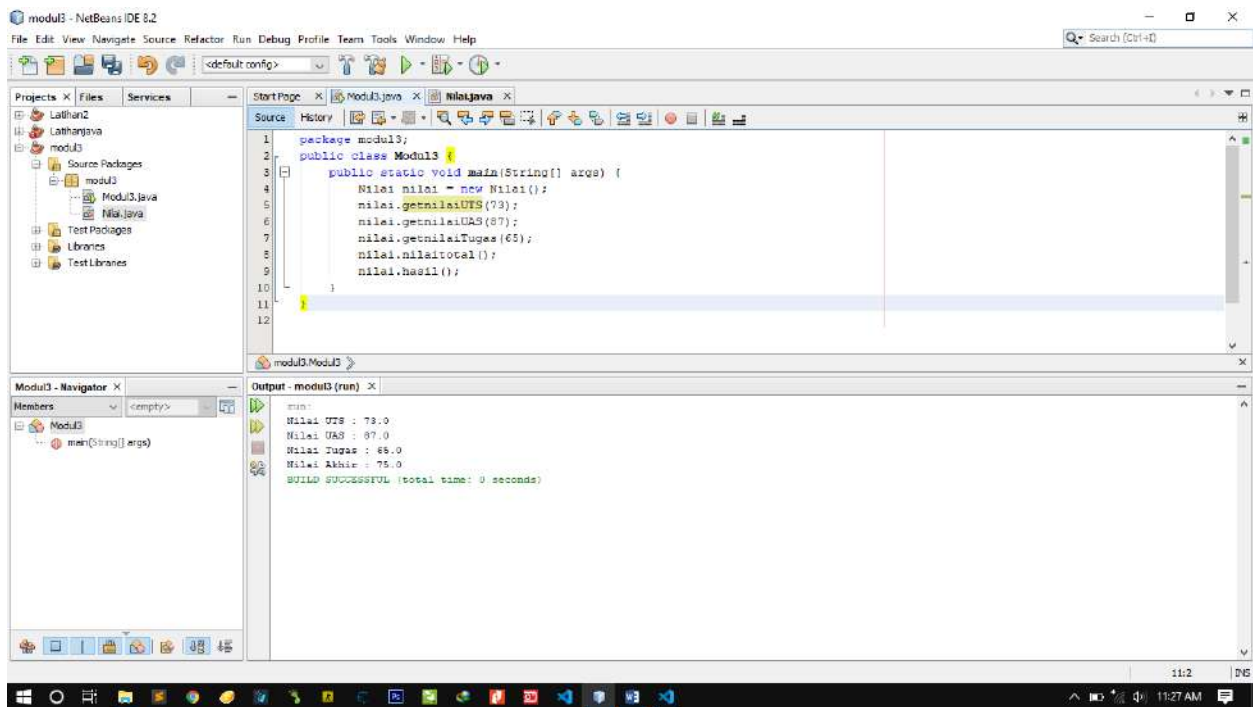


4. Pekerjaan Rumah

a. Nilai.java



b. Modul3.java + output



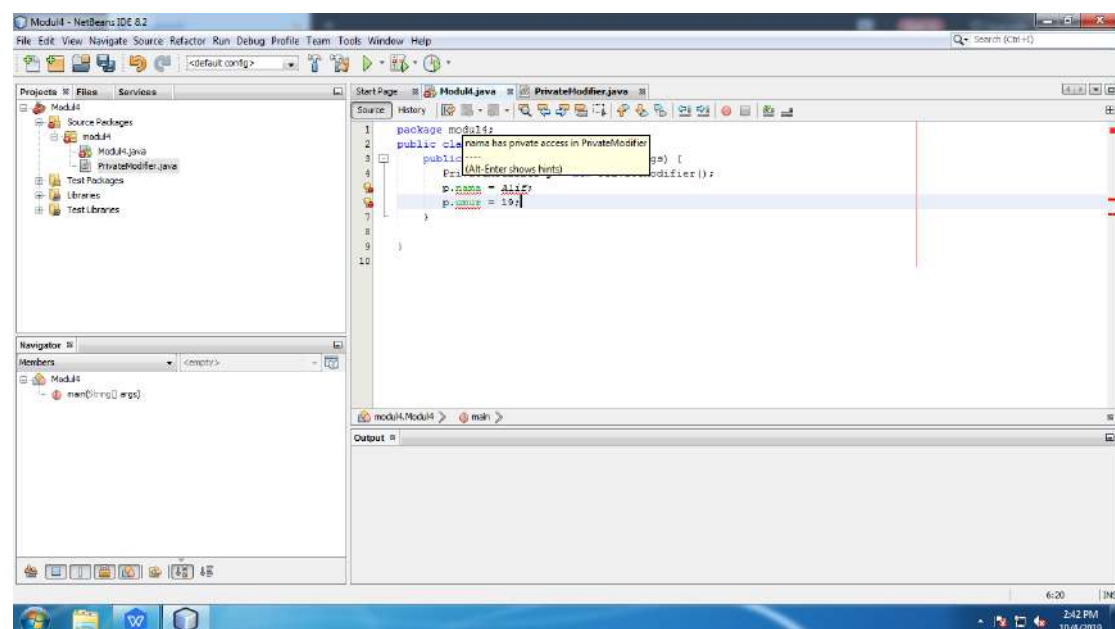
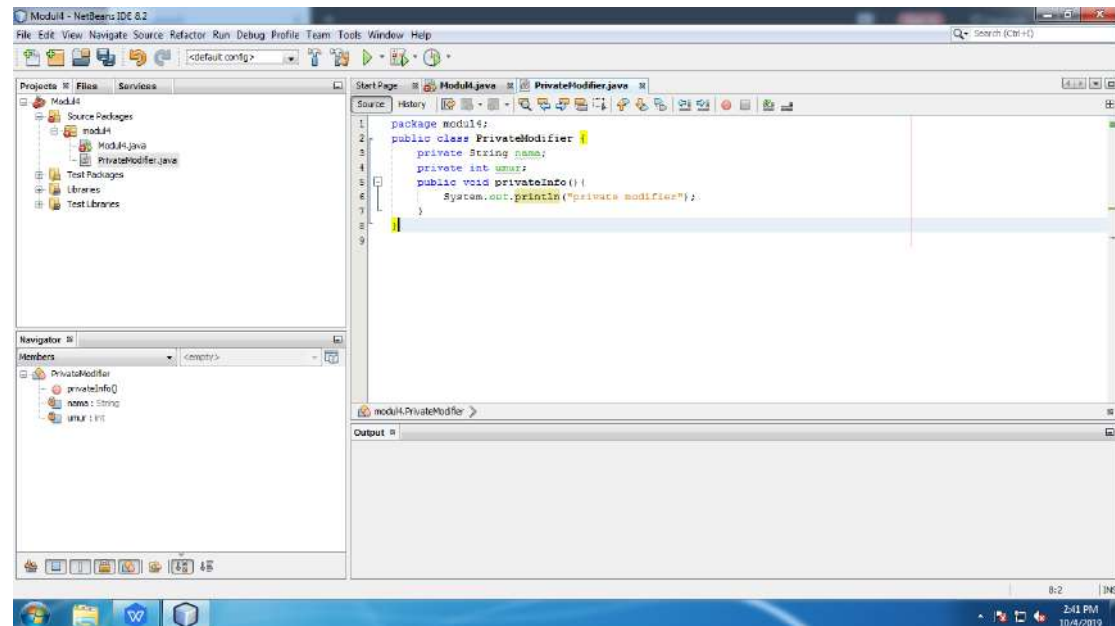
Nama : Alif Al Amin

NIM : L200180082

Kelas : B

Laporan Modul ke-4

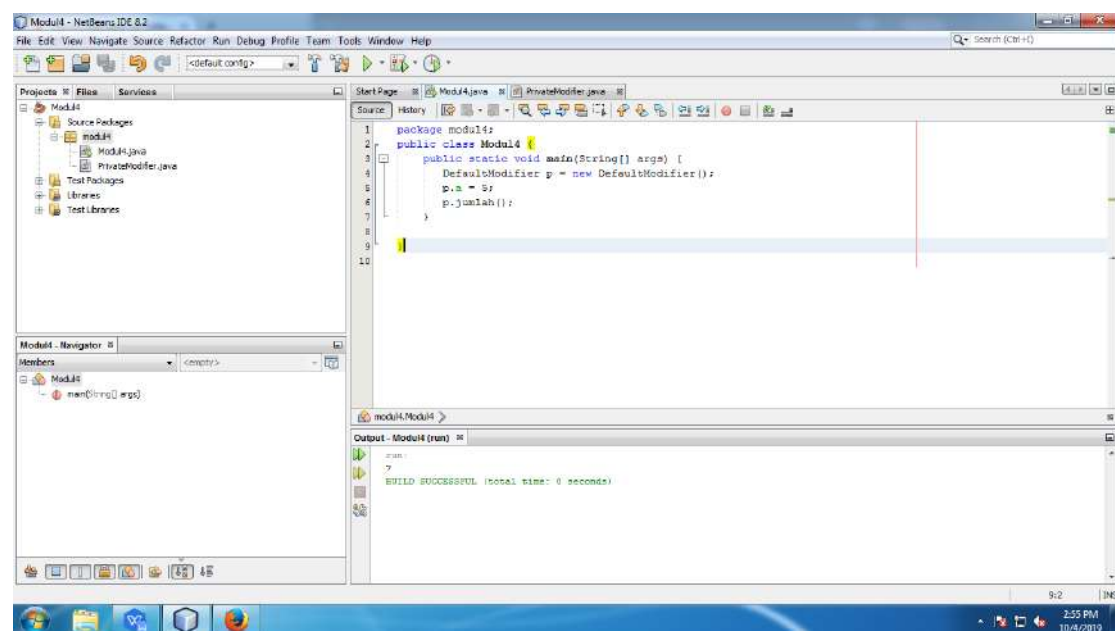
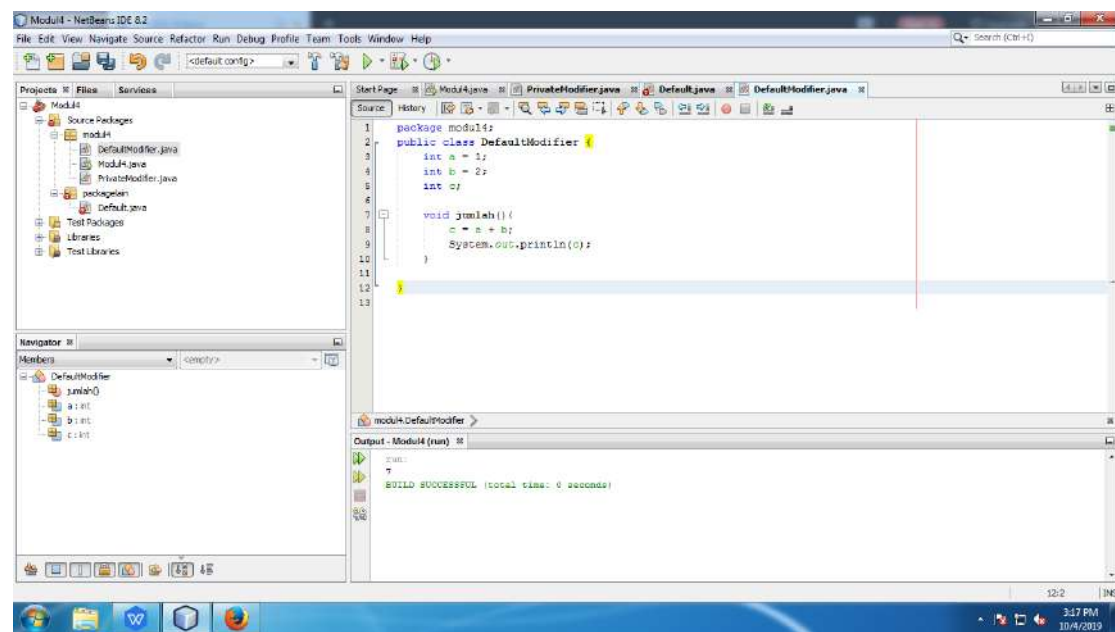
1. Latihan 4.1.1

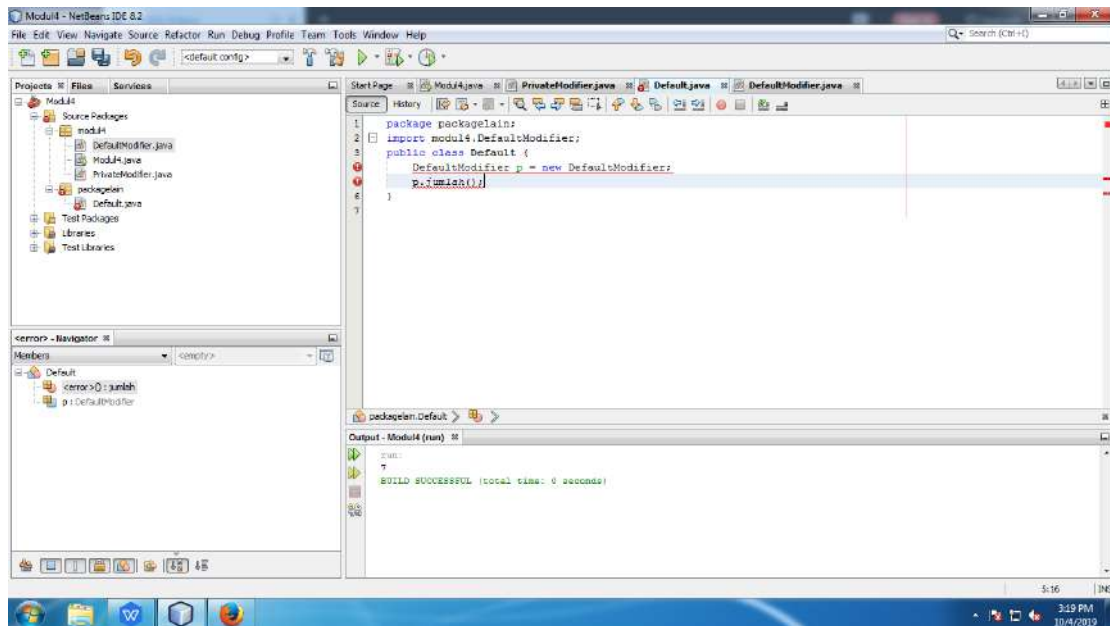


Penjelasan:

Variable `nama` dan `umur` tidak bisa diakses di class lain (modul 4), karena variable tersebut memiliki private modifier, sehingga hanya bisa diakses di classnya sendiri.

2. Latihan 4.2.1

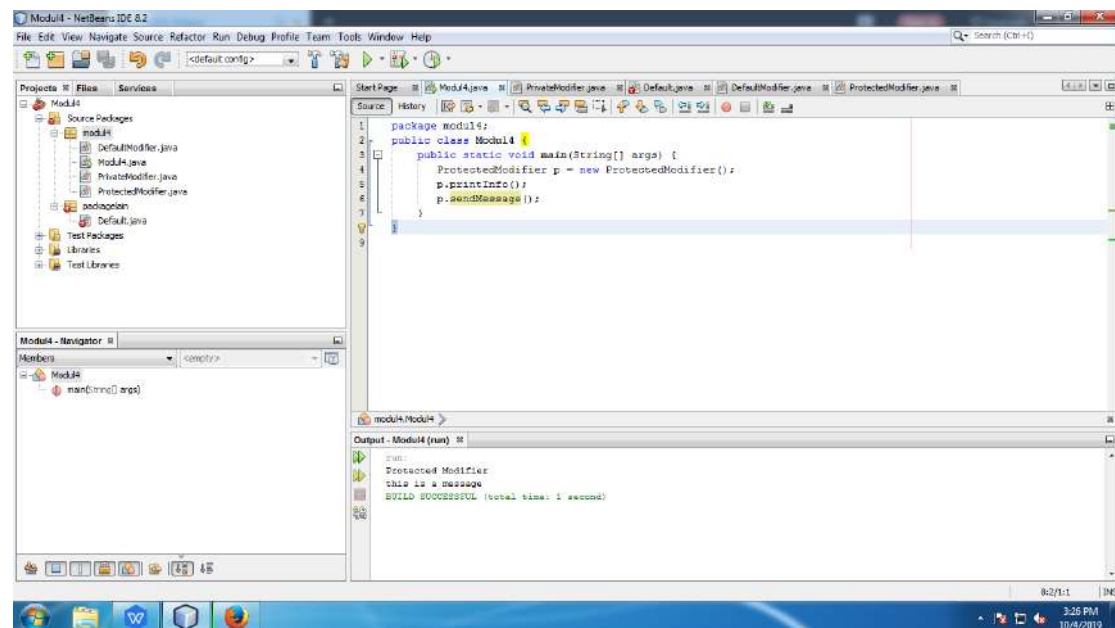
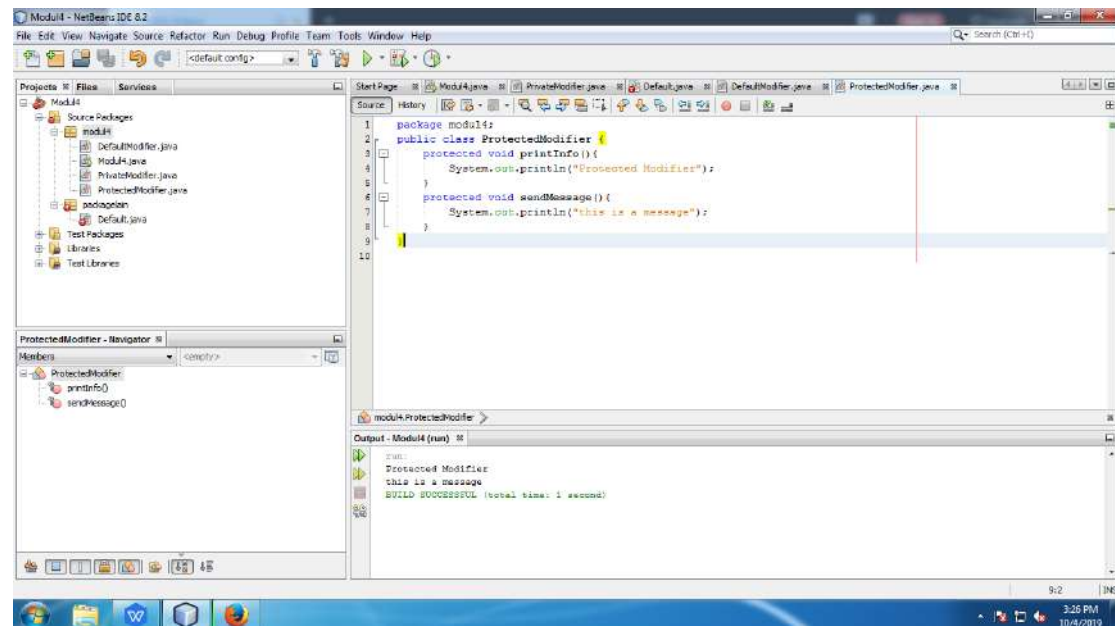




Penjelasan:

Berdasarkan output program diatas, variable dan method di dalam class Default modifier bisa diakses di class lain dan pada package yang sama. Namun, variable dan method tersebut tidak dapat diakses di class lain yang berada di package yang berbeda, walaupun sudah di import.

3. Latihan 4.3.1



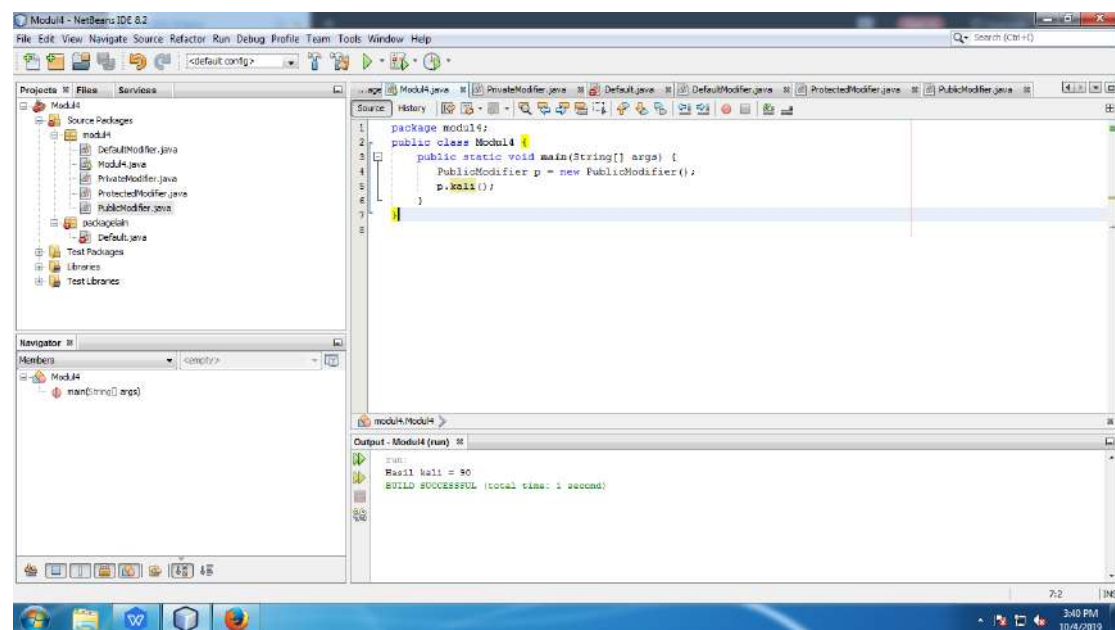
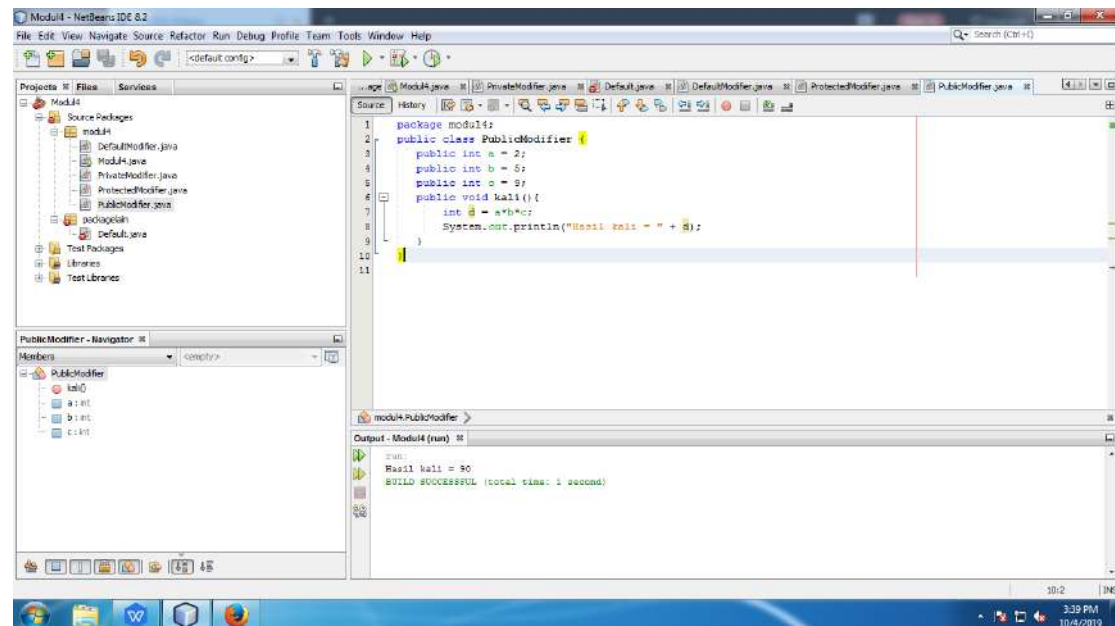
Penjelasan:

Perbandingan

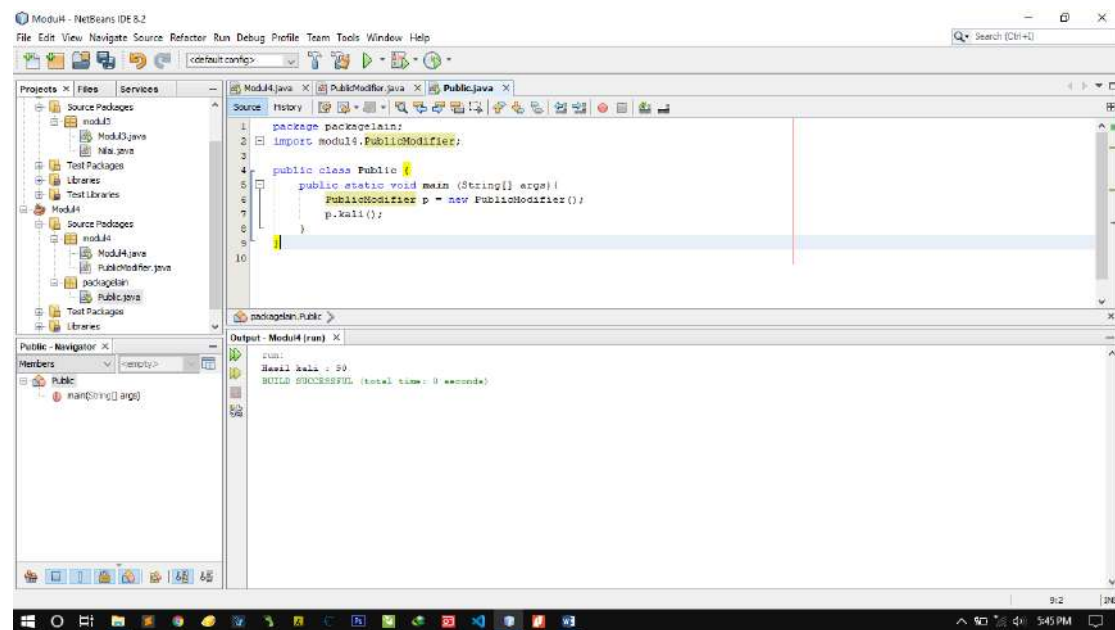
Akses	Private	Default	Protected
Class sama	Bisa	Bisa	Bisa
Class lain, package sama	Tidak bisa	Bisa	Bisa
Class lain, beda package	Tidak bisa	Tidak bisa	Tidak bisa

4. Latihan 4.4.1

1. -Package sama



- package lain

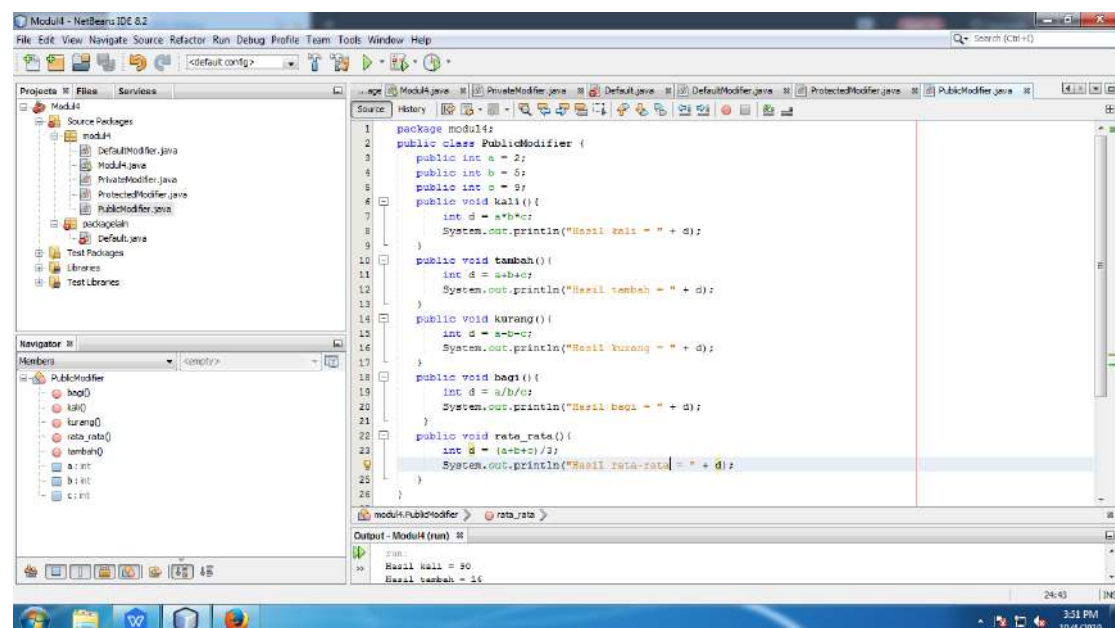


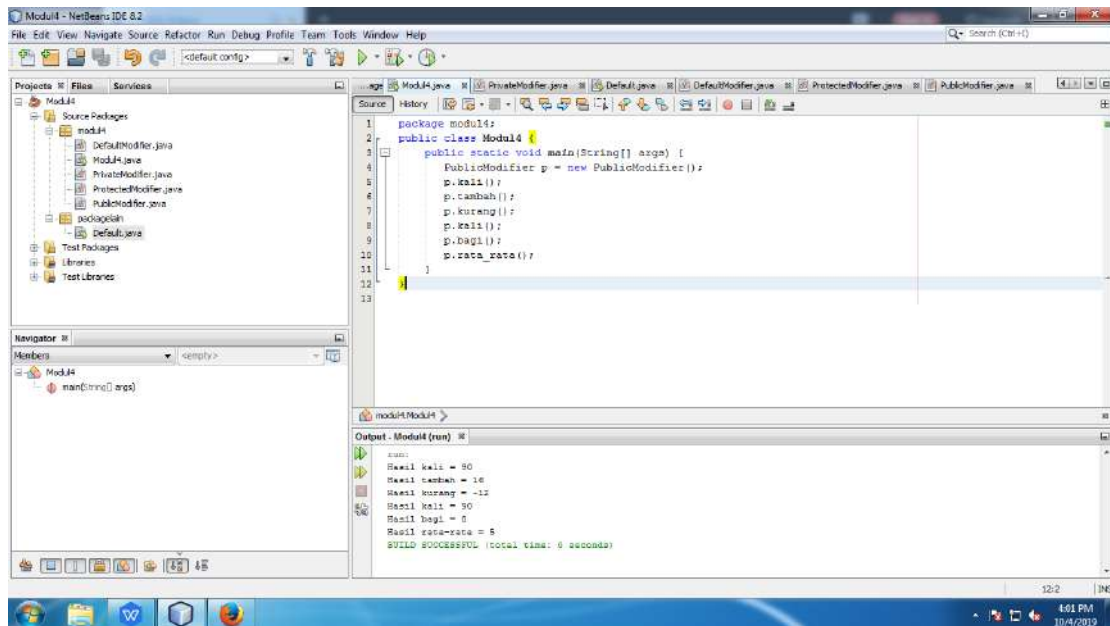
Penjelasan:

Berdasarkan output program diatas, variable dan method pada public modifier bisa diakses di class sendiri, class lain di package yang sama, dan class lain di package yang berbeda.

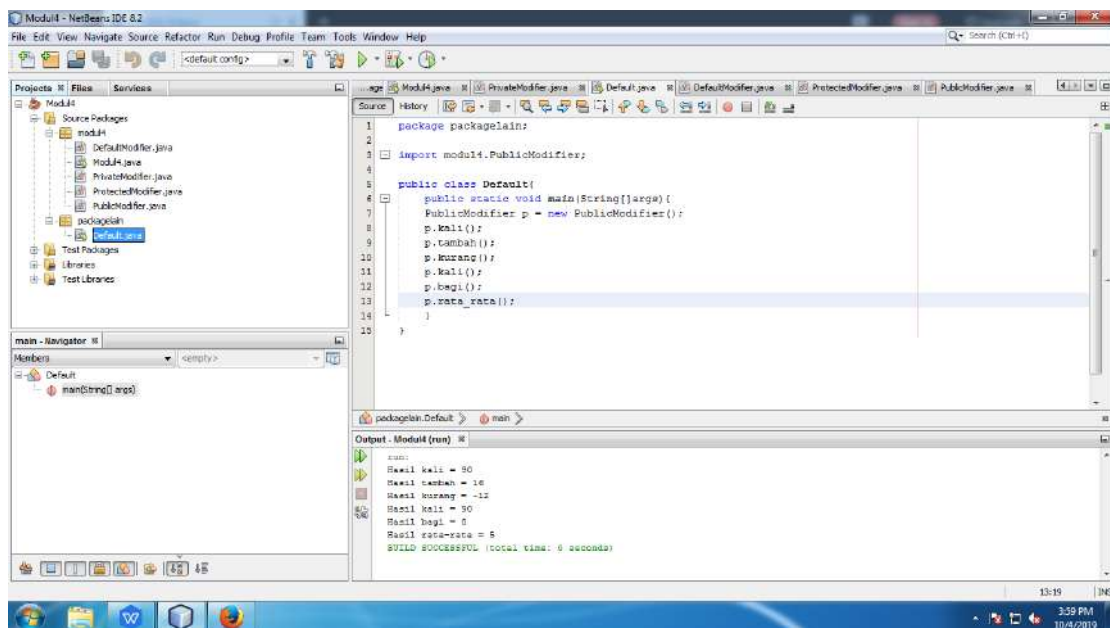
2. Method tambahan

-Package sama





-package lain



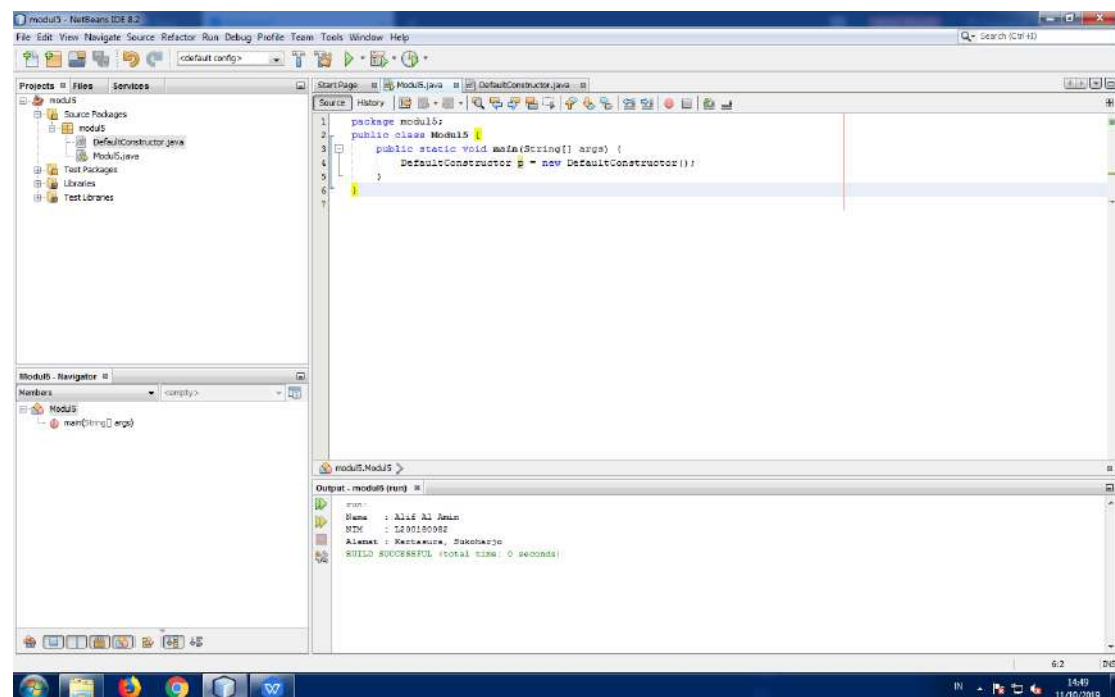
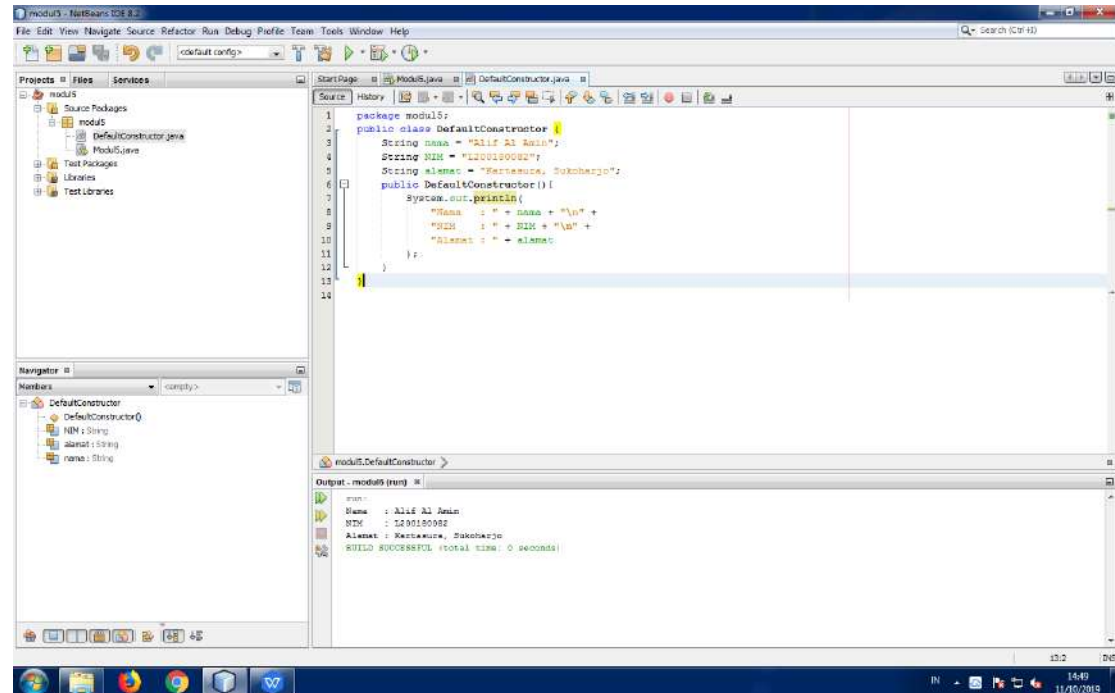
Nama : Alif Al Amin

NIM : L200180082

Kelas : B

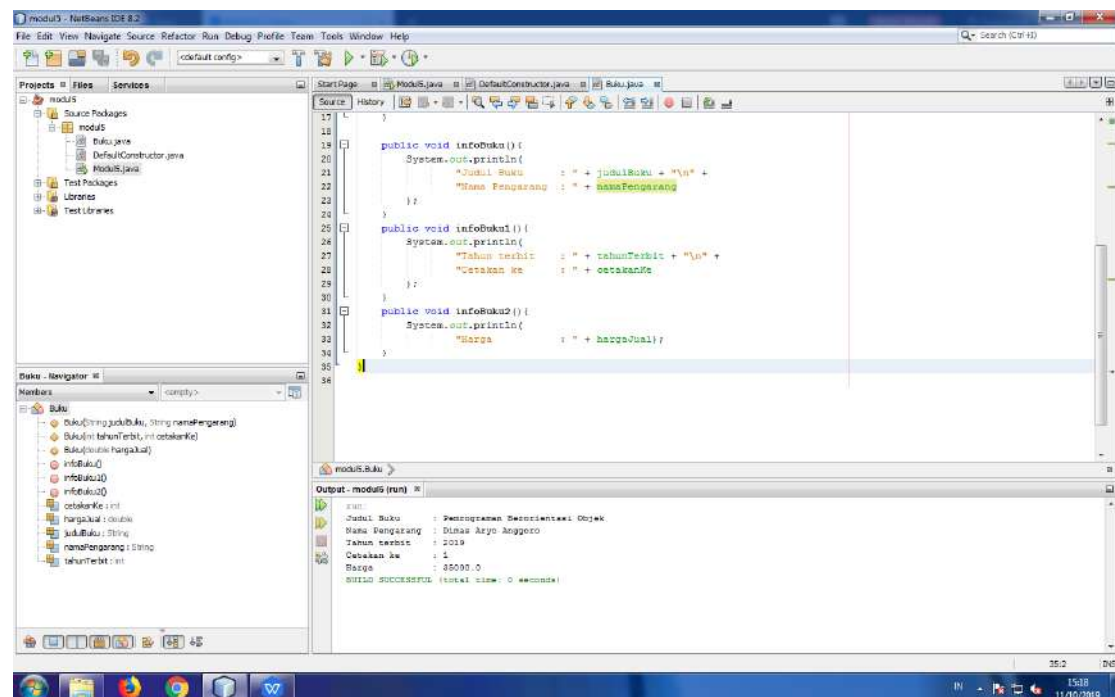
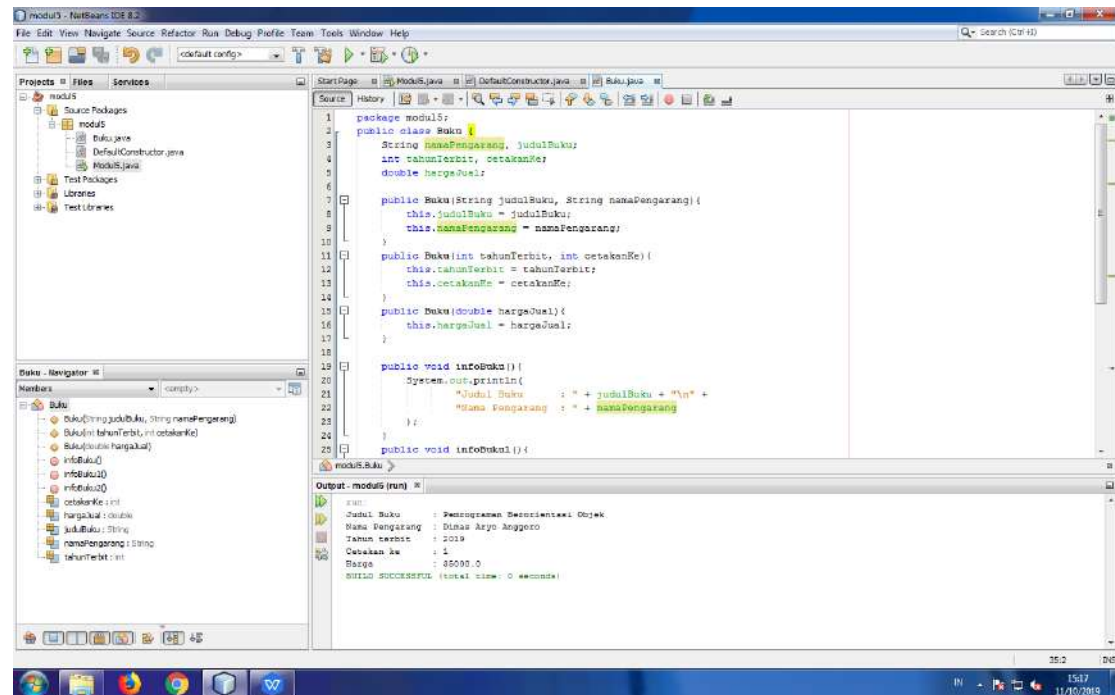
Laporan Modul ke-5

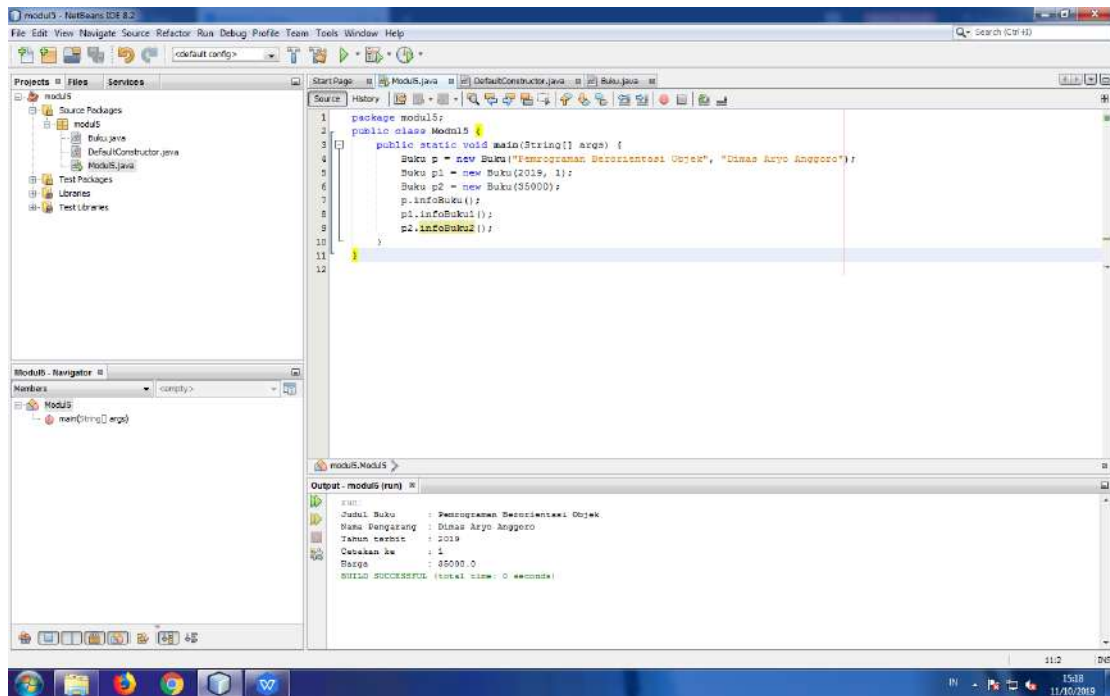
1. Latihan 1



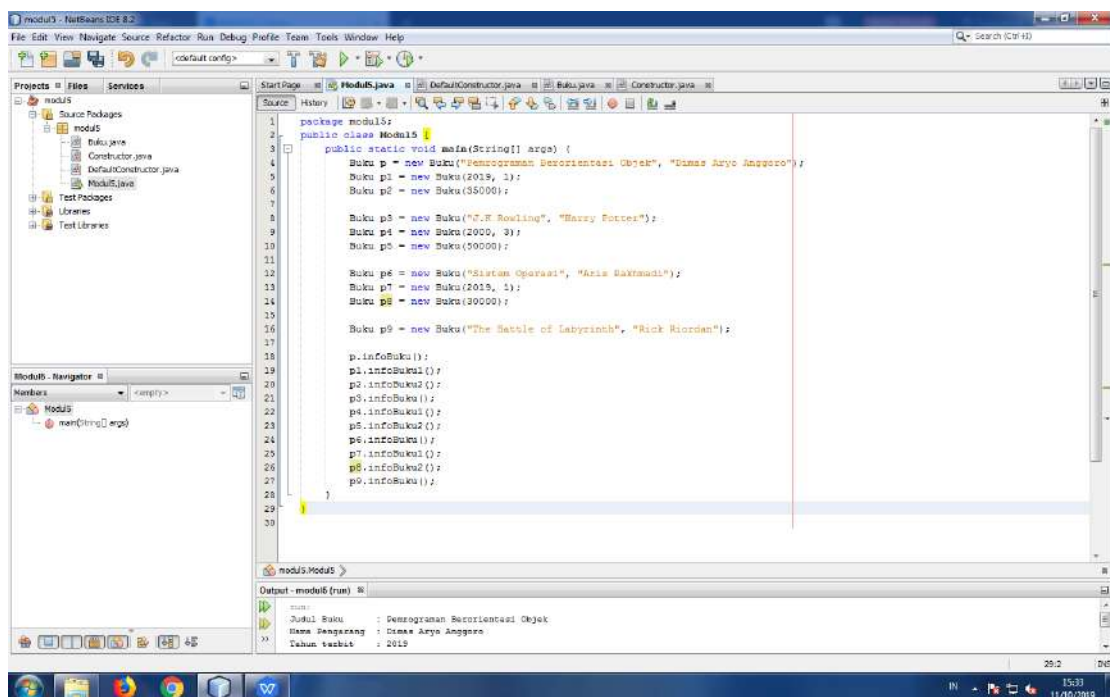
2. Latihan 2

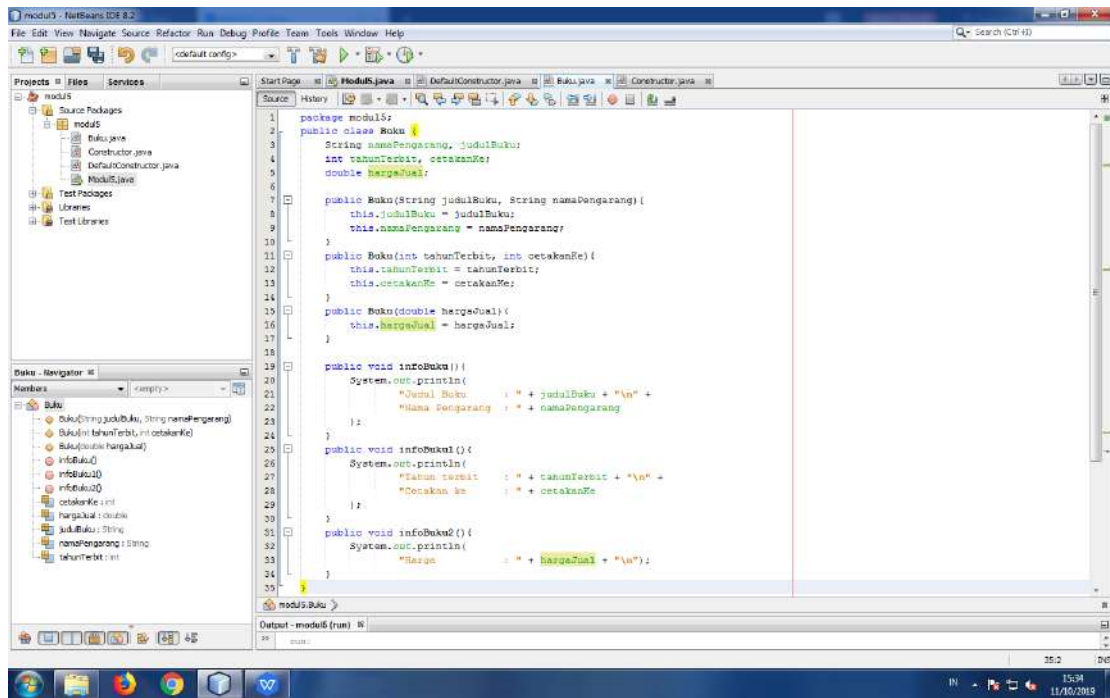
1. Membuat class Buku dan membuat 3 constructor



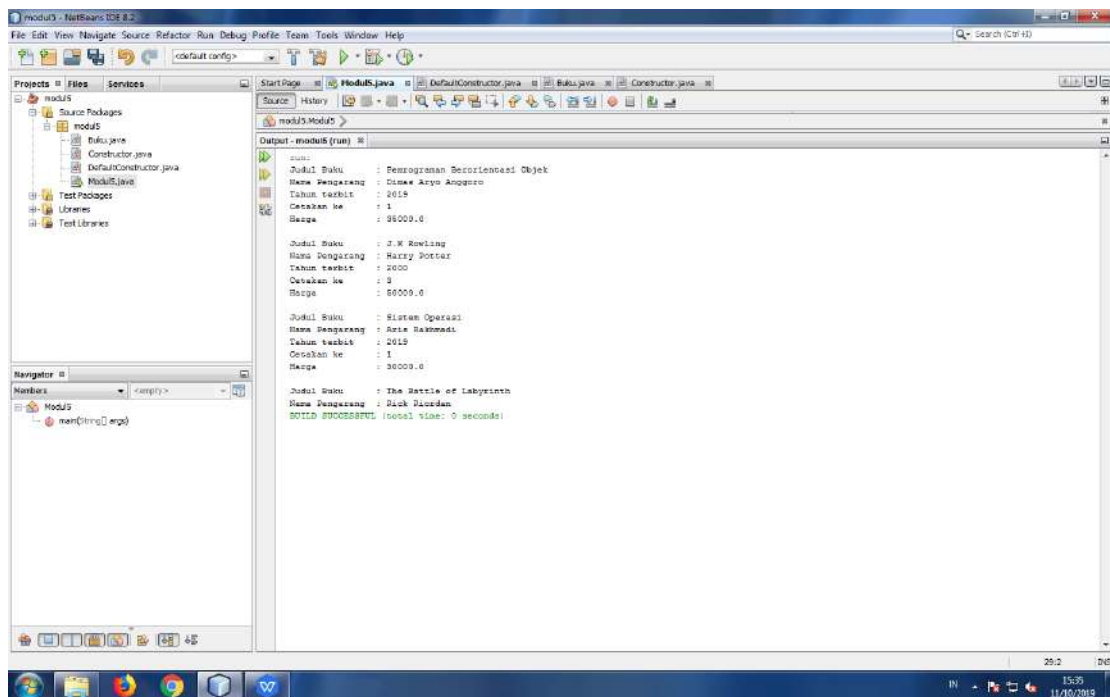


2. Membuat 10 object dengan fungsi main()



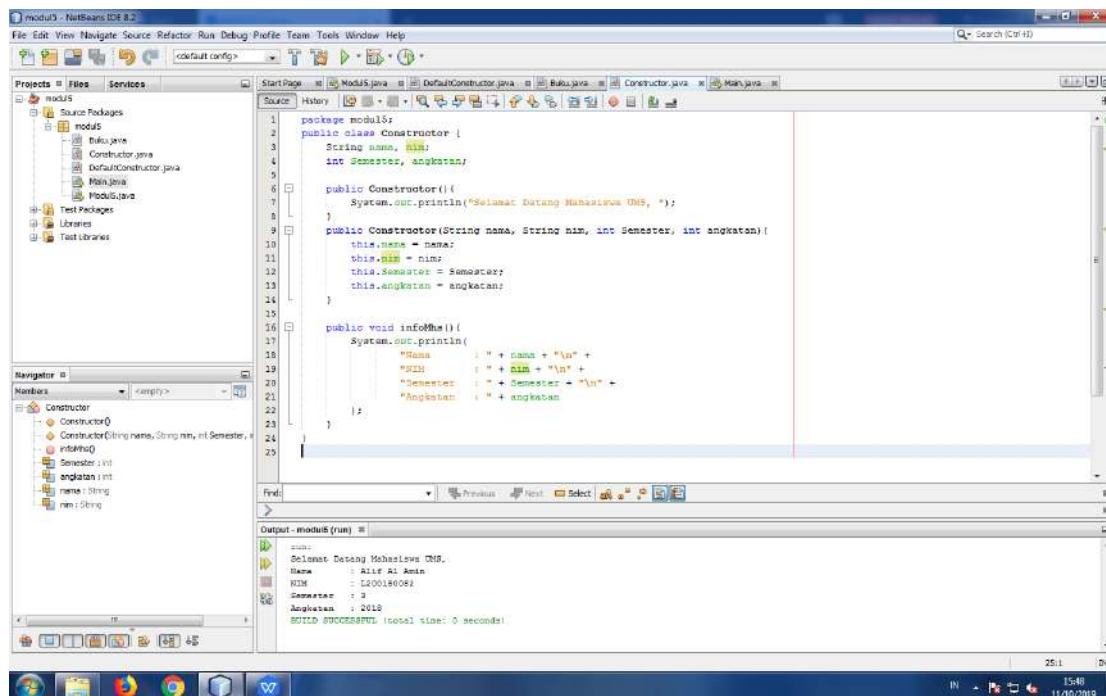


-Output

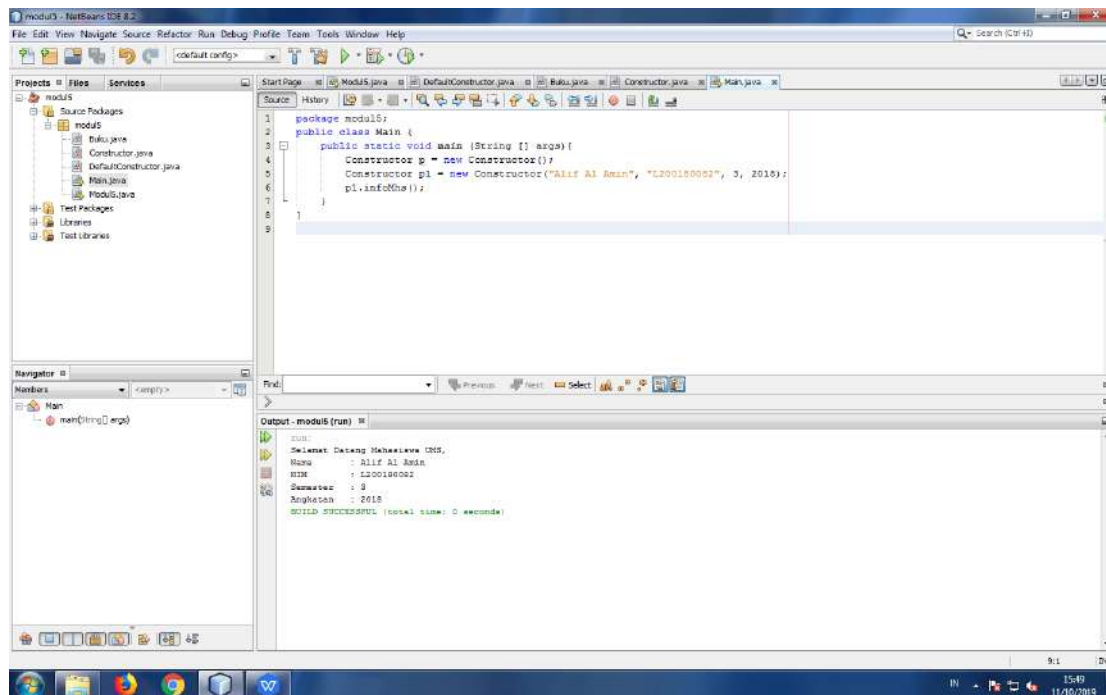


3. Tugas

1. Membuat class yang didalamnya terdapat default constructor dan parameterized constructor



2. Menambahkan class baru berisi method main()



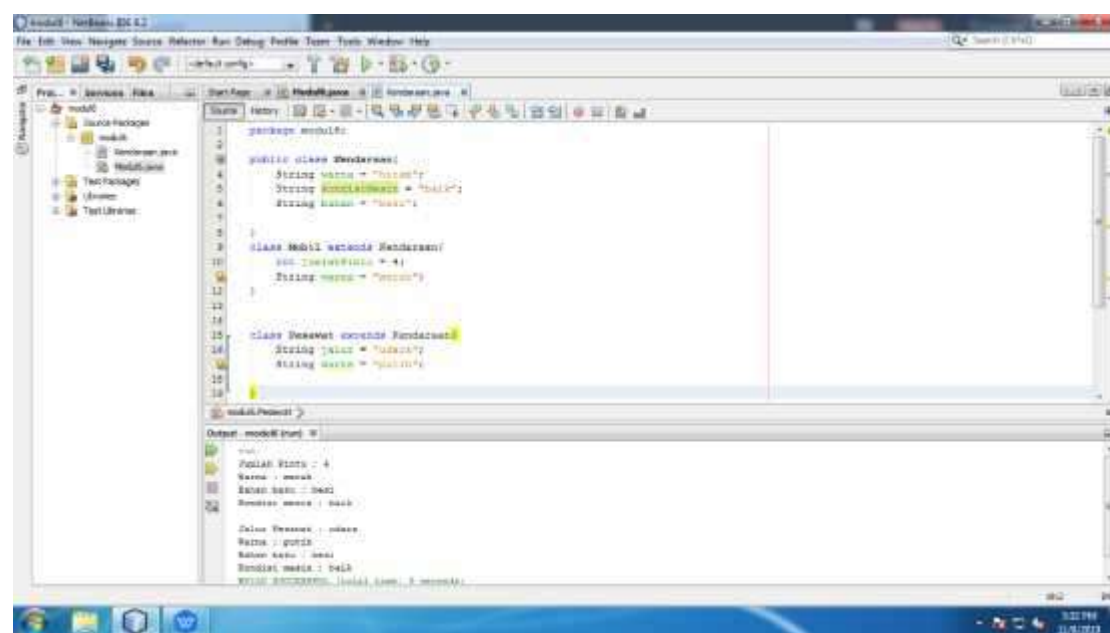
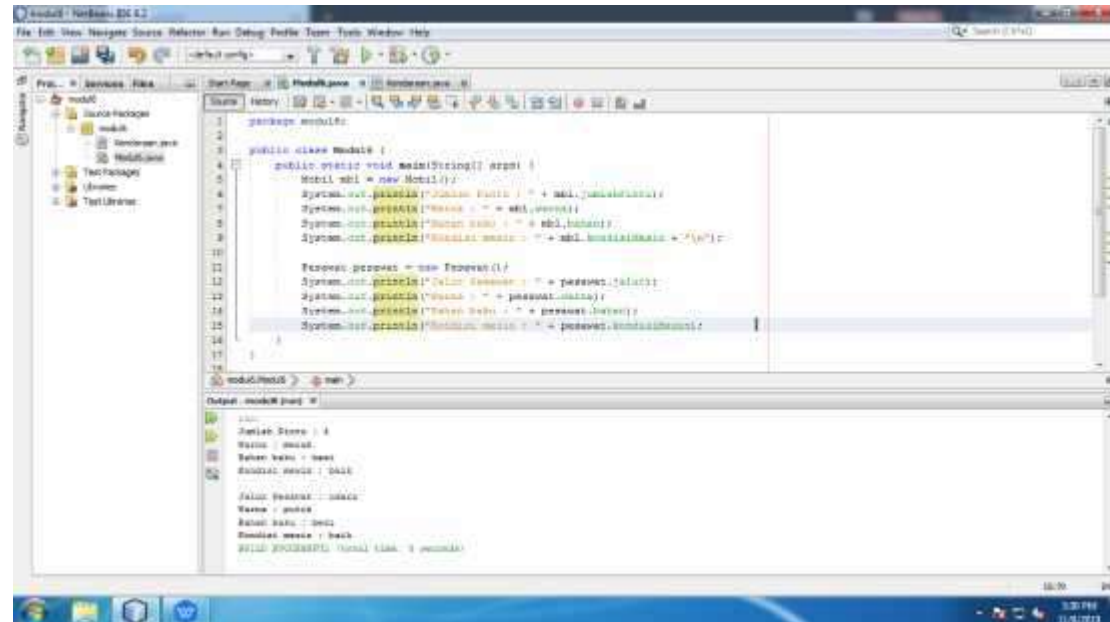
Nama : Alif Al Amin

NIM : L200180082

Kelas : B

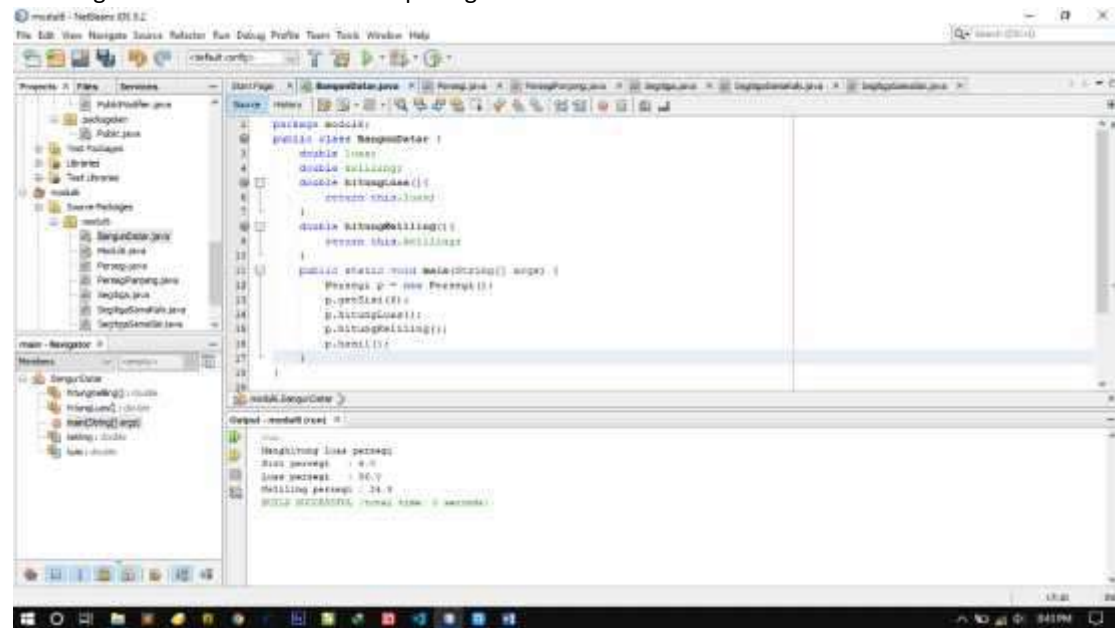
Laporan Modul ke 6

1. Latihan

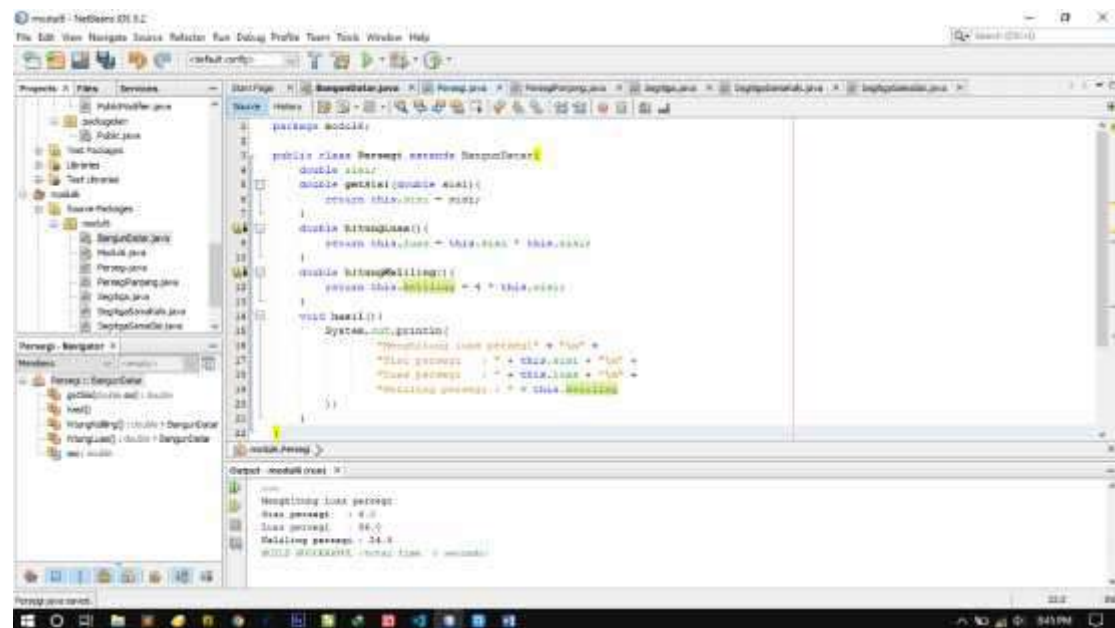


2. Tugas

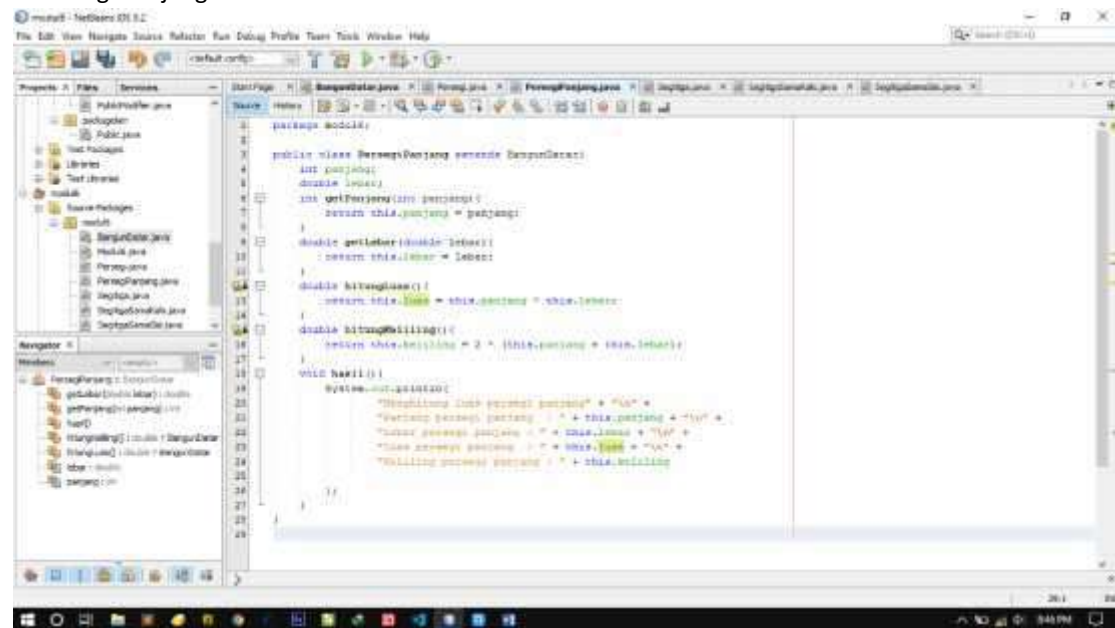
a. BangunDatar + contoh hasil dari persegi



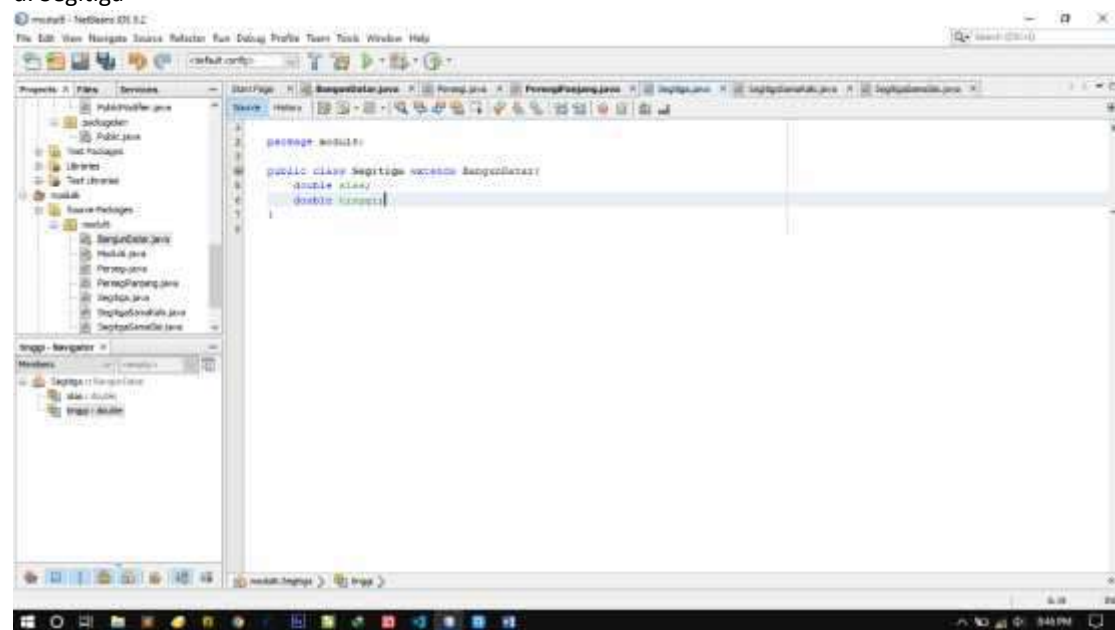
b. Persegi



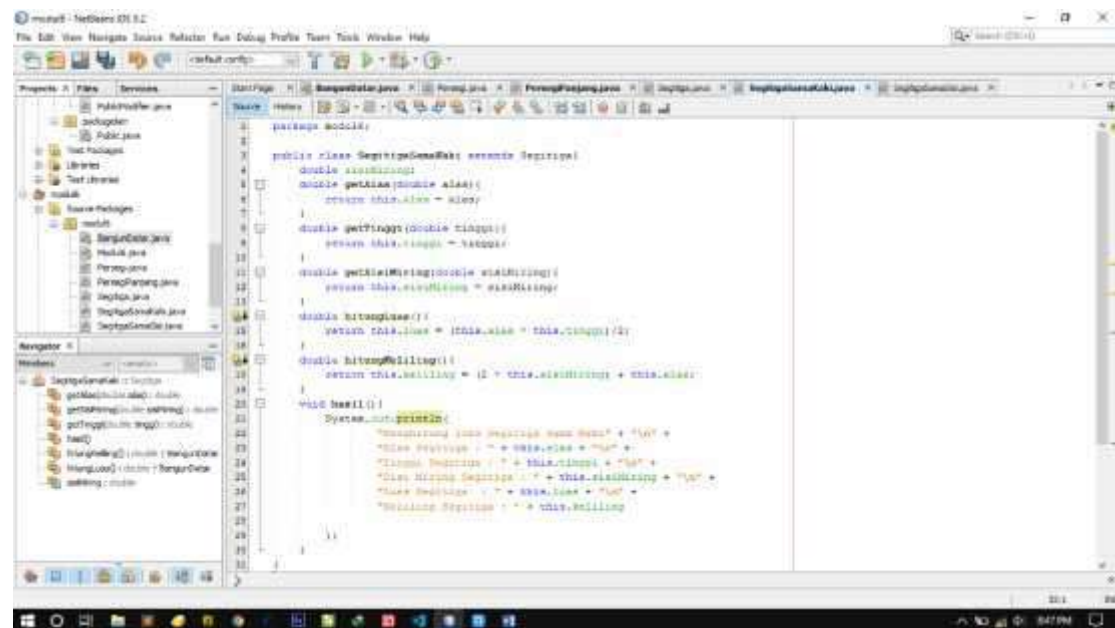
c. Persegi Panjang



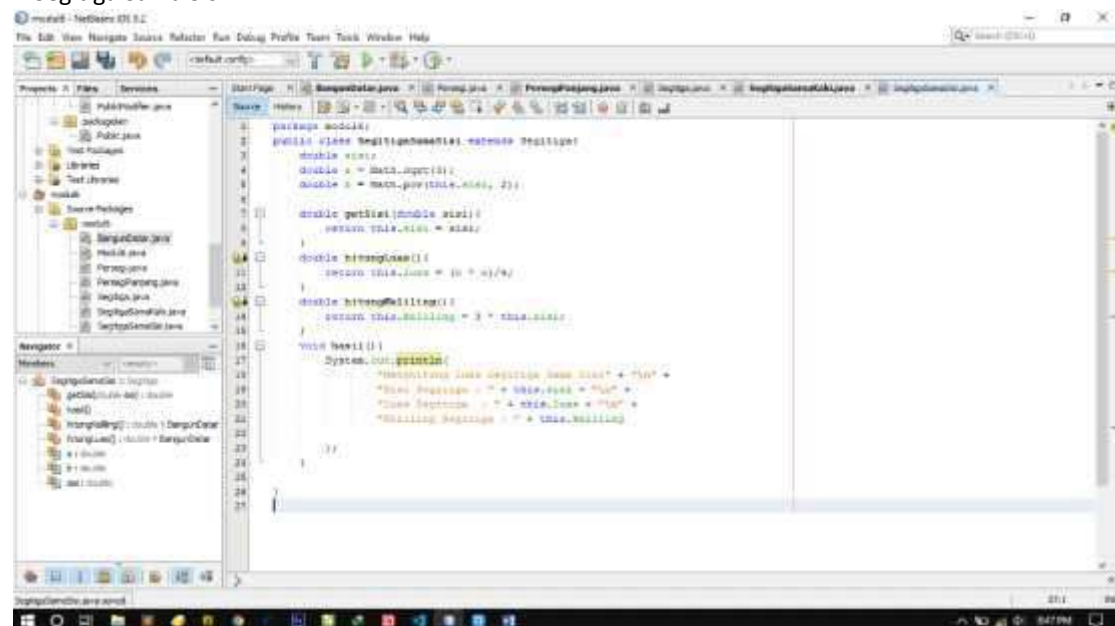
d. Segitiga



e. Segitiga Sama Kaki



f. Segitiga Sama Sisi



Nama : Alif Al Amin

NIM : L200180082

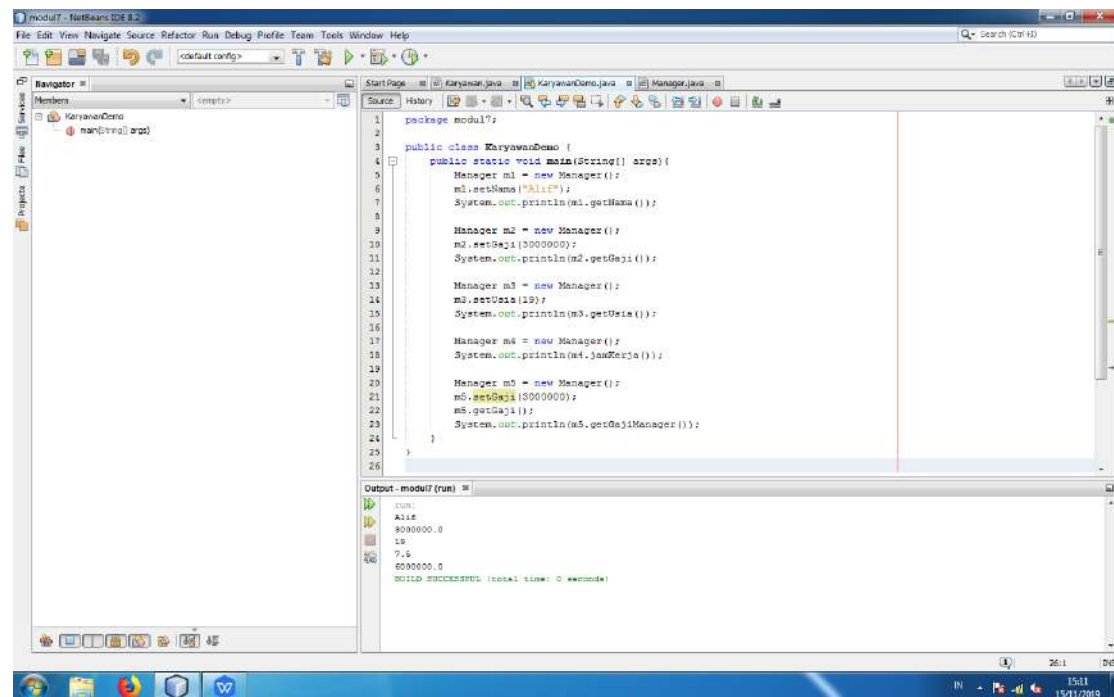
Kelas : B

Laporan Modul ke 7

7.2 Latihan

1. Menampilkan private variable dengan 5 Object

- Main() + output



```
package modul7;

public class KaryawanDemo {
    public static void main(String[] args) {
        Manager m1 = new Manager();
        m1.setName("Ali");
        System.out.println(m1.getName());

        Manager m2 = new Manager();
        m2.setGaji(3000000);
        System.out.println(m2.getGaji());

        Manager m3 = new Manager();
        m3.setUraan(15);
        System.out.println(m3.getUraan());

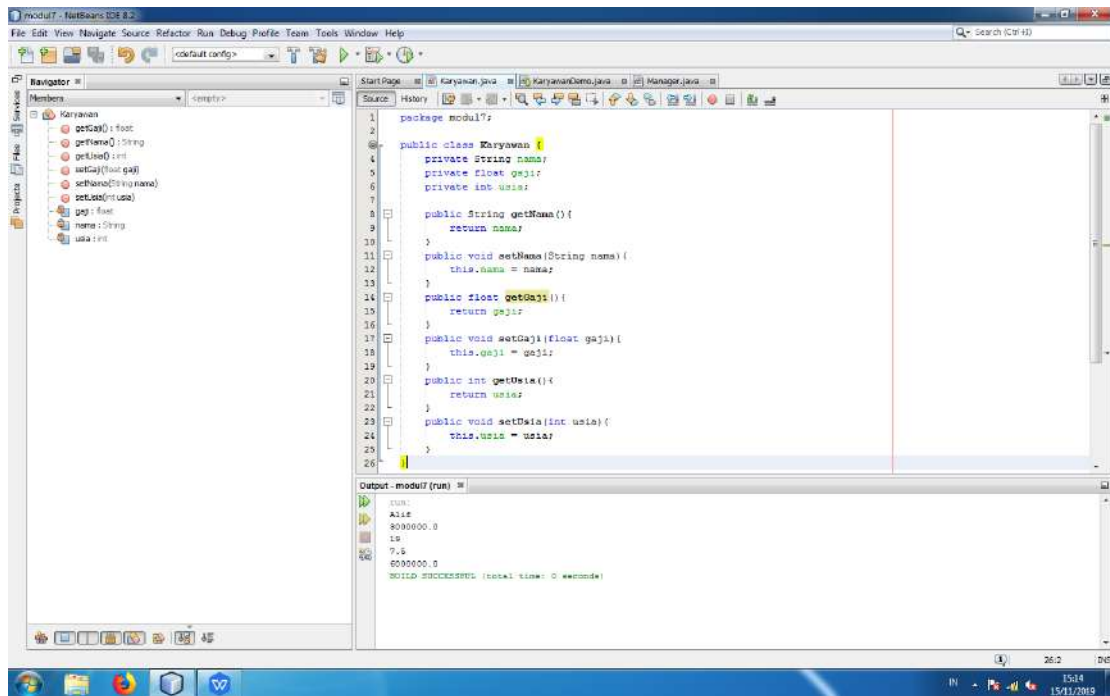
        Manager m4 = new Manager();
        System.out.println(m4.jawabRaja());

        Manager m5 = new Manager();
        m5.setGaji(3000000);
        m5.getGaji();
        System.out.println(m5.getGajiManager());
    }
}
```

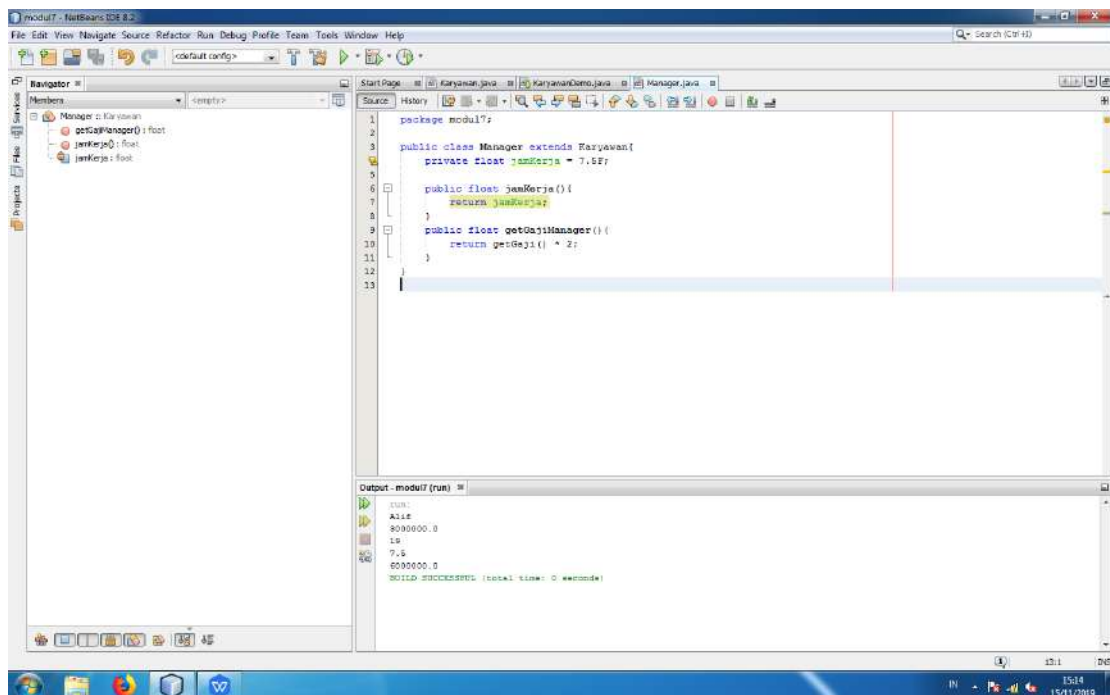
Output - modul7 (run)

```
Ali
3000000.0
15
7.6
3000000.0
BUILD SUCCESSFUL (total time: 0 ms)
```

- class Karyawan

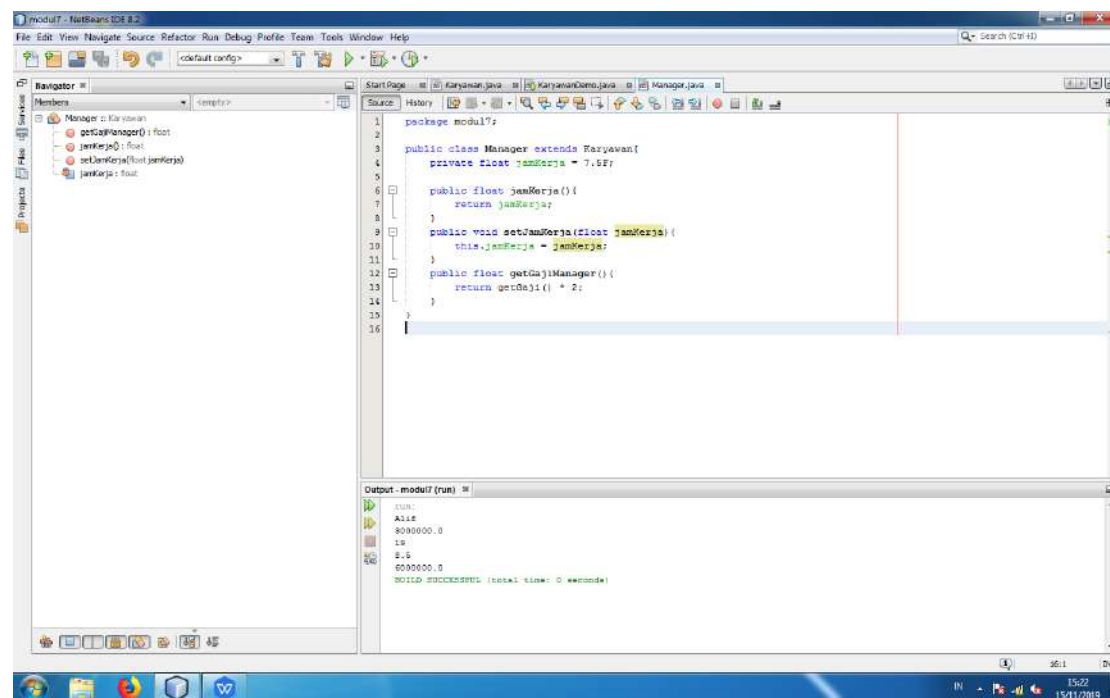


- class Manager

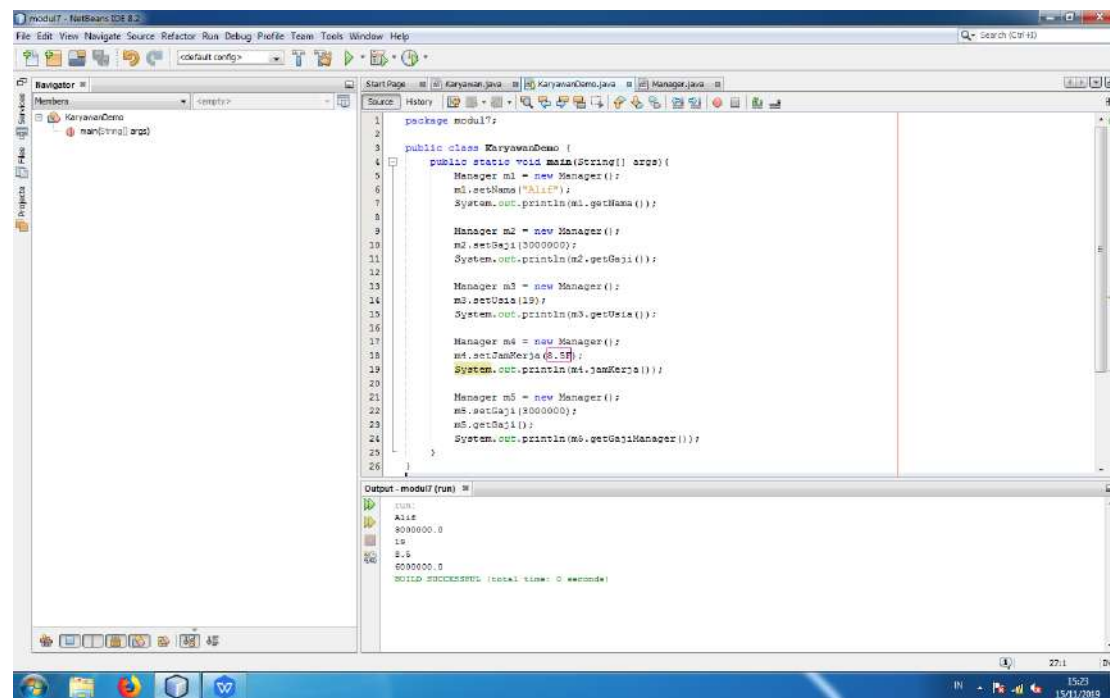


2. Menambahkan setter pada class Manager untuk memodifikasi nilai baru jamKerja menjadi 8.5

- class manager setelah ditambah setter



- main() setelah memodifikasi nilai jamKerja menjadi 8.5 + output



Nama : Alif Al Amin

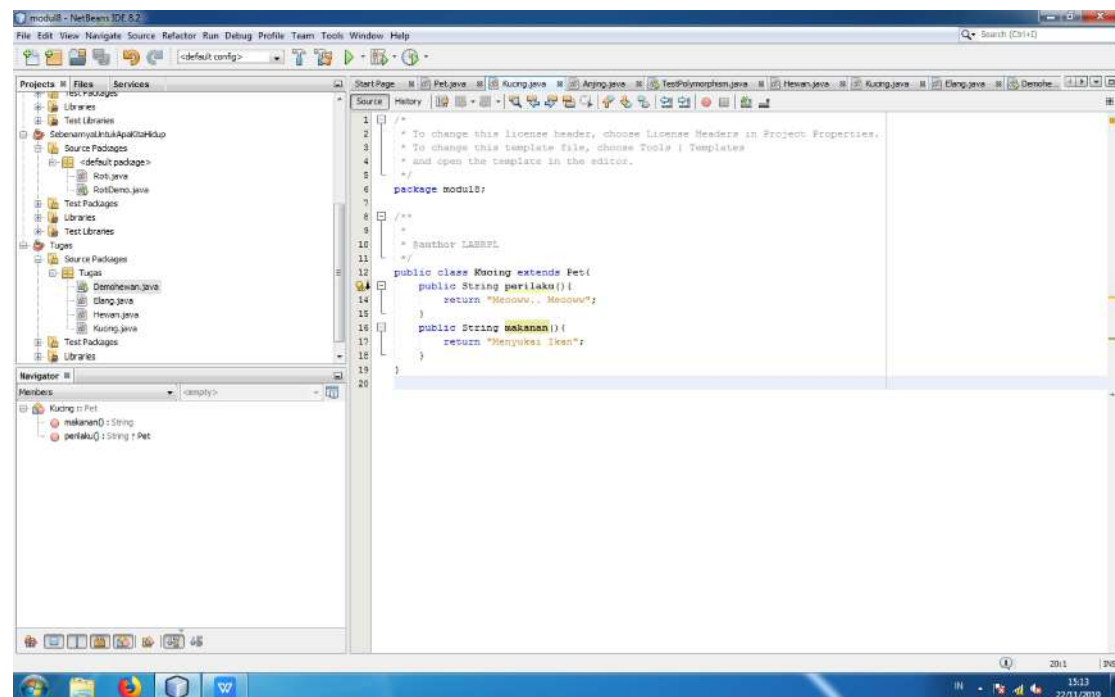
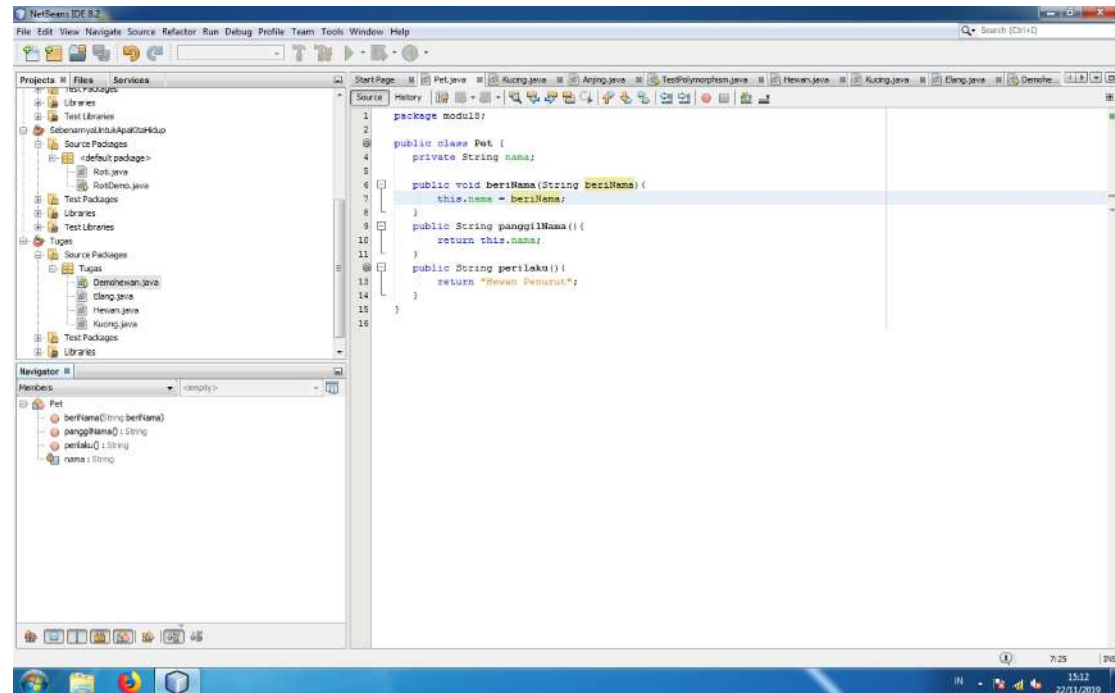
NIM : L200180082

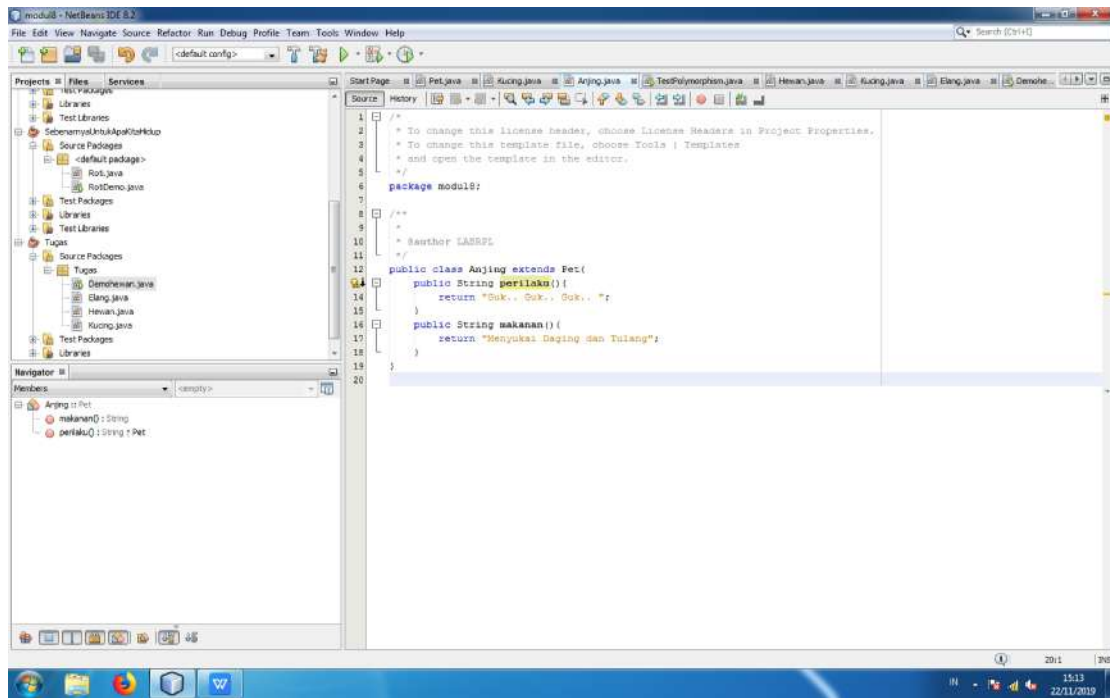
Kelas : B

Laporan Modul ke 8

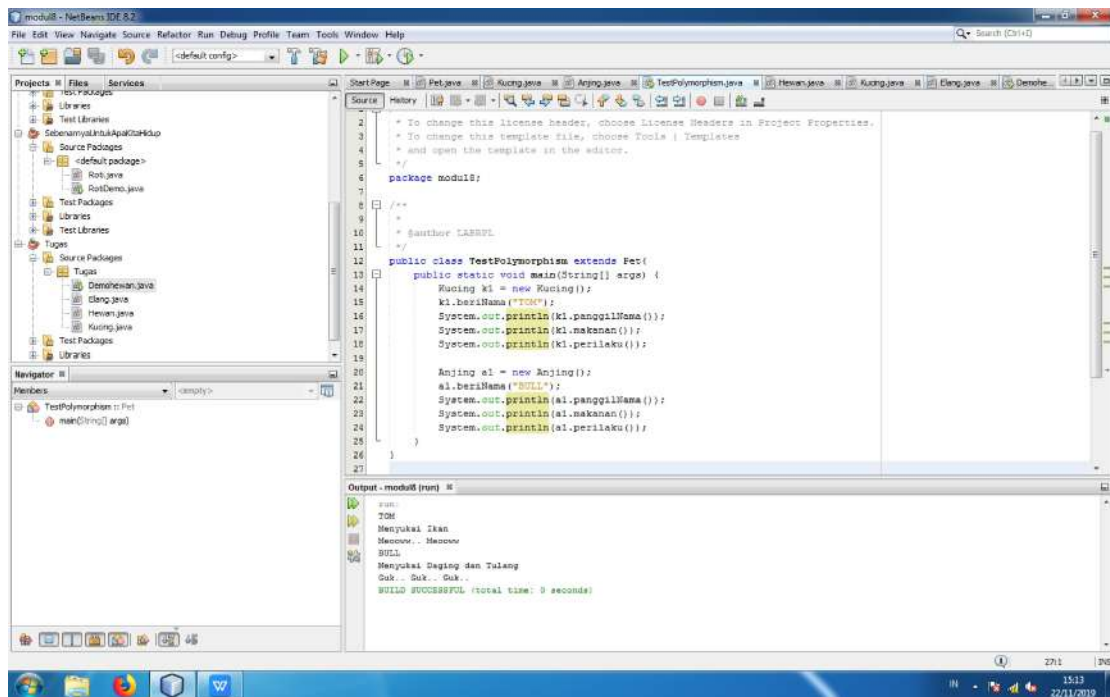
8.2 Latihan

1. Membuat class kucing dan anjing dan melakukan overriding terhadap method perilaku() dan menambah satu method khusus



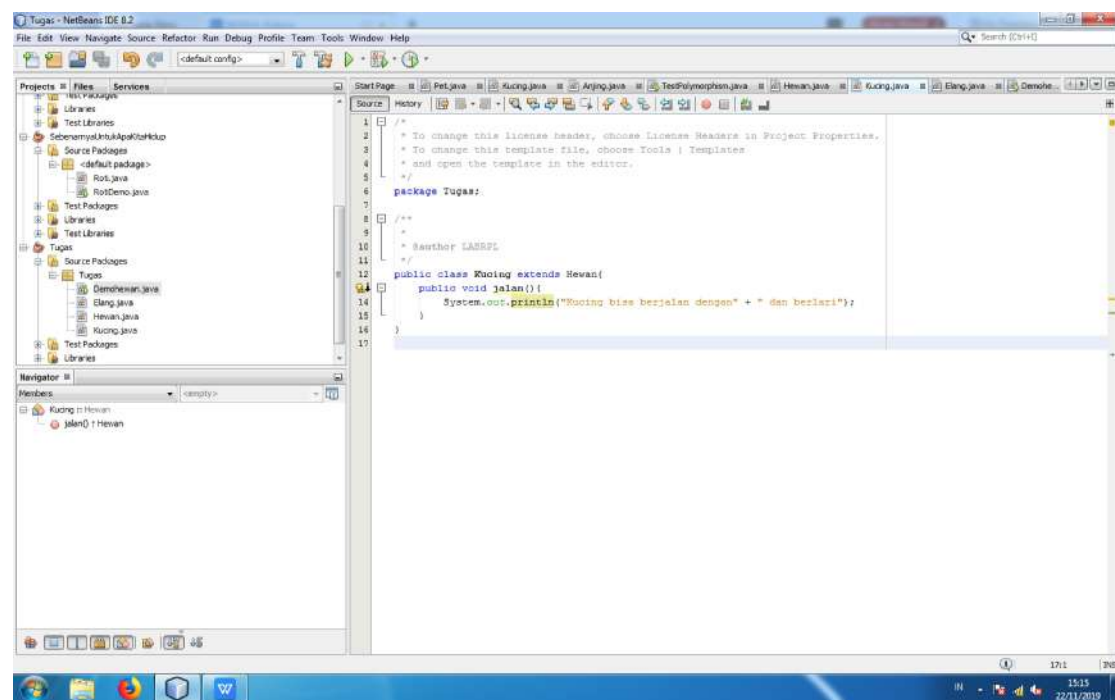
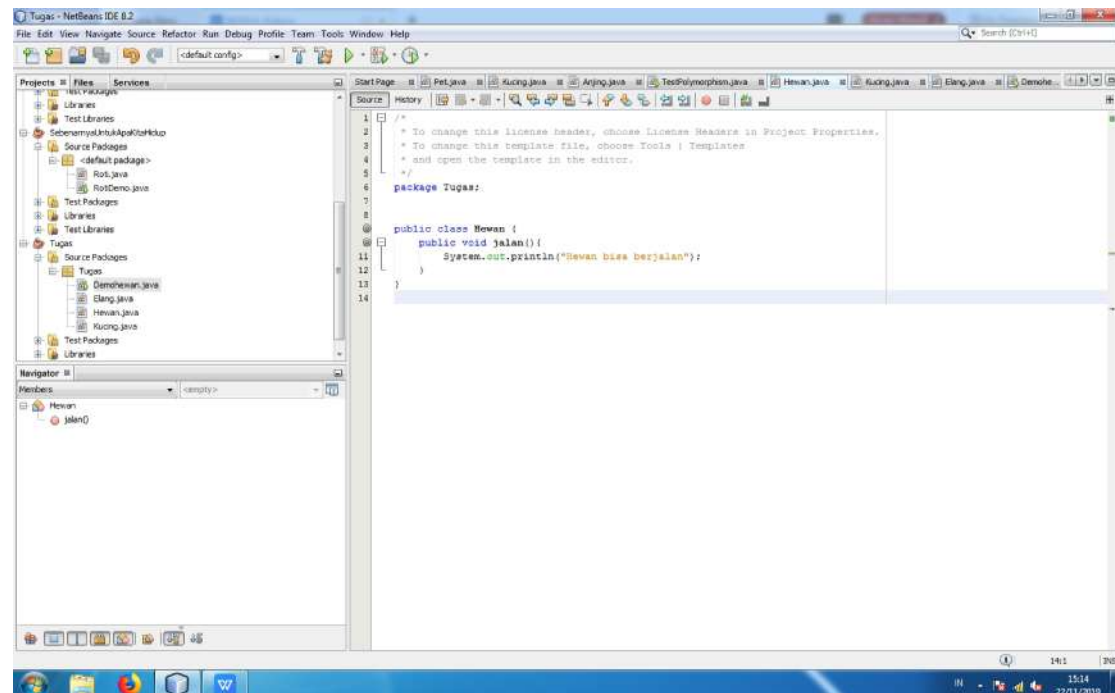


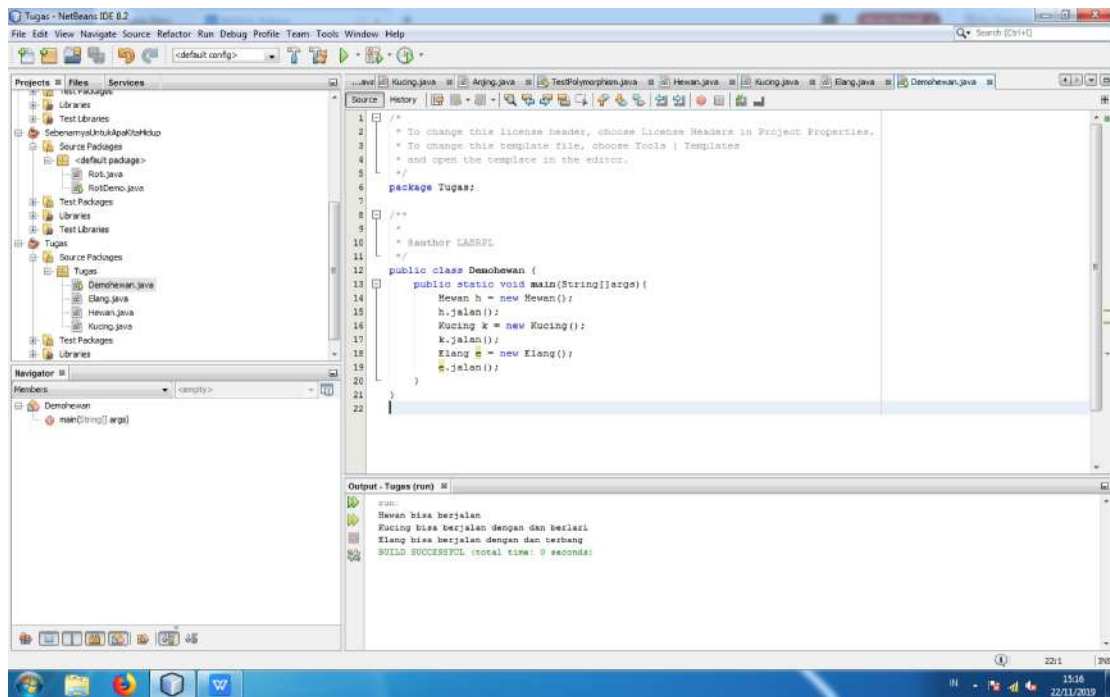
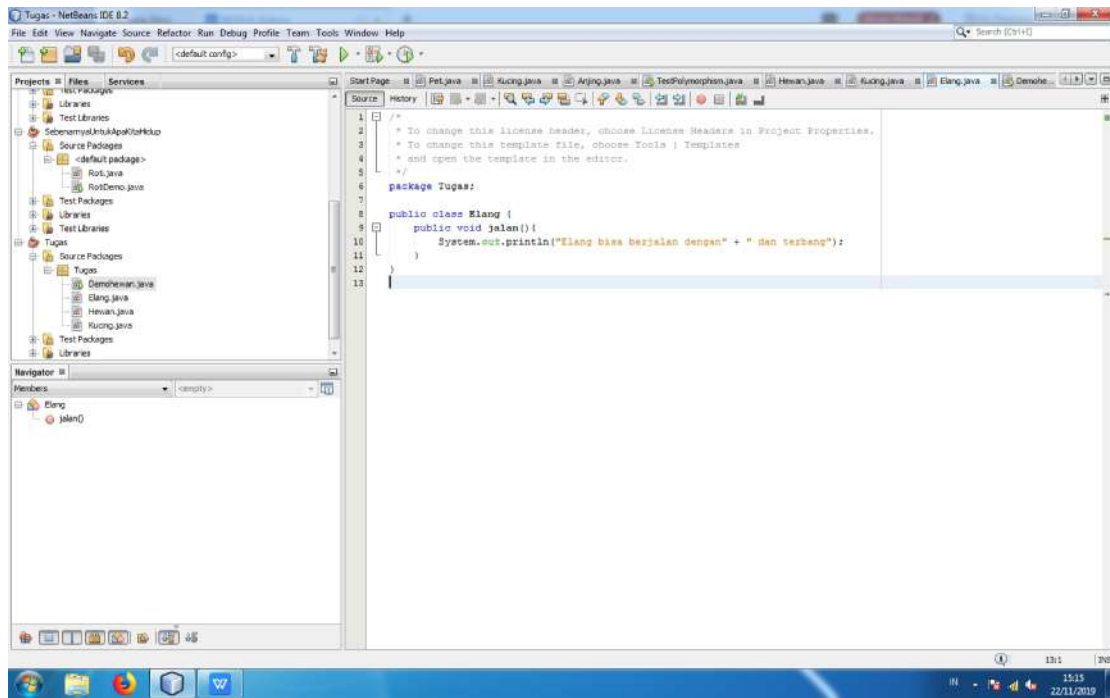
2. Membuat class TestPolymorphism untuk output



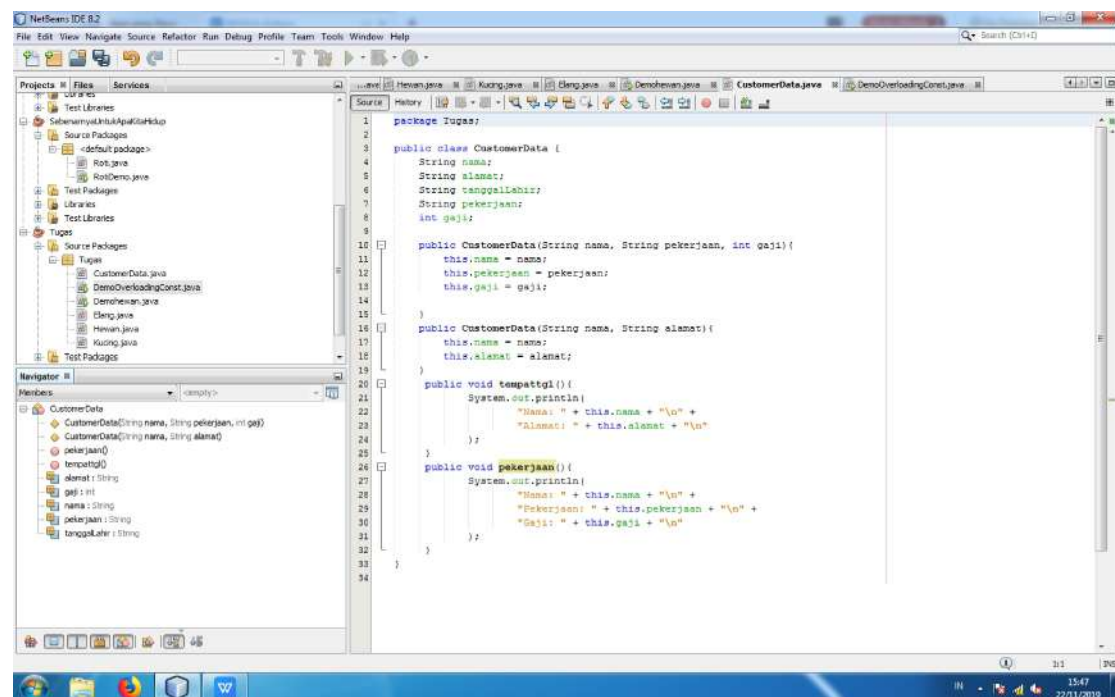
8.3 Tugas

1. Membuat class Elang dan overriding method jalan()

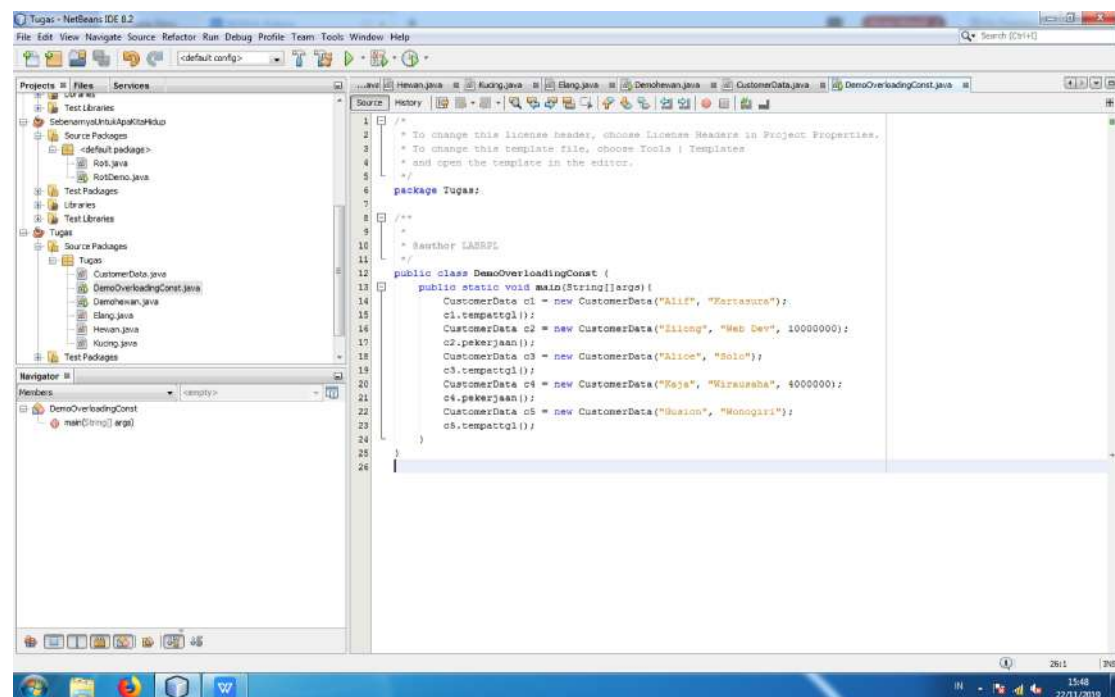


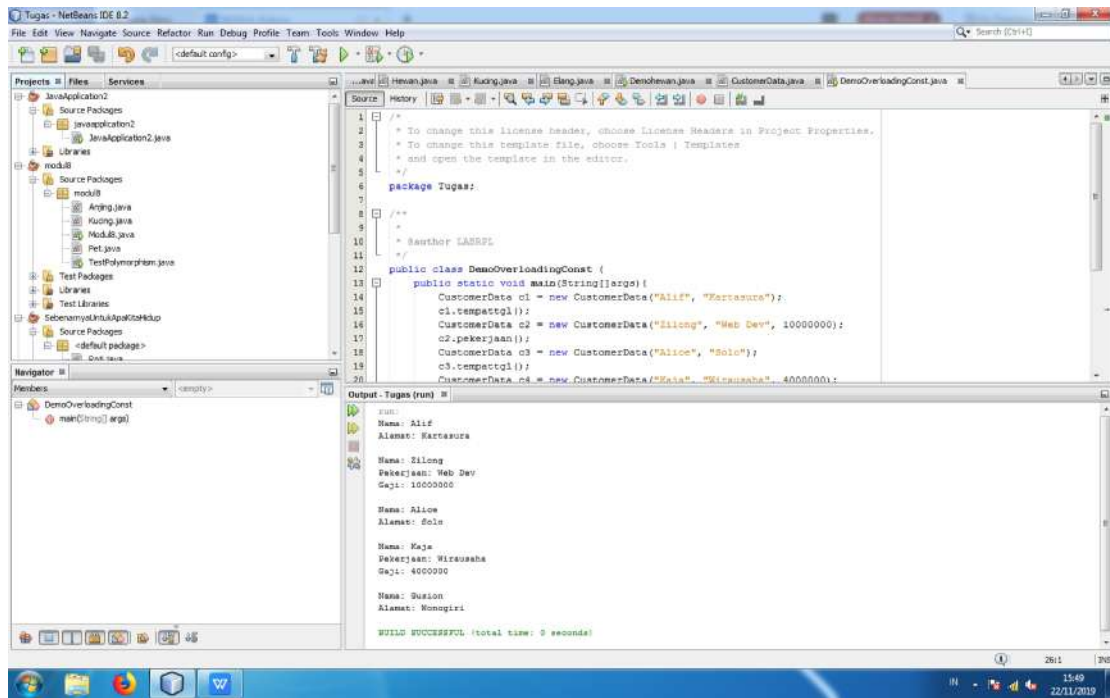


2. Membuat overloading constructor dari class CustomerData

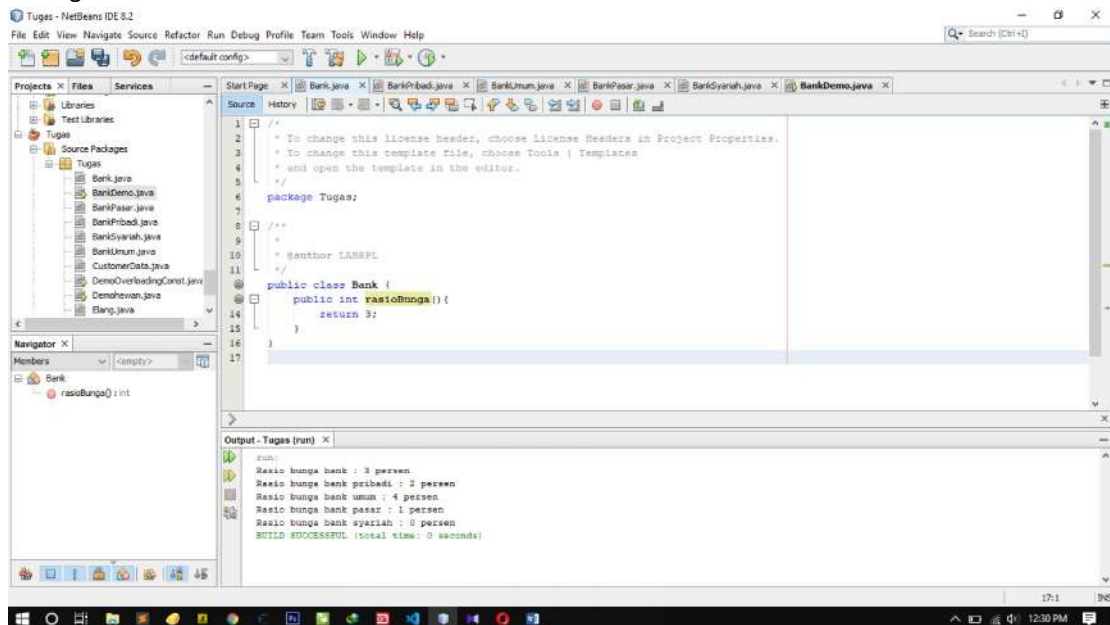


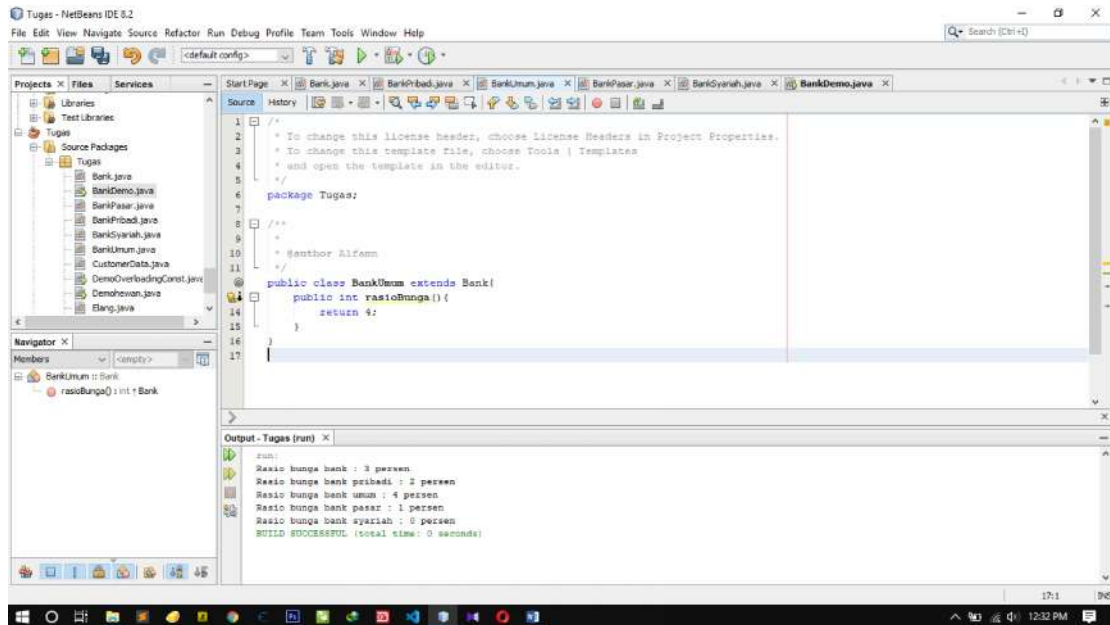
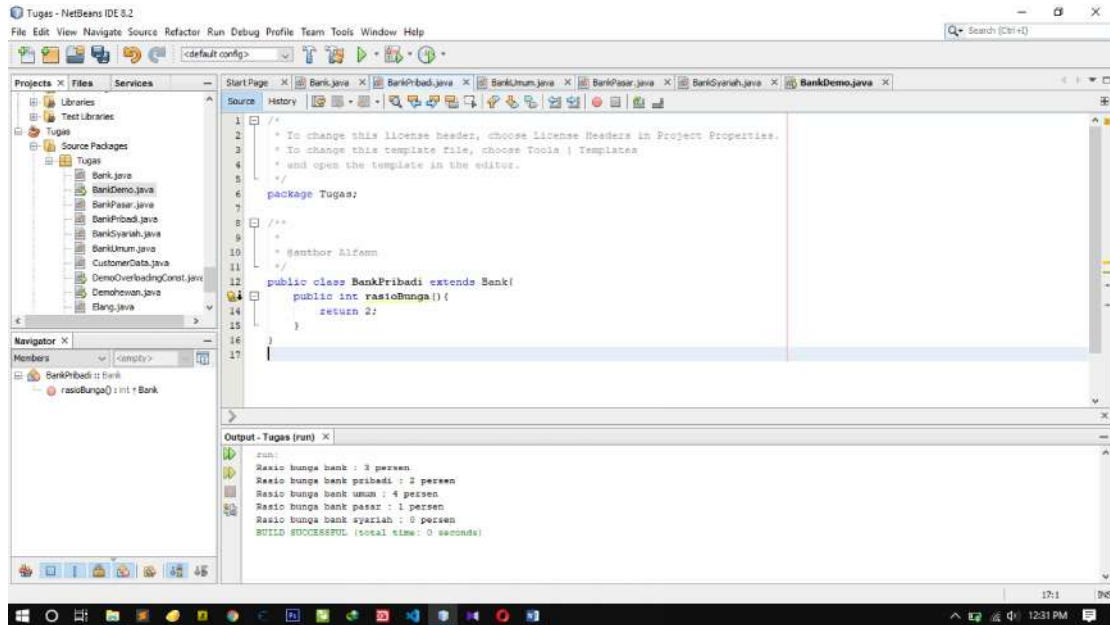
3. Membuat method main() dan membuat 5 object beserta output

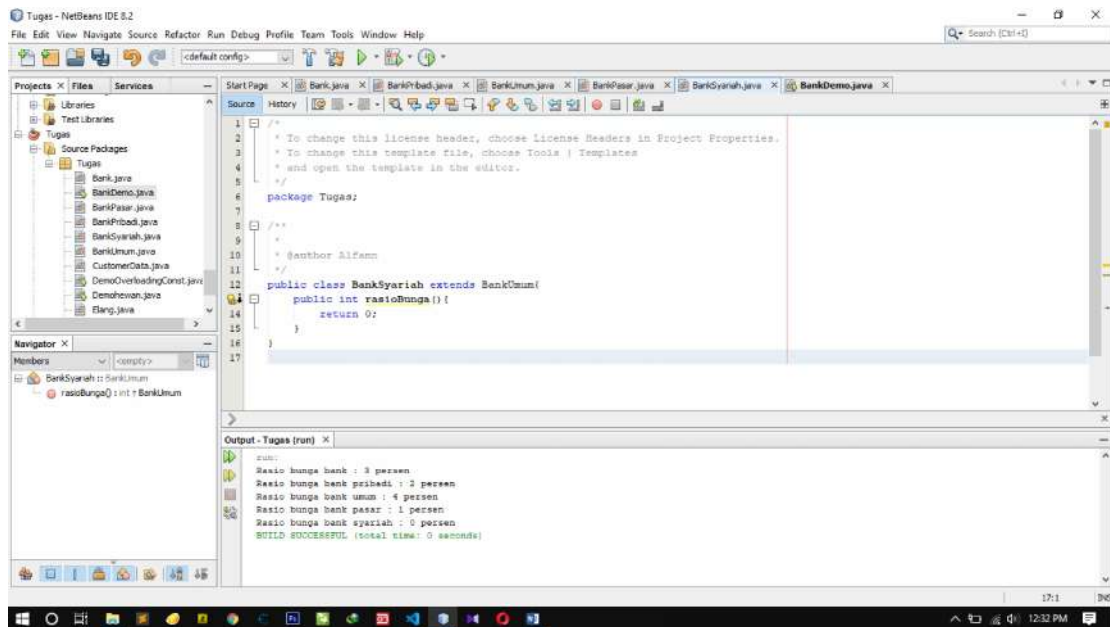
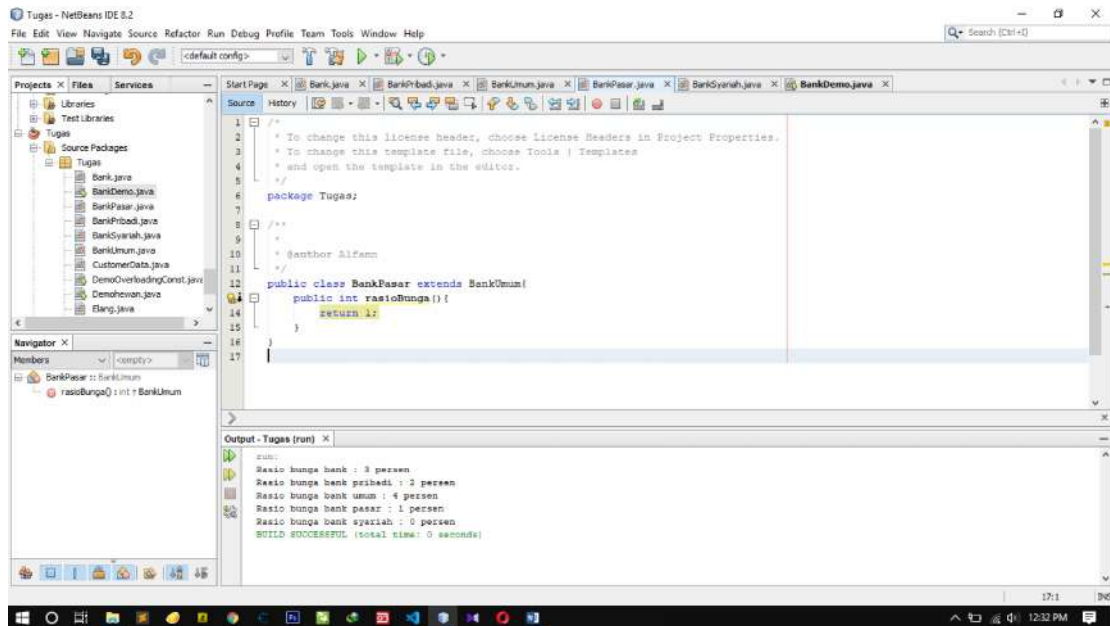




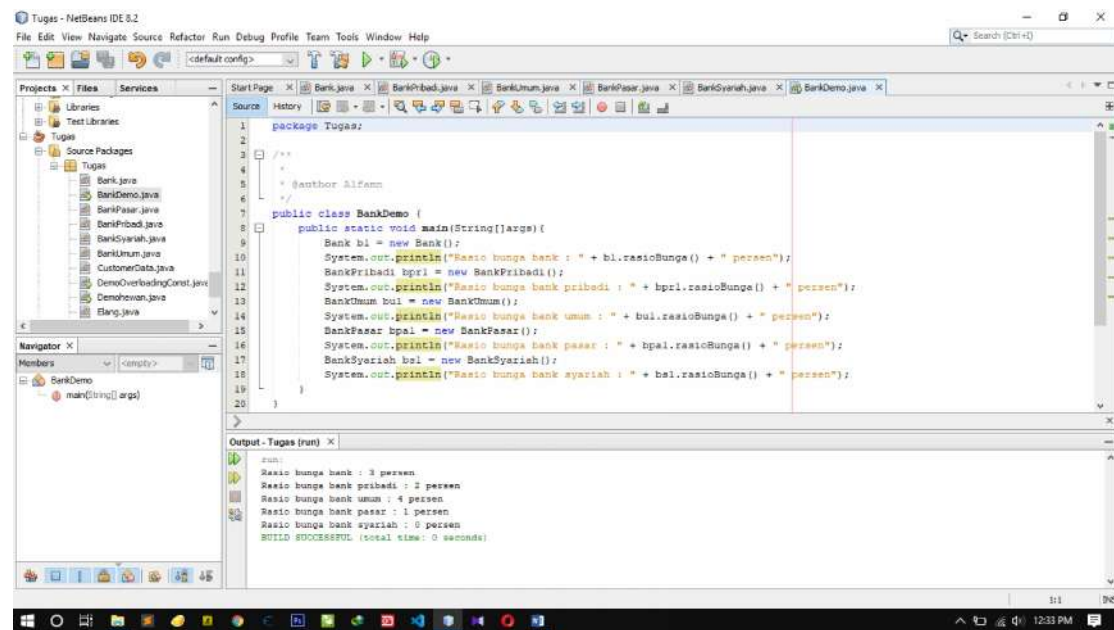
4. Diagram UML class bank







- Main() dan output



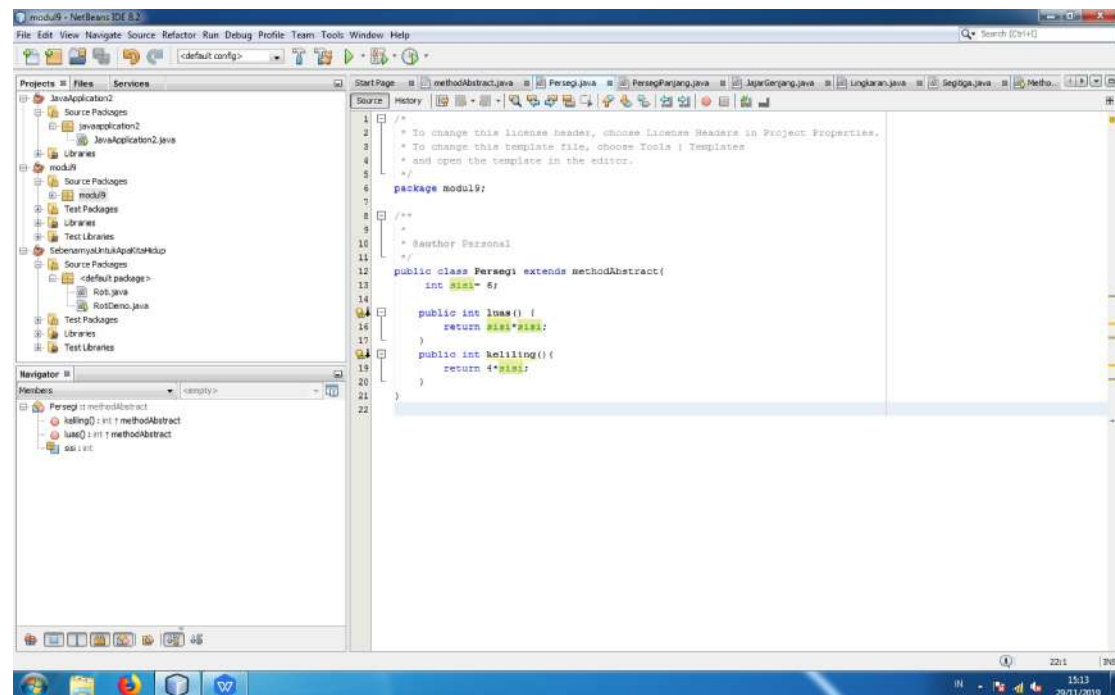
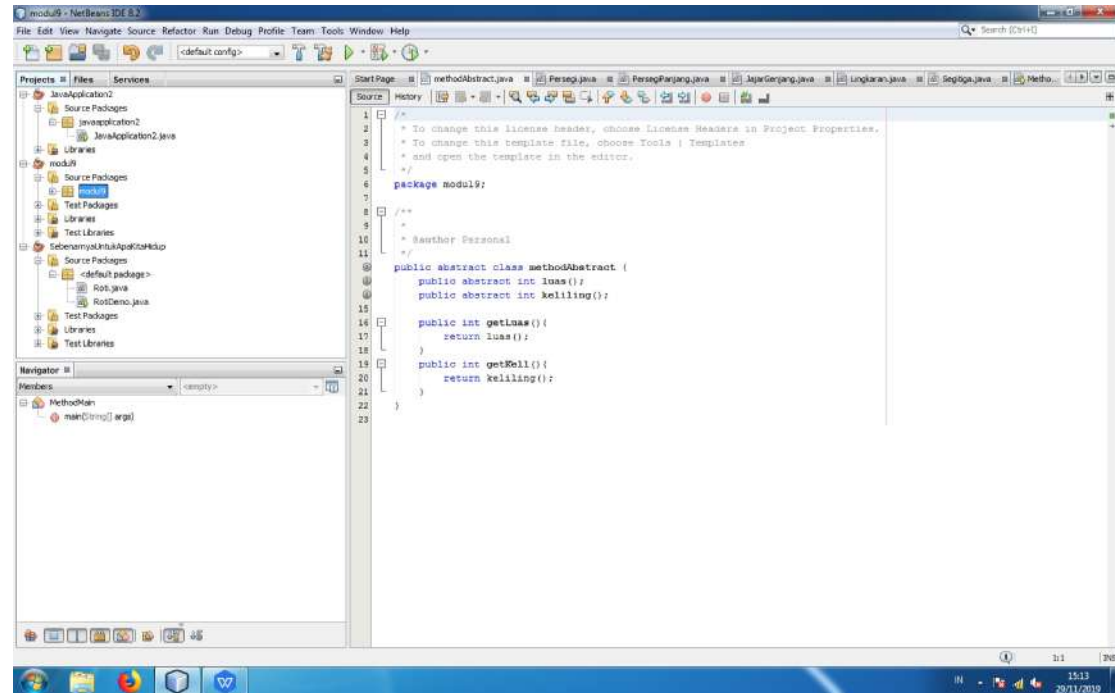
Nama : Alif Al Amin

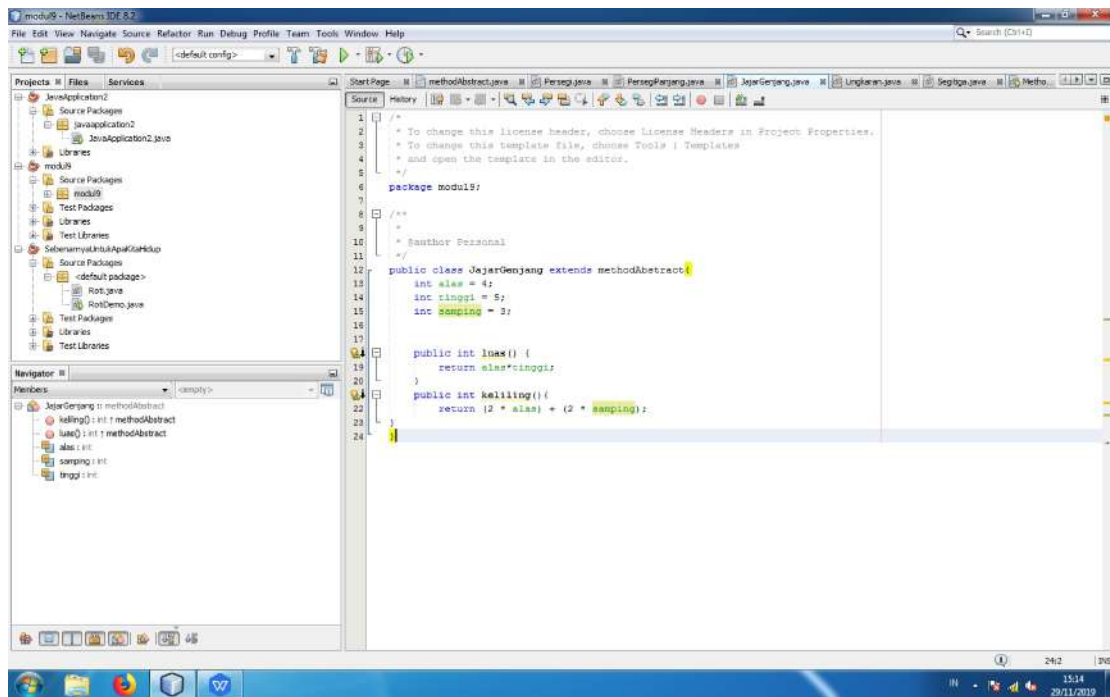
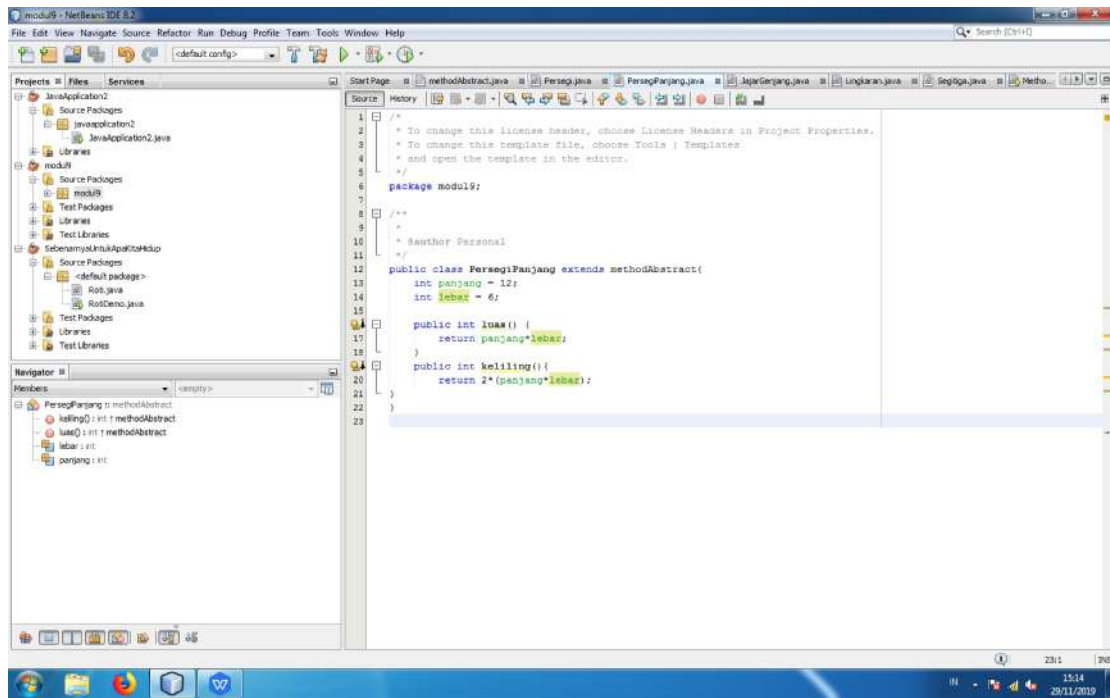
NIM : L200180082

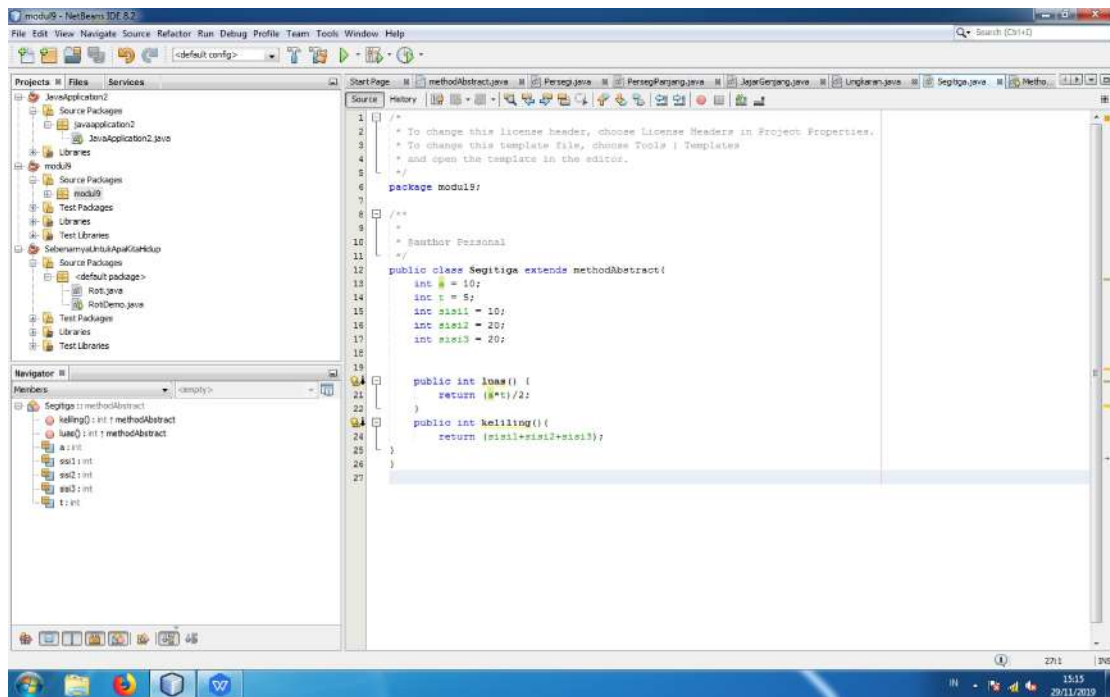
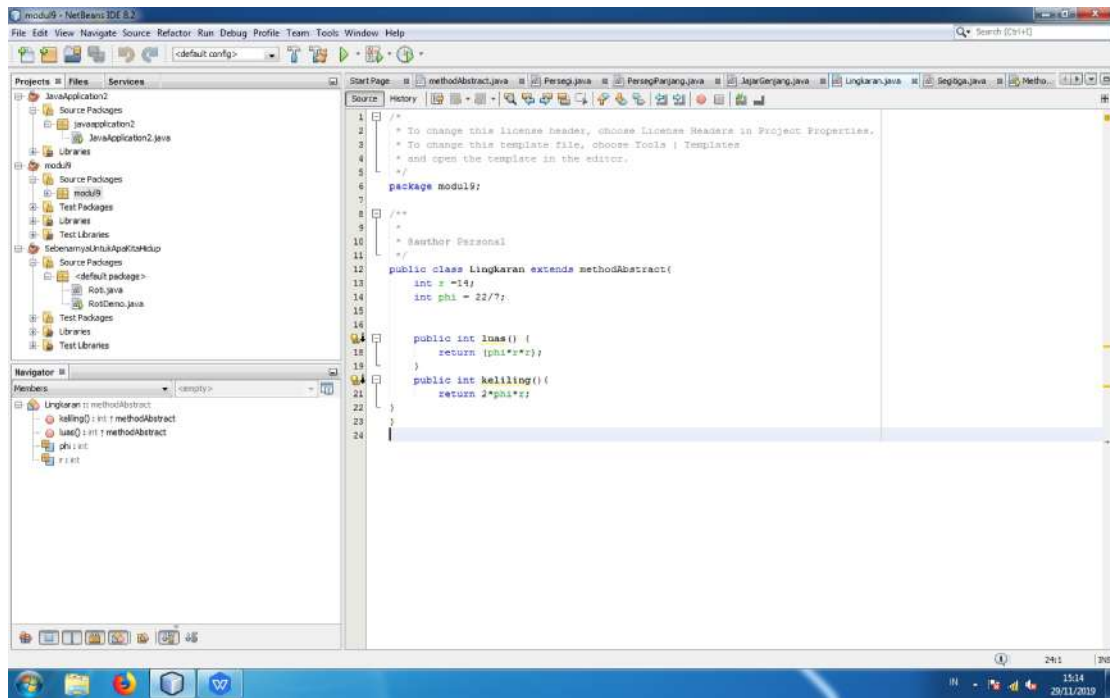
Kelas : B

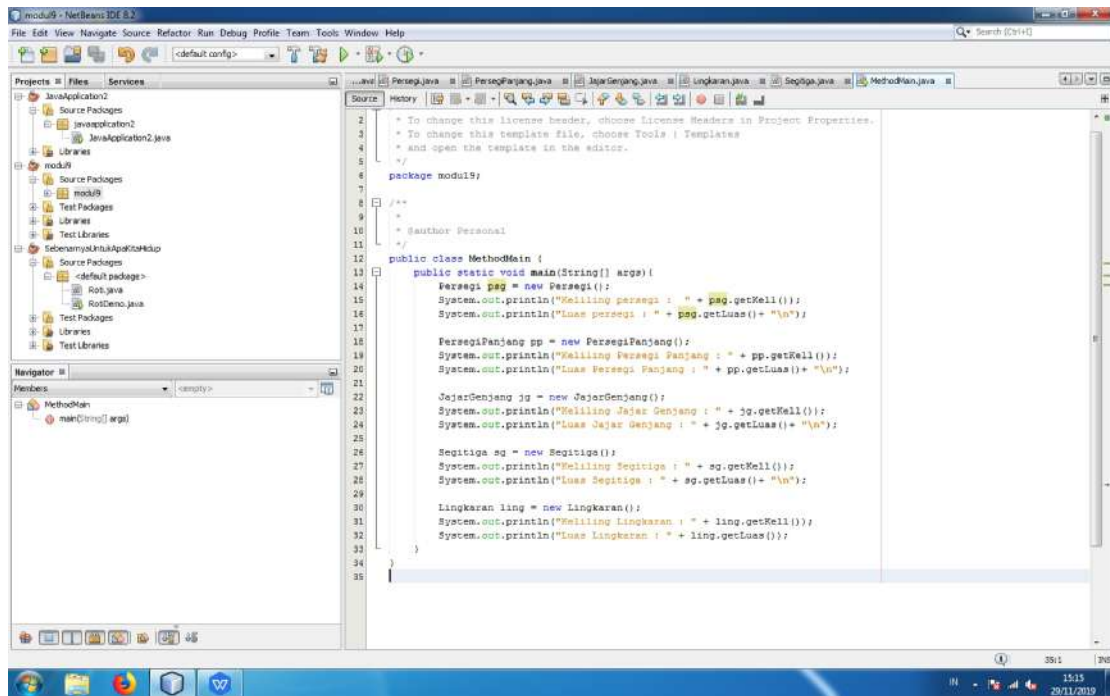
Laporan Modul ke 9

9.3 Latihan

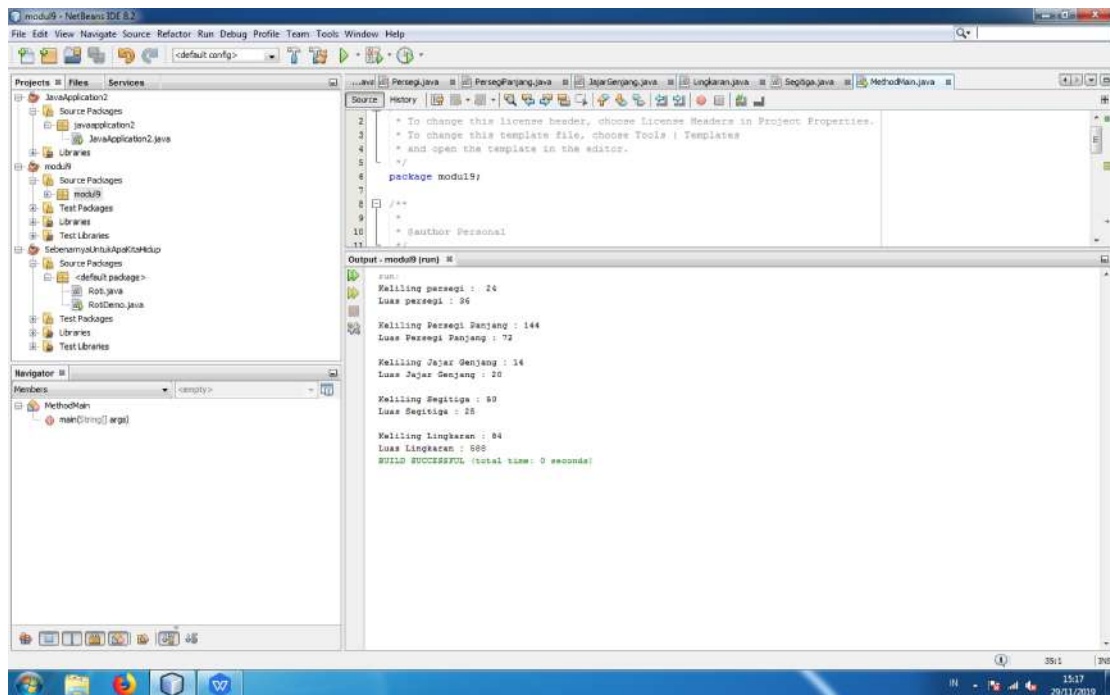




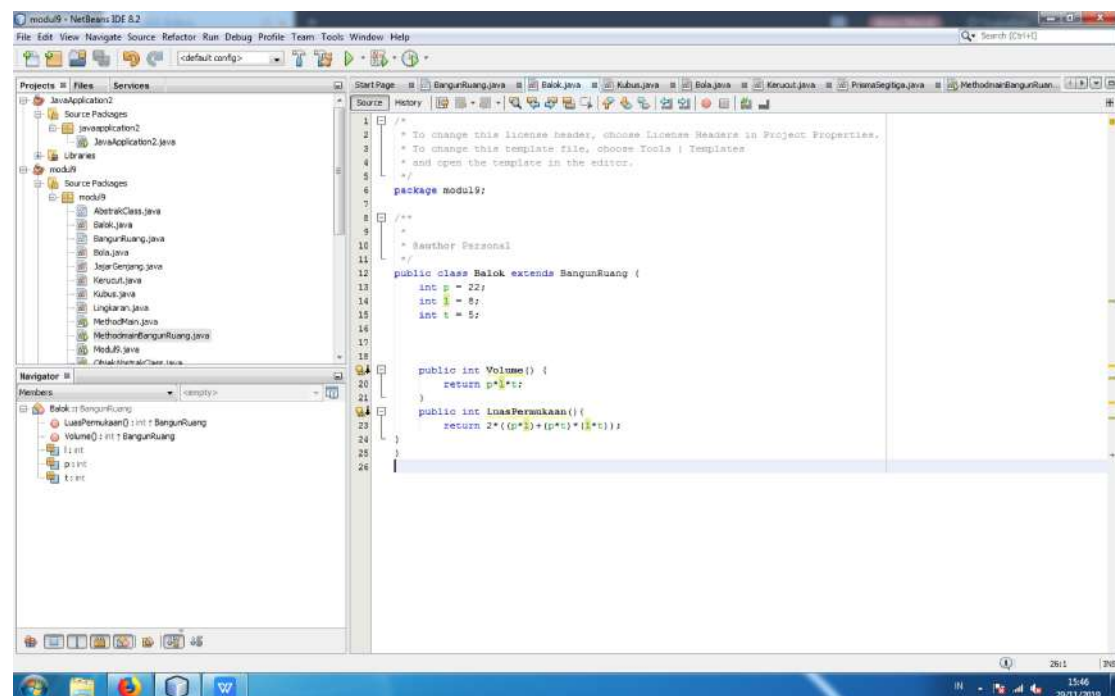
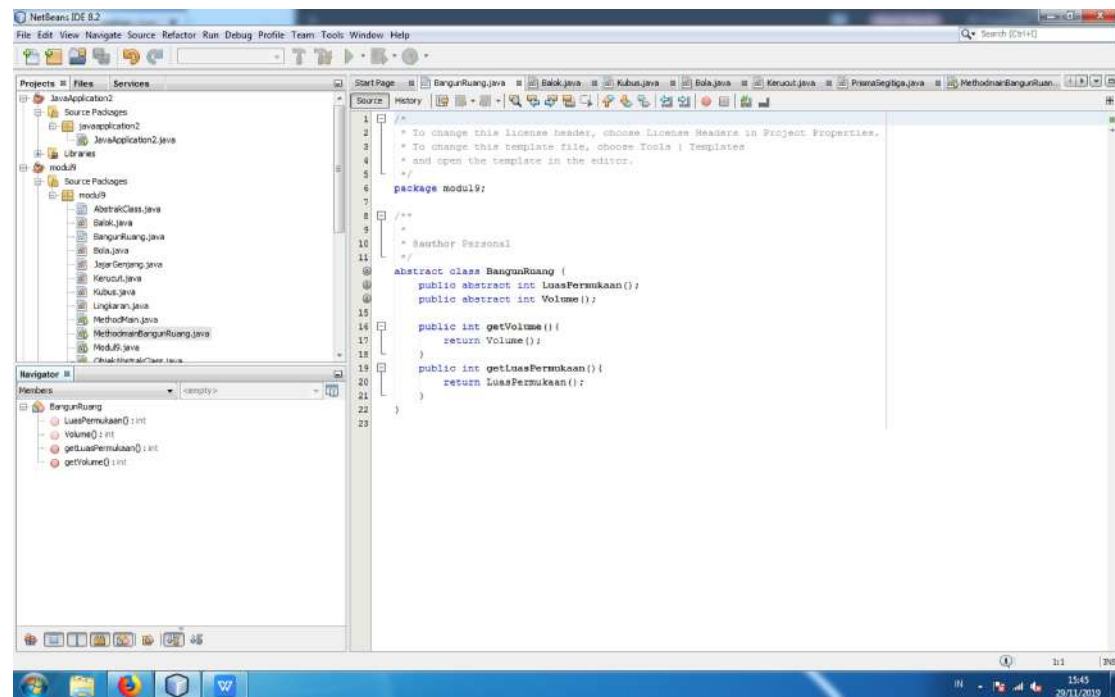


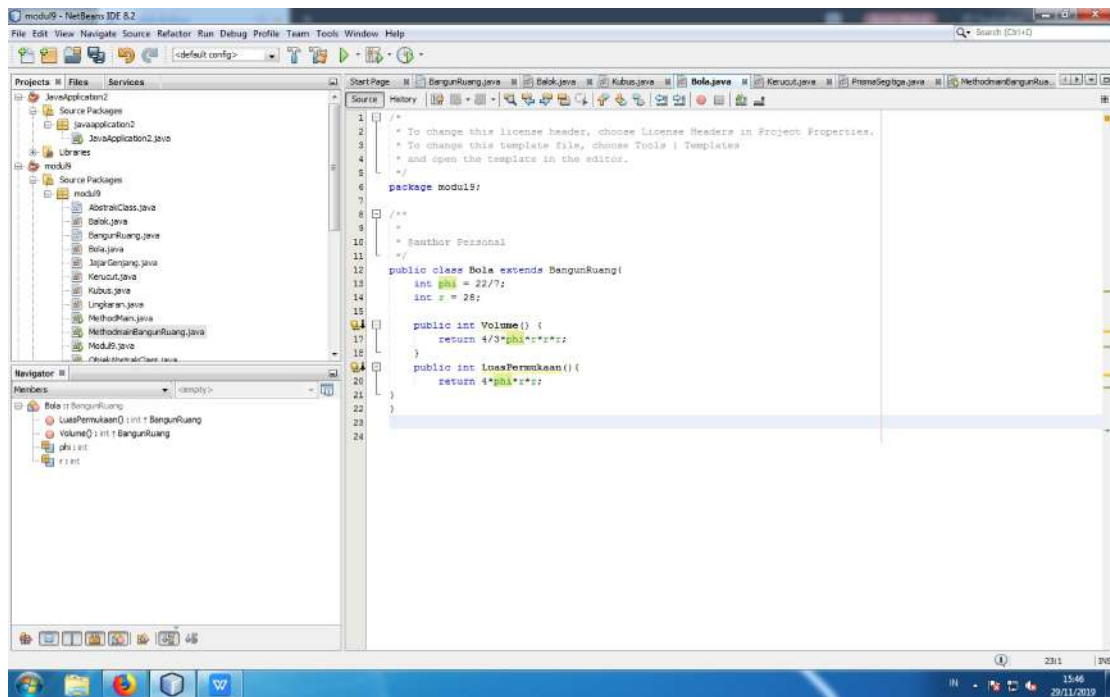
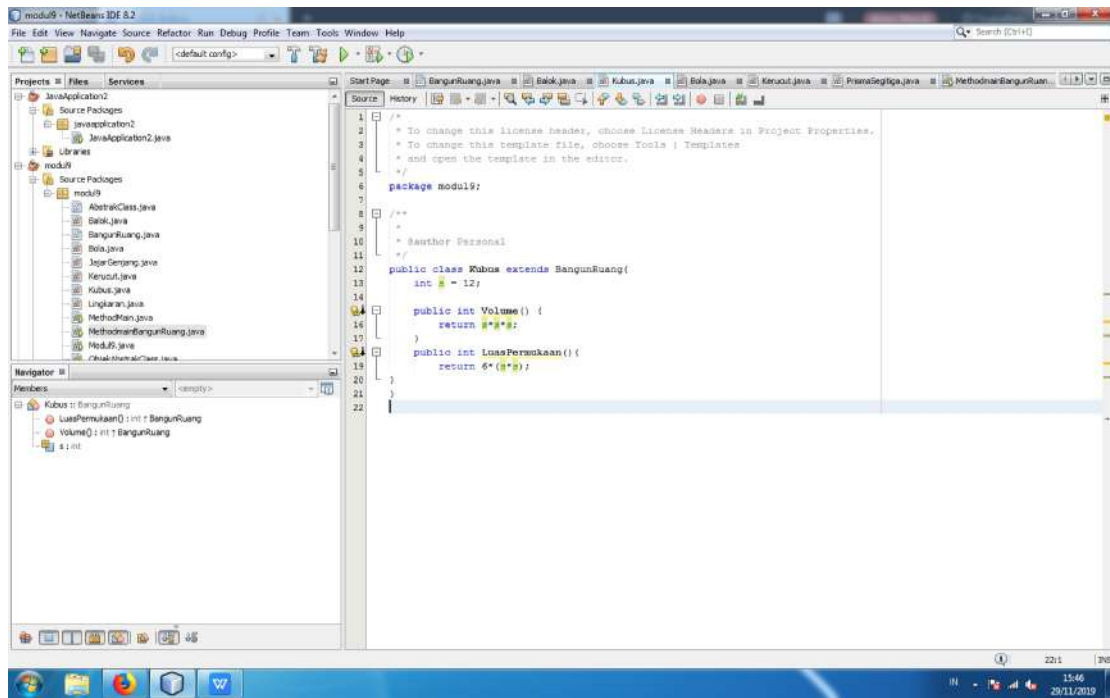


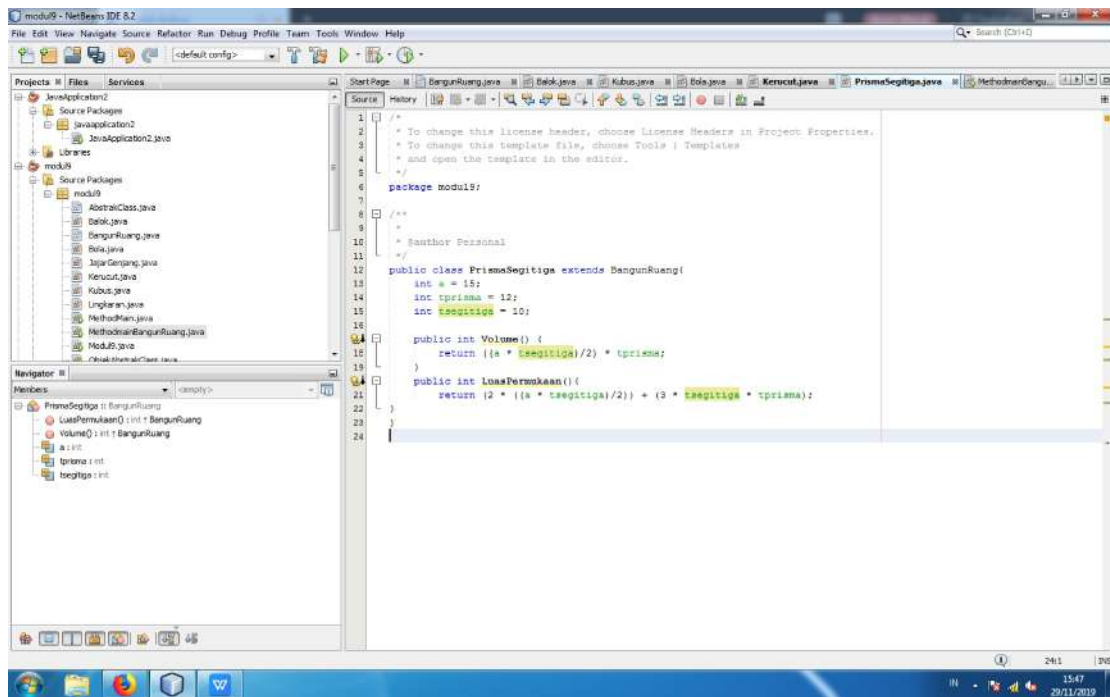
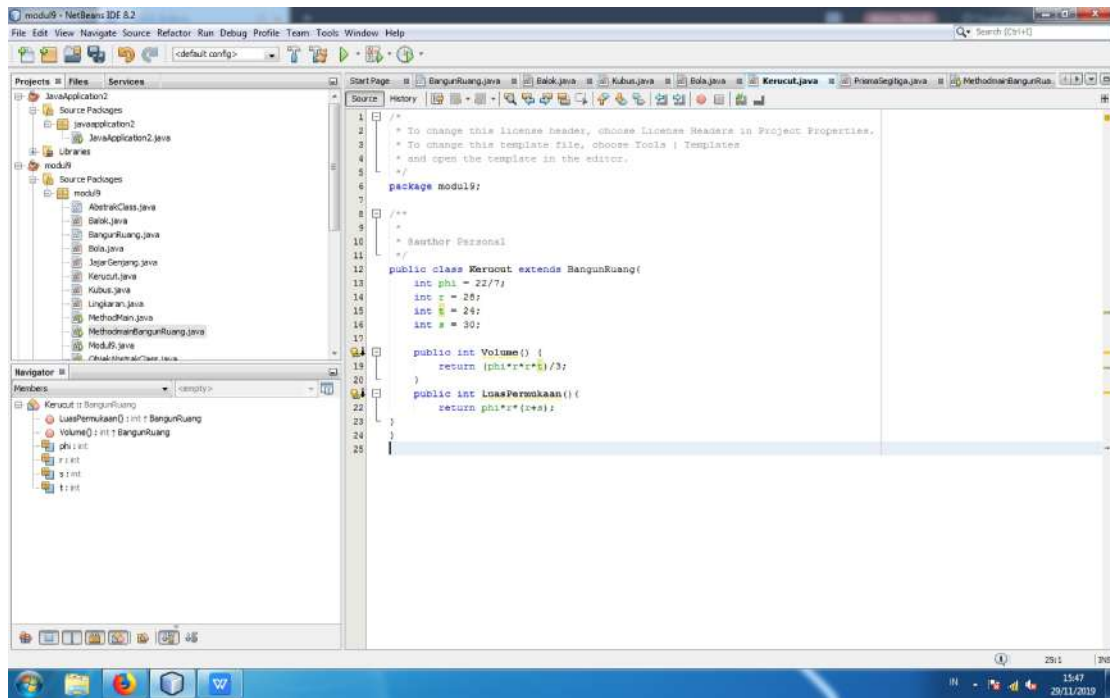
- output

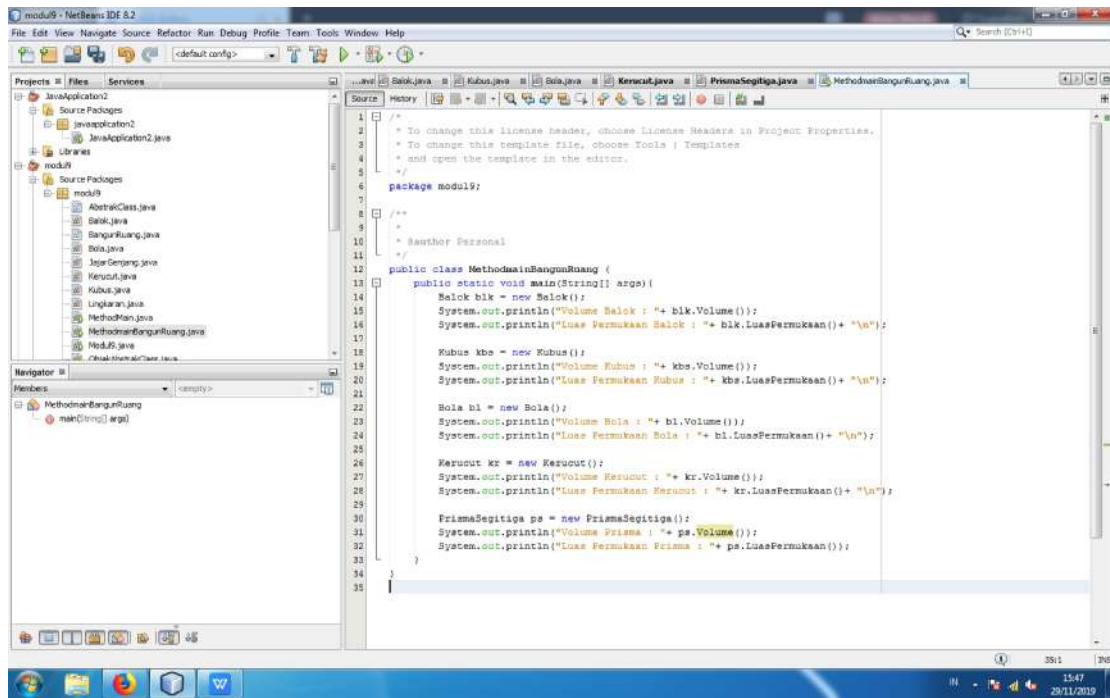


9.4 Tugas

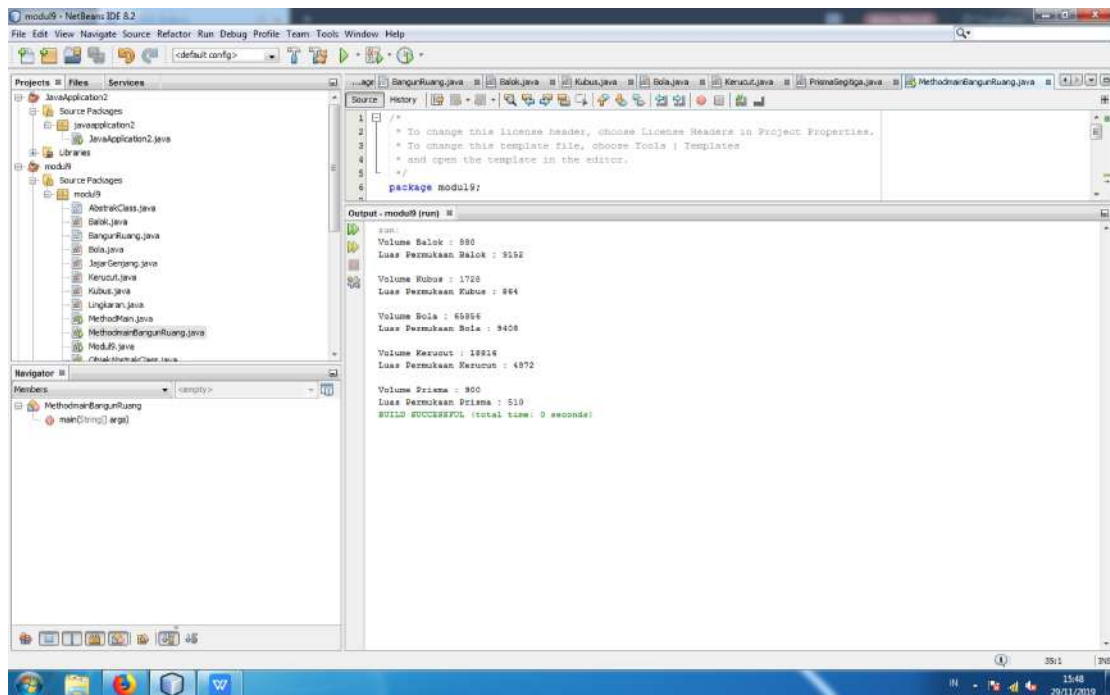








- output



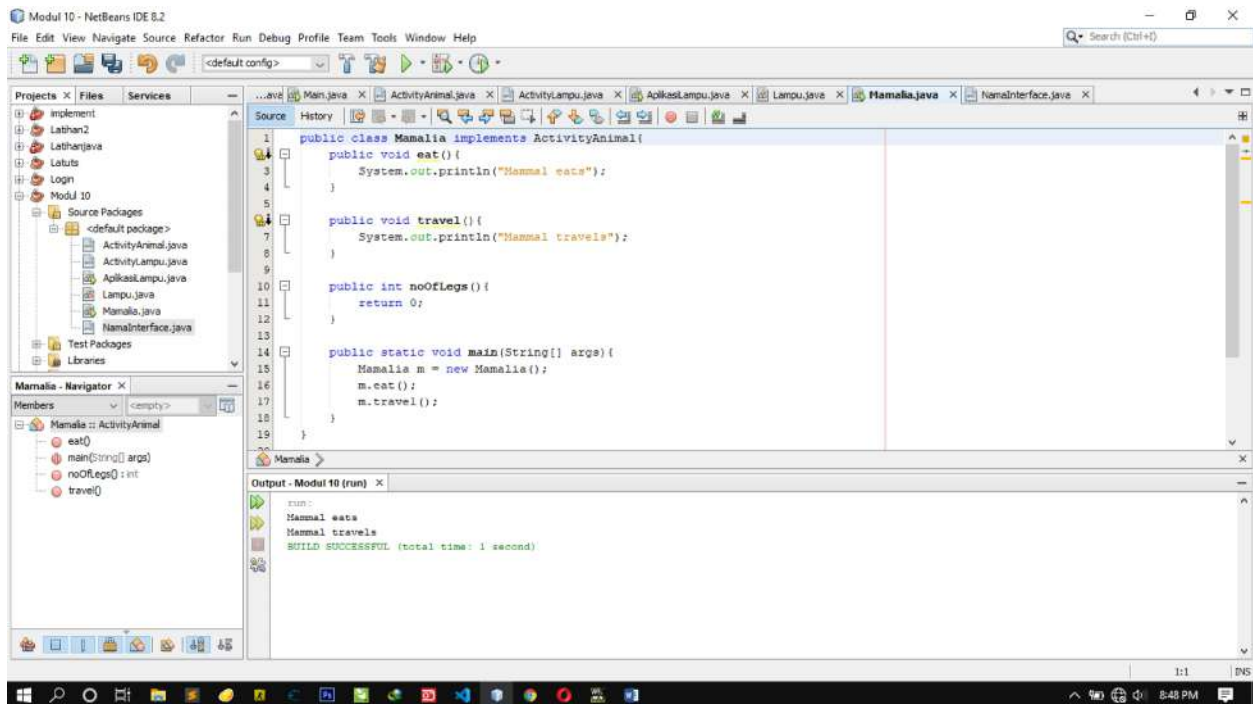
Nama : Alif Al Amin

NIM : L200180082

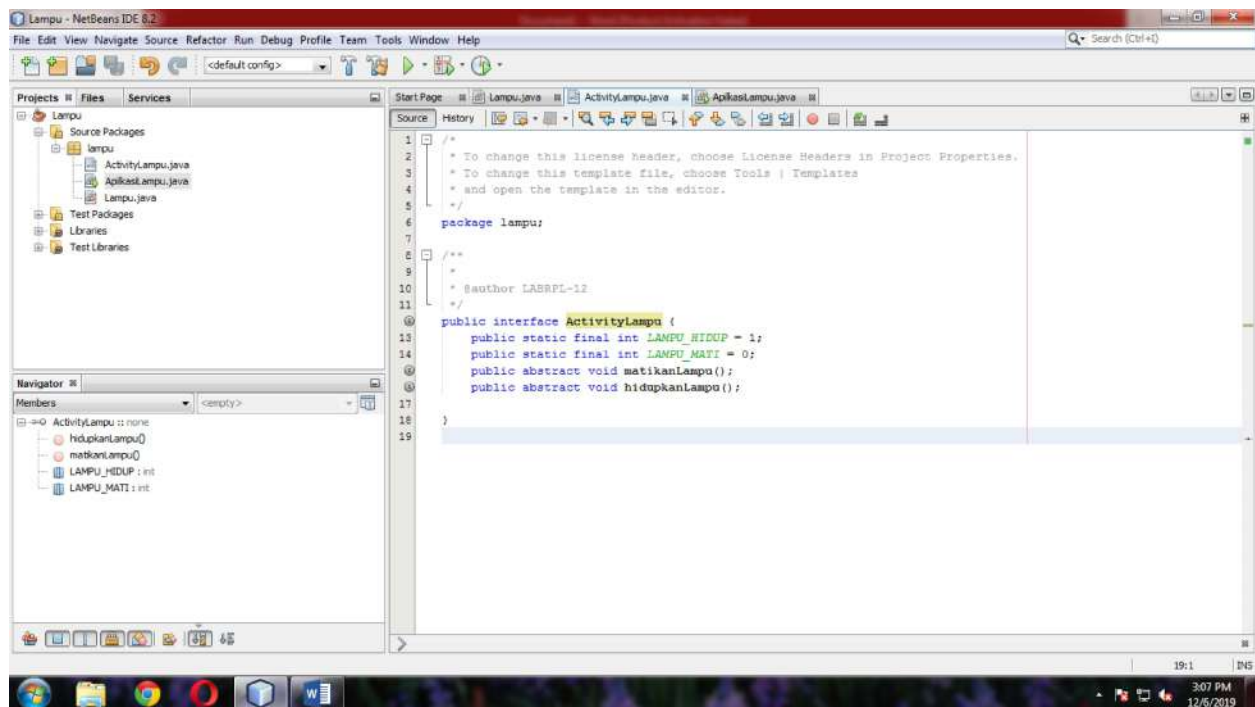
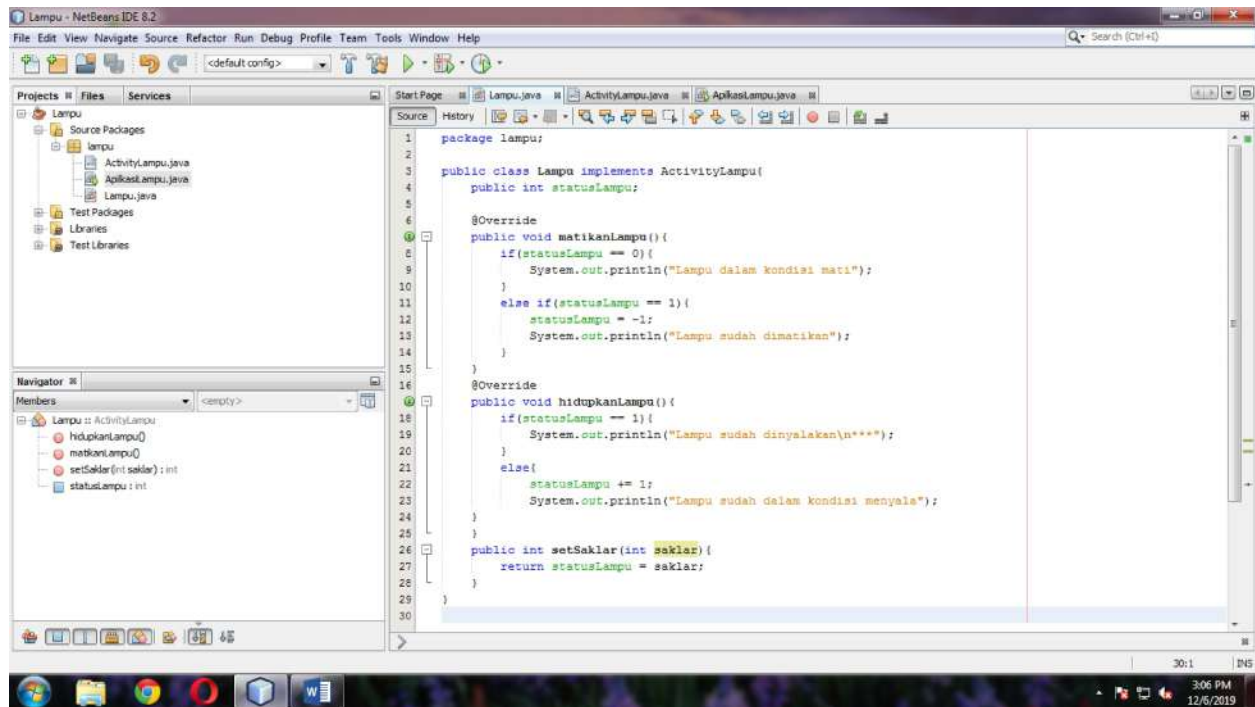
Kelas : B

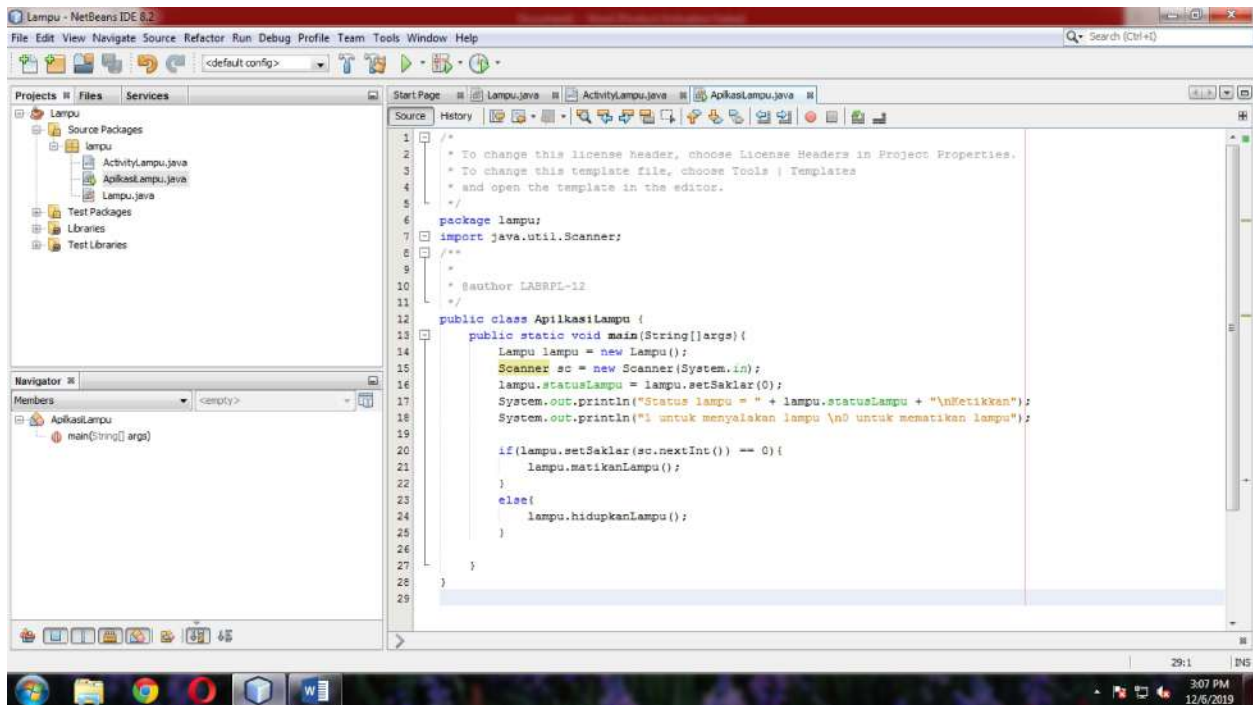
Laporan Modul ke 10

10.2 Implementasi Interface

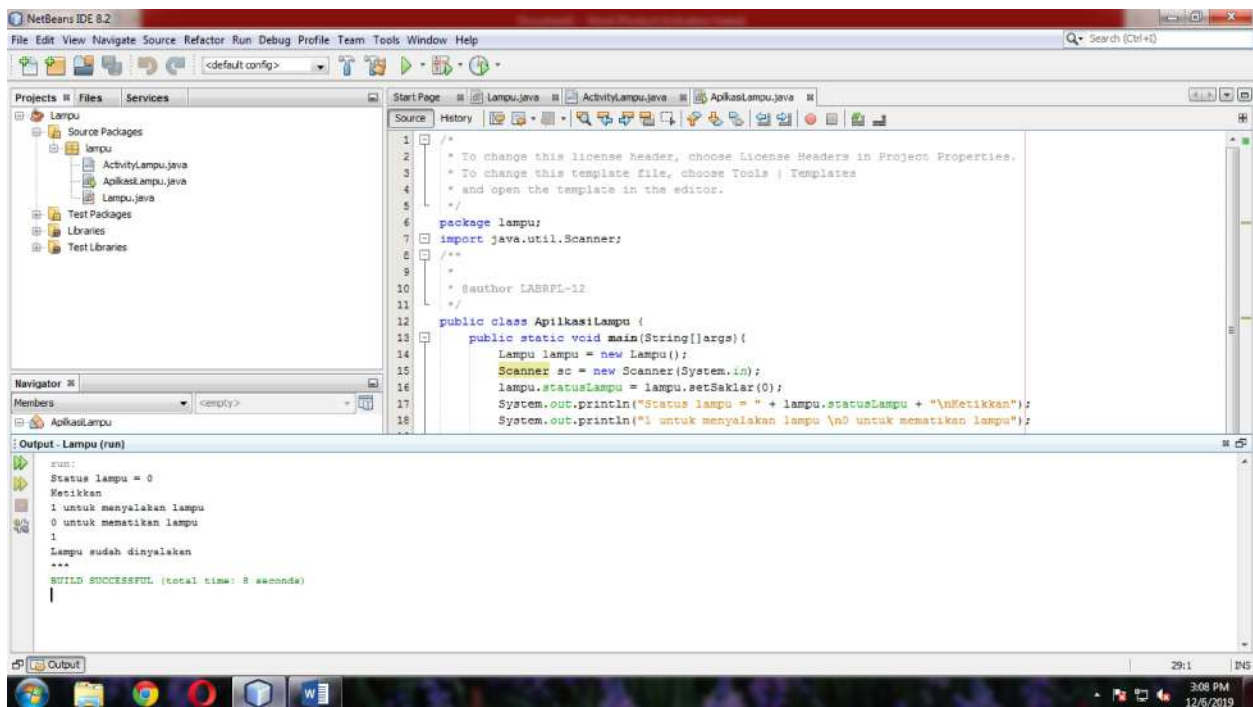


10.2.1 Percobaan

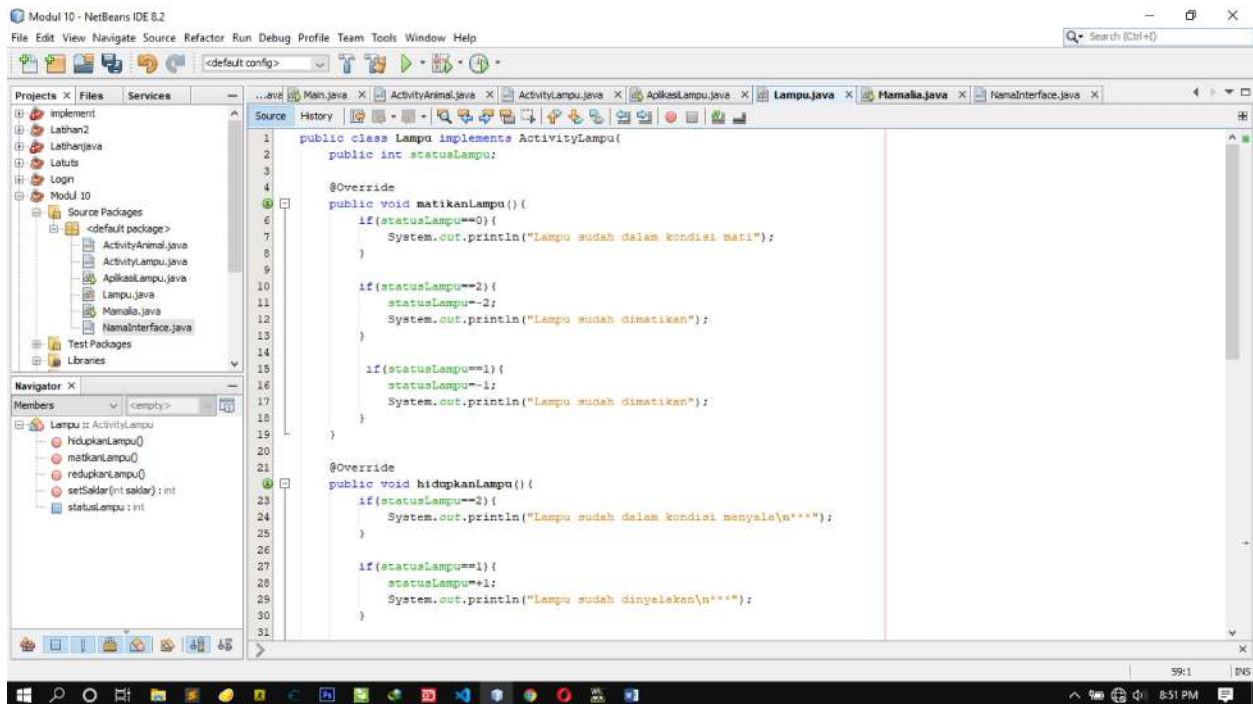
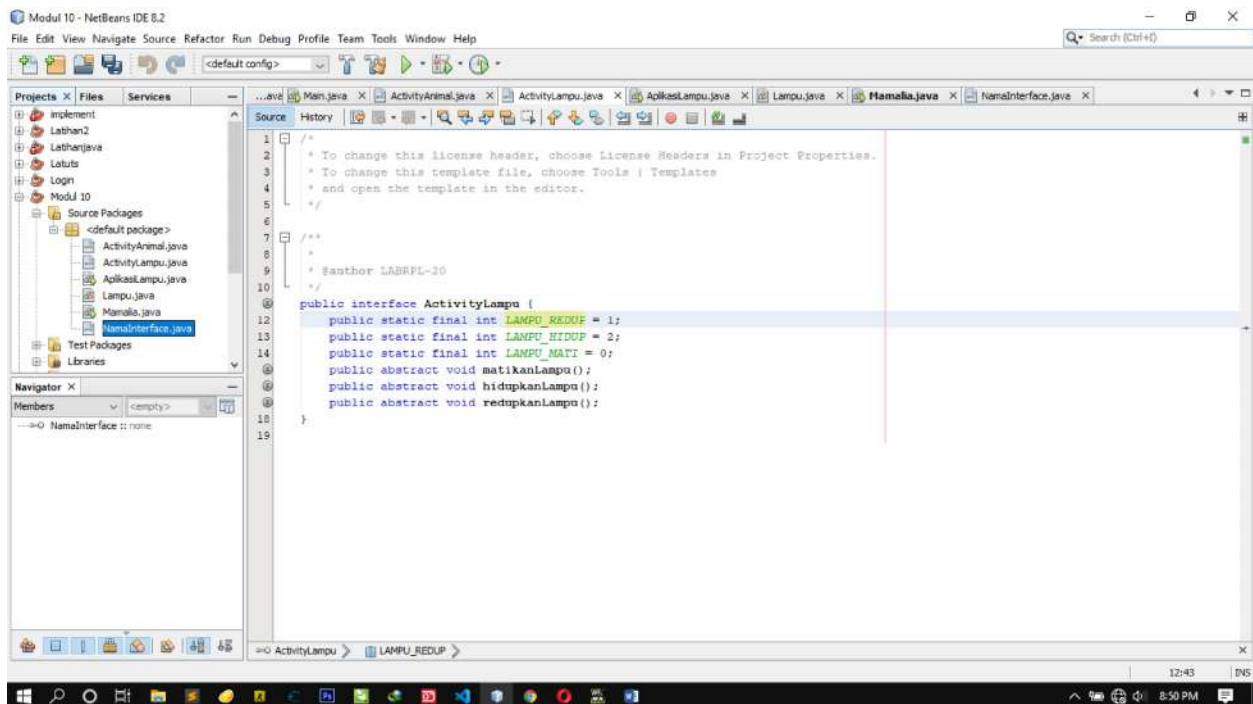




-output



10.3 Tugas



Modul 10 - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+F)

Projects X Files Services

Source Packages

ActivityAnimal.java

ActivityLampu.java

AplikasiLampu.java

Lampu.java

Mamalia.java

NamaInterface.java

Test Packages

Libraries

Navigator X

Members

Lampu: ActivityLampu

hidupkanLampu()

matikanLampu()

redupkanLampu()

setSaklar(int saklar): int

statusLampu: int

```
26
27     if(statusLampu==1){
28         statusLampu++;
29         System.out.println("Lampu sudah dinyalakan\n***");
30     }
31
32     if(statusLampu==0){
33         statusLampu++;
34         System.out.println("Lampu sudah dinyalakan");
35     }
36
37 }
38
39 @Override
40 public void redupkanLampu() {
41     if(statusLampu==1){
42         System.out.println("Lampu sudah dalam kondisi redup");
43     }
44
45     if(statusLampu==0){
46         System.out.println("Lampu dalam kondisi mati");
47     }
48
49     if(statusLampu==2){
50         statusLampu--;
51         System.out.println("Lampu sudah diredupkan");
52     }
53 }
54
55 public int setSaklar(int saklar){
56     return statusLampu = saklar;
57 }
```

59:1 DNS

Modul 10 - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+F)

Projects X Files Services

Source Packages

ActivityAnimal.java

ActivityLampu.java

AplikasiLampu.java

Lampu.java

Mamalia.java

NamaInterface.java

Test Packages

Libraries

Navigator X

Members

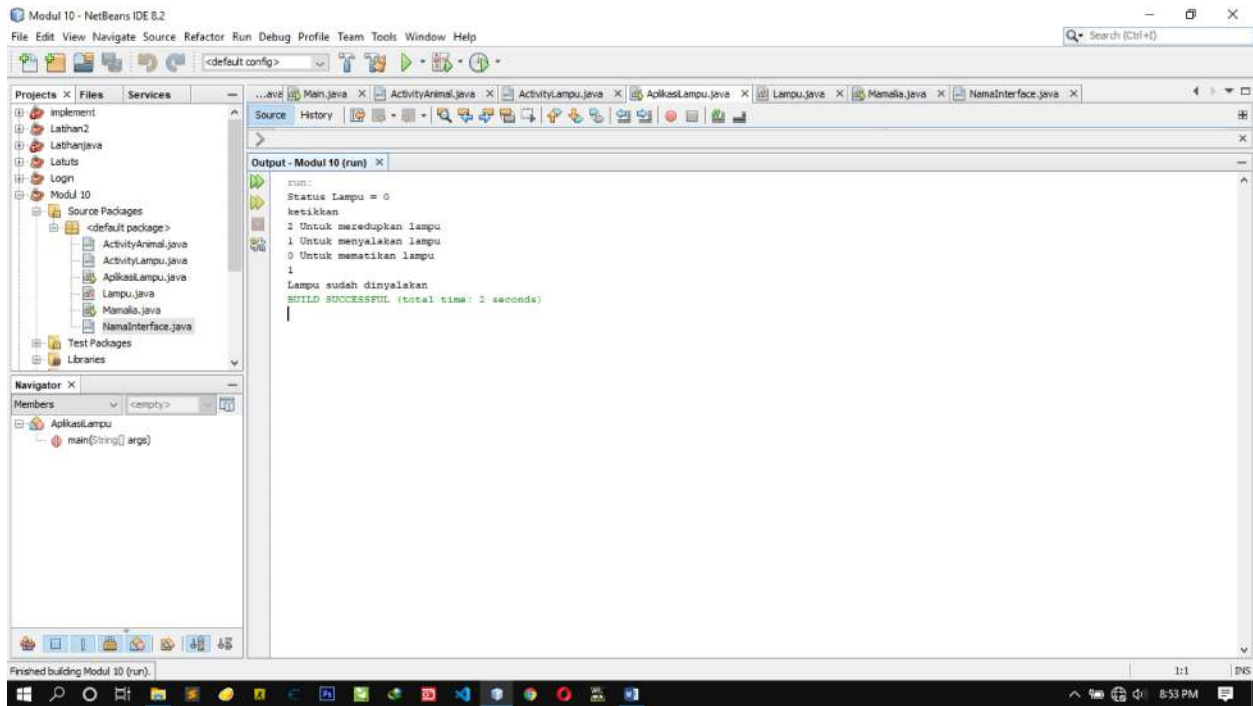
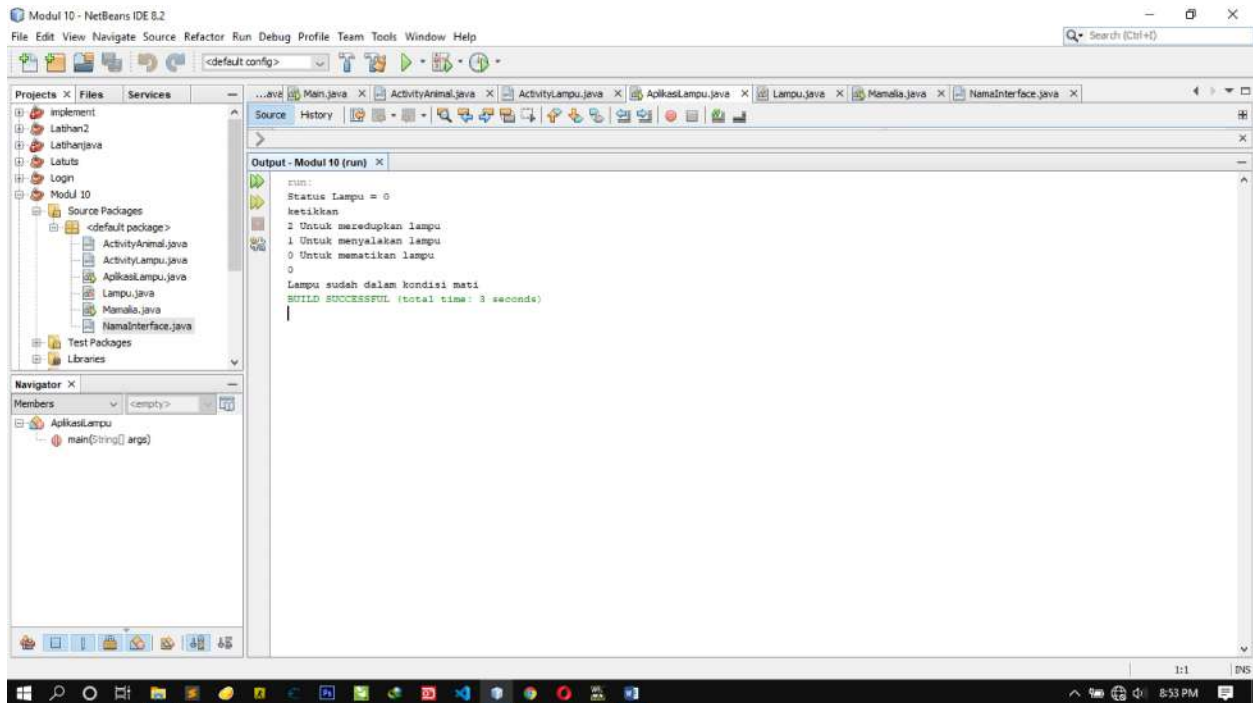
AplikasiLampu

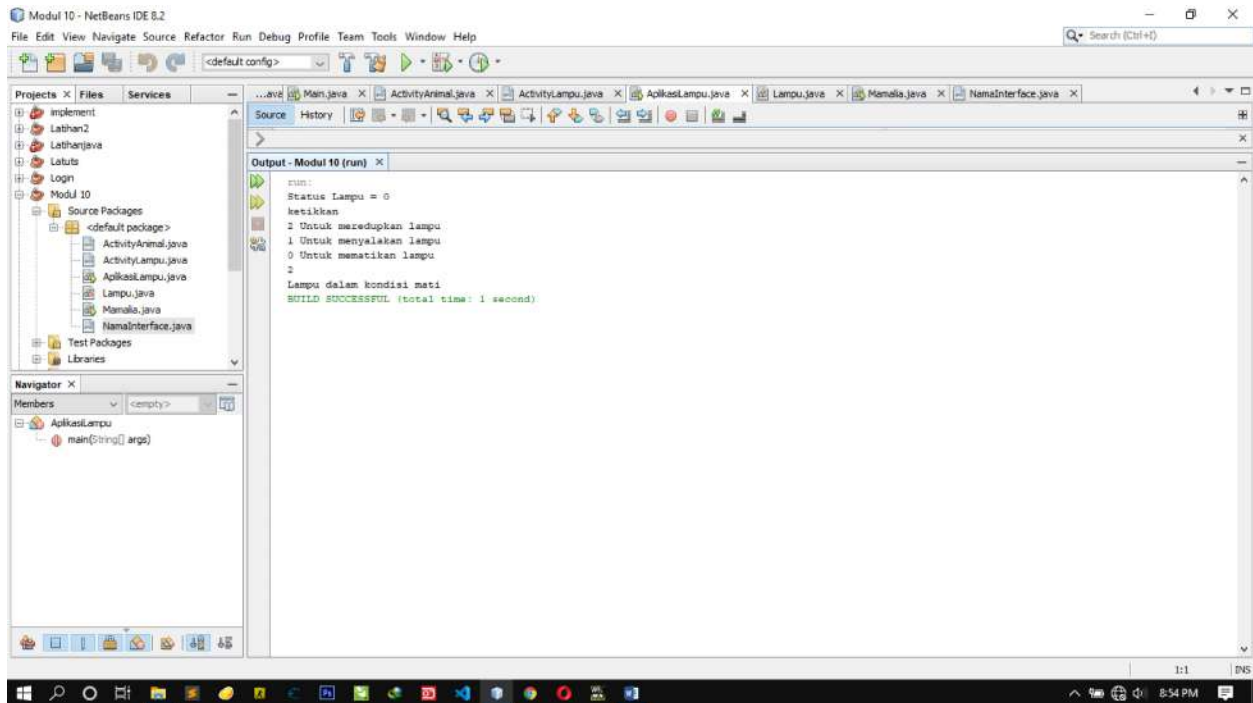
main(String[] args)

```
10
11 import java.util.Scanner;
12
13 public class AplikasiLampu {
14     public static void main(String[] args){
15         Lampu lampu = new Lampu();
16         Scanner sc = new Scanner(System.in);
17         lampu.statusLampu = lampu.setSaklar(0);
18
19         System.out.println("Status Lampu = " + lampu.statusLampu + "\nketikkan");
20         System.out.println("1 Untuk meredupkan lampu\n2 Untuk menyalakan lampu\n0 Untuk mematikan lampu");
21
22         int userInput = sc.nextInt();
23
24         if(userInput==0){
25             lampu.matikanLampu();
26         }
27
28         if(userInput==2){
29             lampu.redupkanLampu();
30         }
31
32         if(userInput==1){
33             lampu.hidupkanLampu();
34         }
35     }
36 }
37 }
```

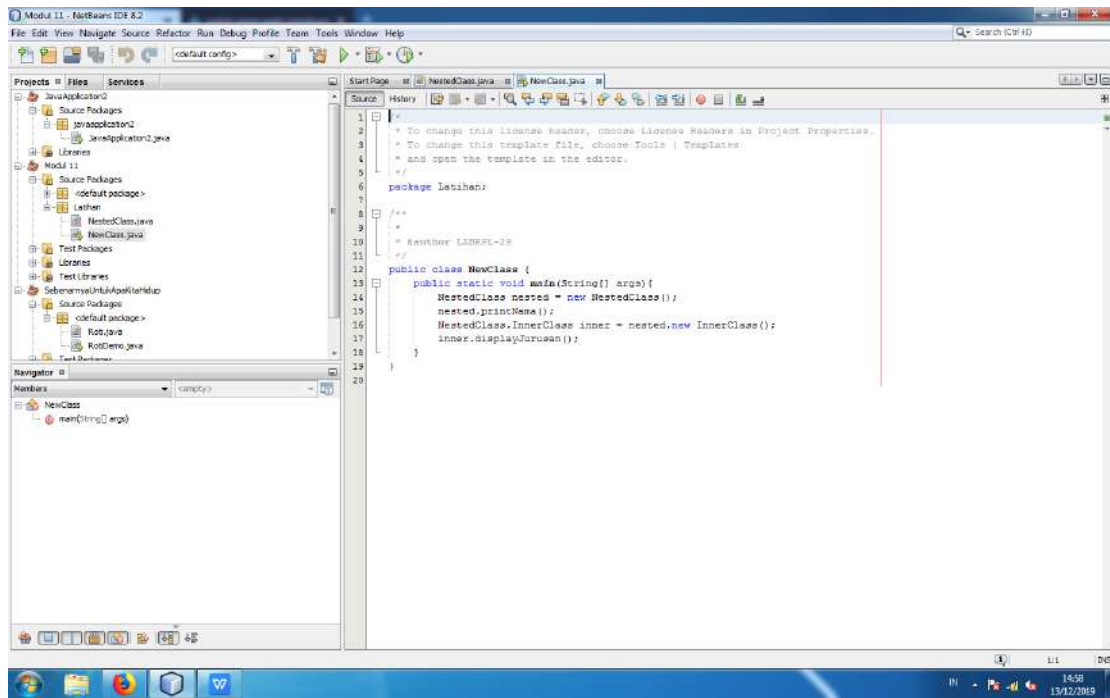
1:1 DNS

-output

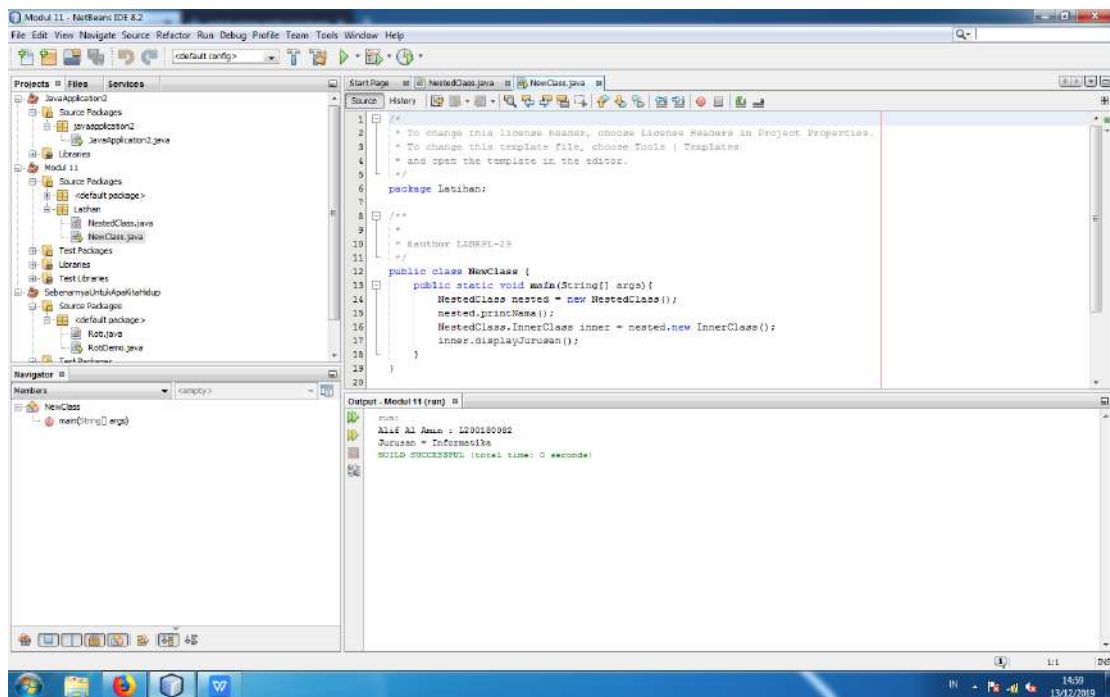




3. Membuat class dengan fungsi main() untuk menampilkan hasil code



- Output



Nama : Alif Al Amin

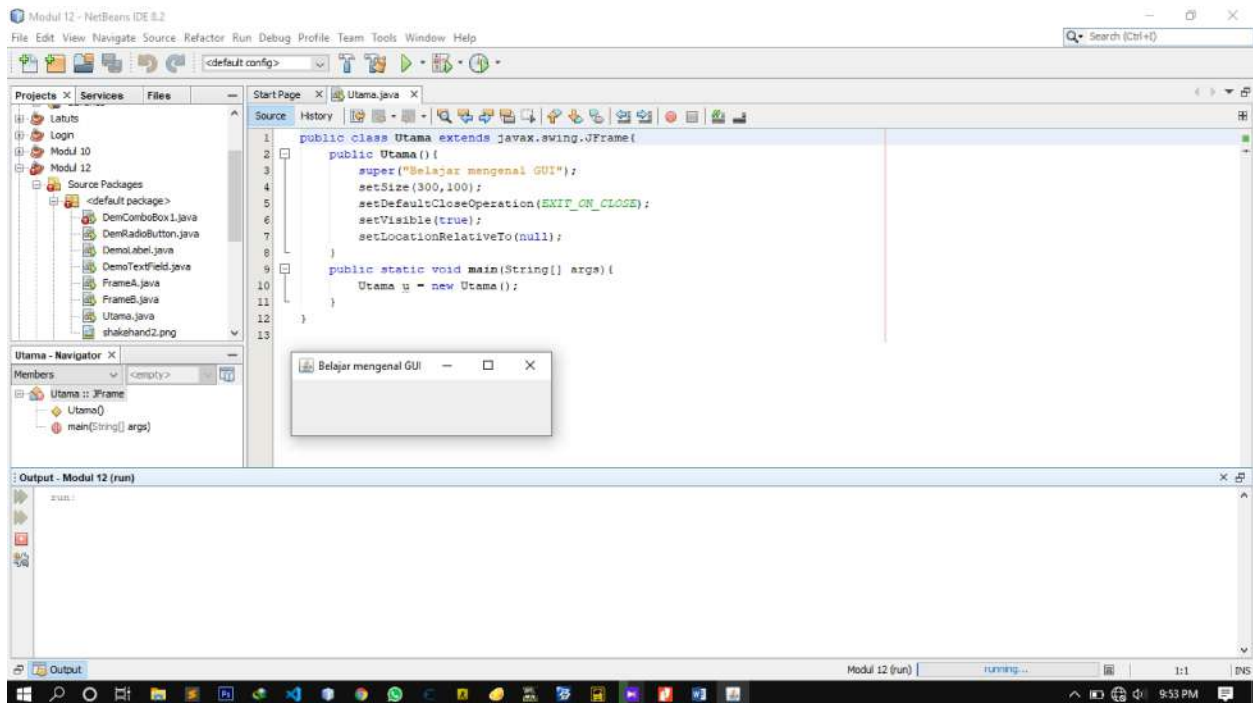
NIM : L200180082

Kelas : B

Laporan Modul ke 12

12.5. Latihan

12.5.1 Frame



Konstruktor

`JFrame()`
`JFrame(String Judul)`

Metode

`void setSize(int lebar, int tinggi)`

`void setLocation(int x, int y)`

`void setVisible(Boolean)`

`void setLocationRelativeTo(Component)`

Keterangan

Membuat JFrame tanpa judul
Membuat JFrame dengan judul

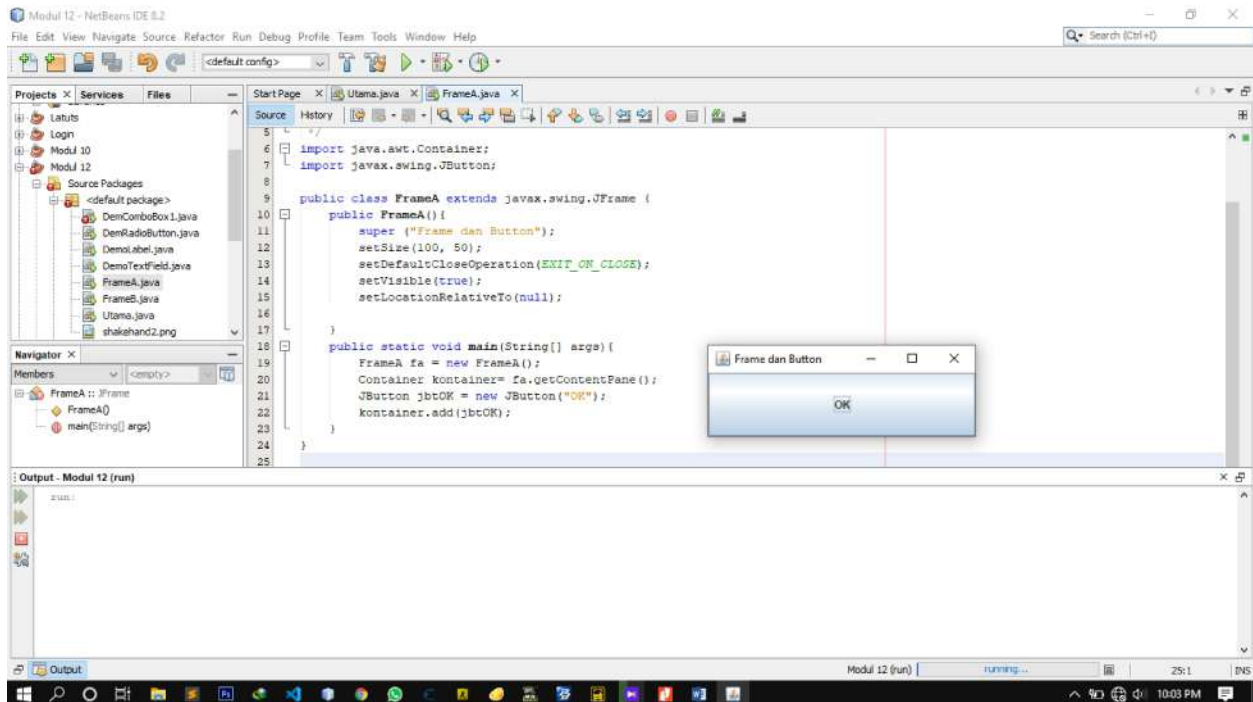
Keterangan

Menentukan ukuran frame

Menentukan lokasi frame dihitung dari kiri atas
Menentukan JFrame ditampilkan atau tidak

Menentukan letak JFrame relative dengan komponen lainnya. Jika diset null, maka JFrame akan ditampilkan di tengah.

12.5.2. Button



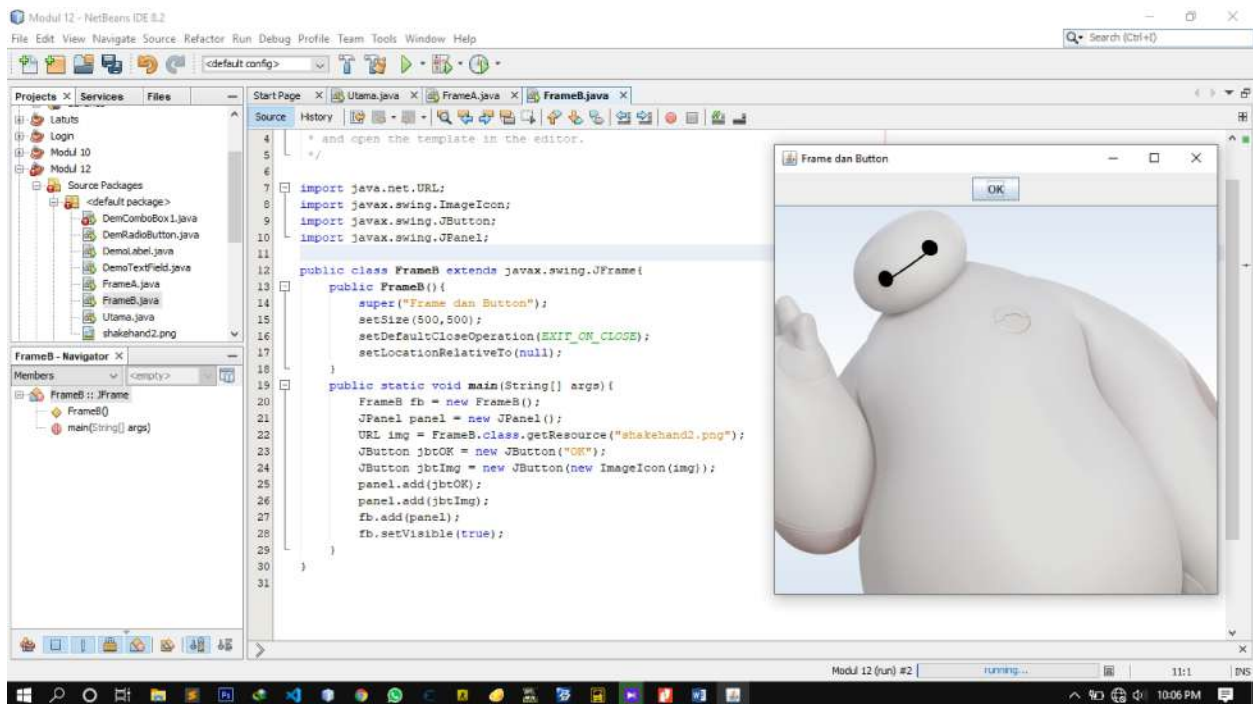
Konstruktor

`JButton()`
`JButton(String teks)`
`JButton(Icon icon)`
`JButton(String teks, Icon icon)`

Keterangan

Membuat `JButton` tanpa teks maupun icon
Membuat `JButton` dengan teks
Membuat `JButton` dengan icon
Membuat `JButton` dengan teks dan icon

12.5.3. Container



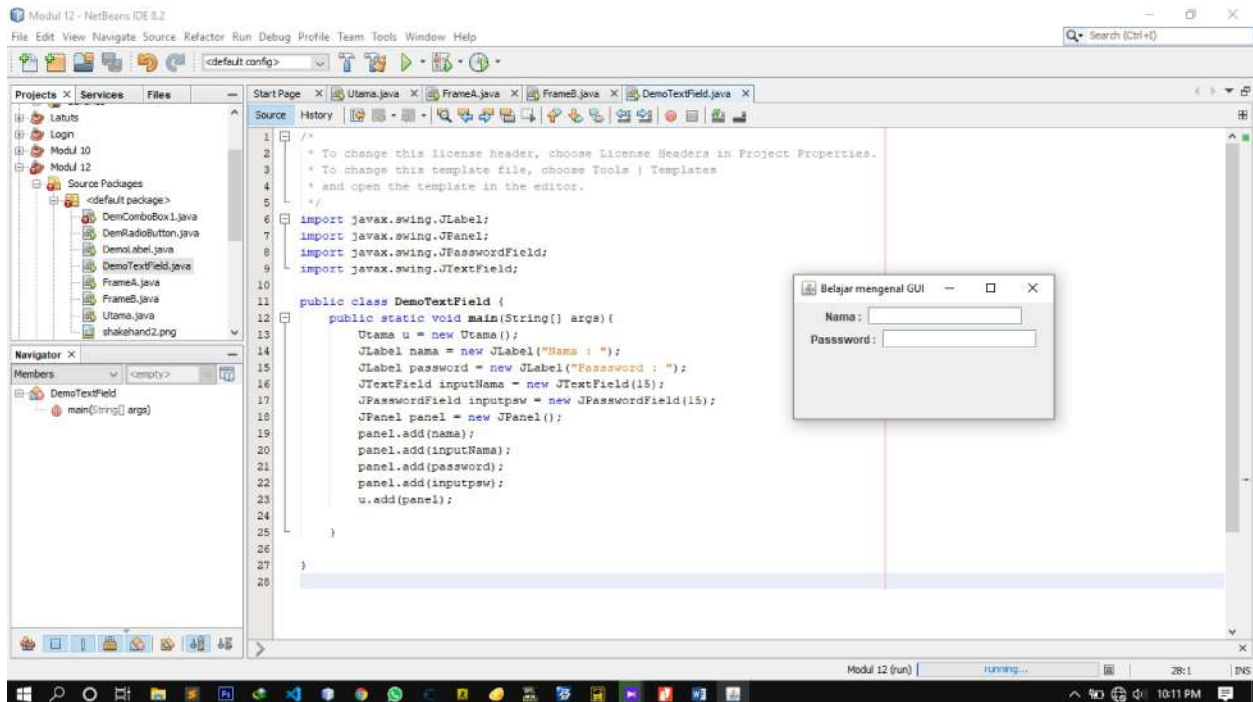
Konstruktor

JLabel (String teks)
JLabel (String teks,
int i)
JLabel (String teks,
Icon ic, int i)

Keterangan

Label dengan teks
Label dengan teks dan alignment (perataan).
Terdapat tiga jenis alignment : LEFT,
CENTER, dan RIGHT
Label dengan teks, icon, dan alignment

12.5.5. TextField dan PasswordField



Konstruktor

JTextField()
 JTextField(int i)
 JTextField(String i)
 JTextField(String teks, int i)

Penjelasan

Text field kosong tanpa tulisan
 Text field dengan panjang yang ditentukan
 Text field dengan teks yang sudah ditentukan
 Text field dengan teks dan Panjang kolom yang sudah ditentukan

Parameter dalam class JTextField

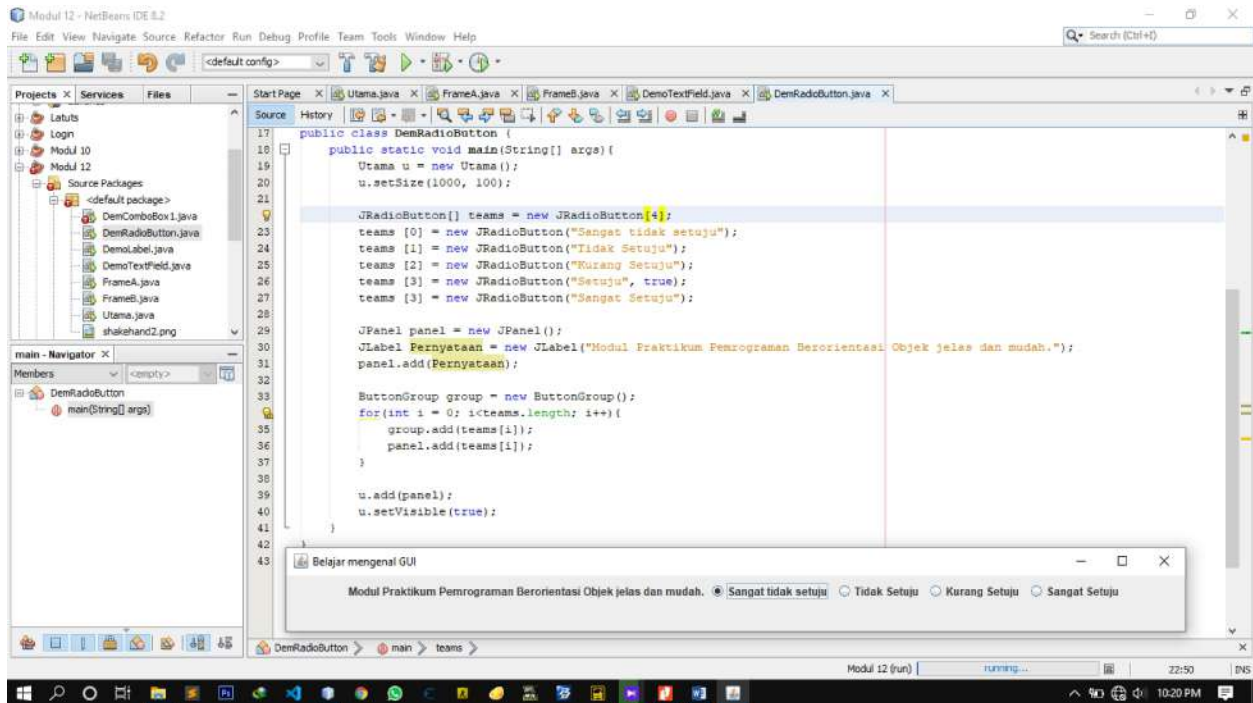
String Text
 Boolean Editable

 Int columns
 Int horizontalAlignment

Penjelasan

Teks dalam Text field
 Menentukan bisa tidaknya teks dalam Text field untuk diedit
 Jumlah kolom pada Text field
 Perataan horizontal pada Text field

12.5.6. Radio Button dan CheckBox



Konstruktor

JCheckBox(String teks)
JCheckBox(String,
Boolean)

JCheckBox(icon)
JCheckBox(icon,
Boolean)
JCheckBox(String,
Icon)
JCheckBox(String,
Icon, Boolean)

Penjelasan

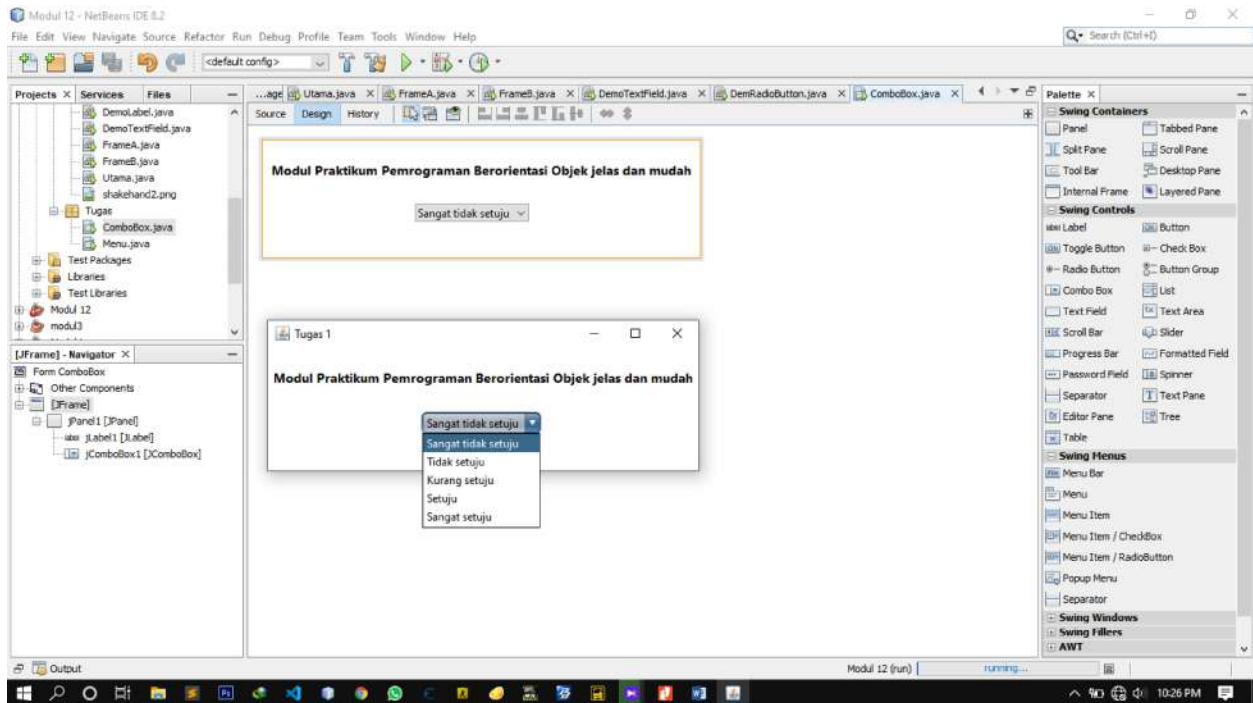
Check box dengan teks yang ditentukan
Check box dengan teks yang ditentukan. jika
kondisi pada parameter kedua true, maka Check
box ini akan dipilih

Check box dengan gambar icon
Check box dengan gambar icon dan terpilih
jika true
Check box dengan tulisan dan gambar icon

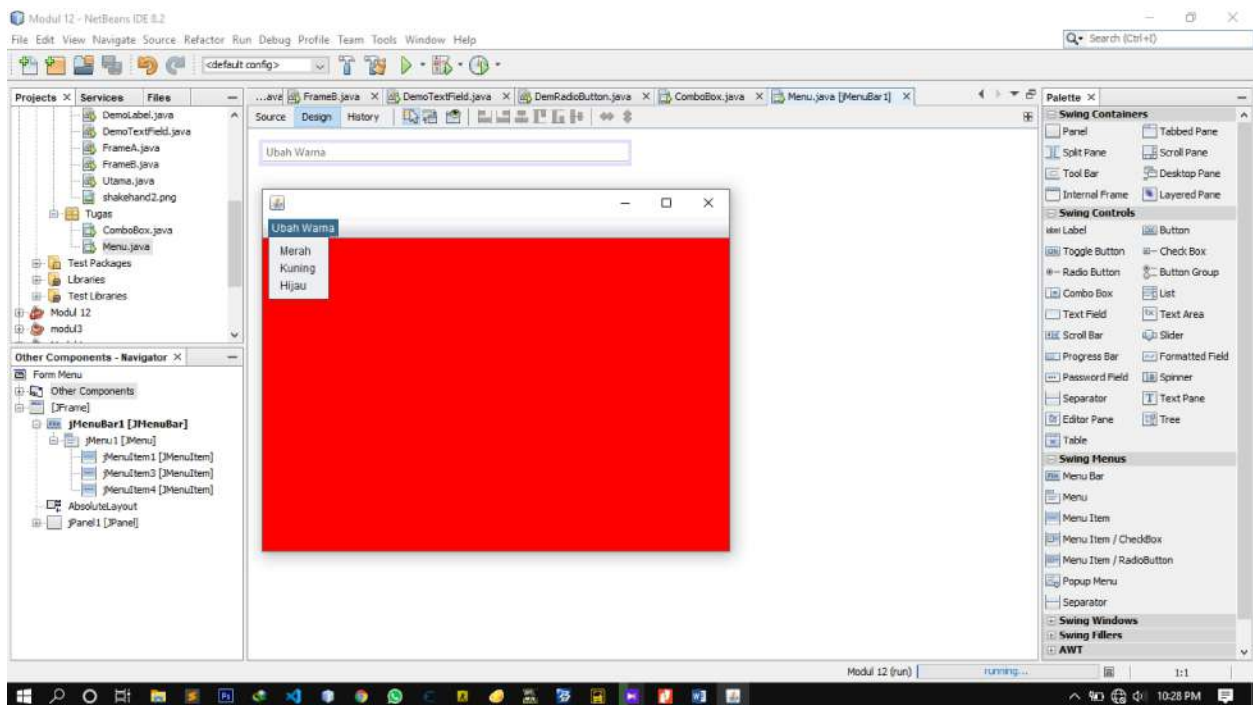
Check box dengan tulisan dan gambar icon,
akan dipilih secara default jika kondisi true.

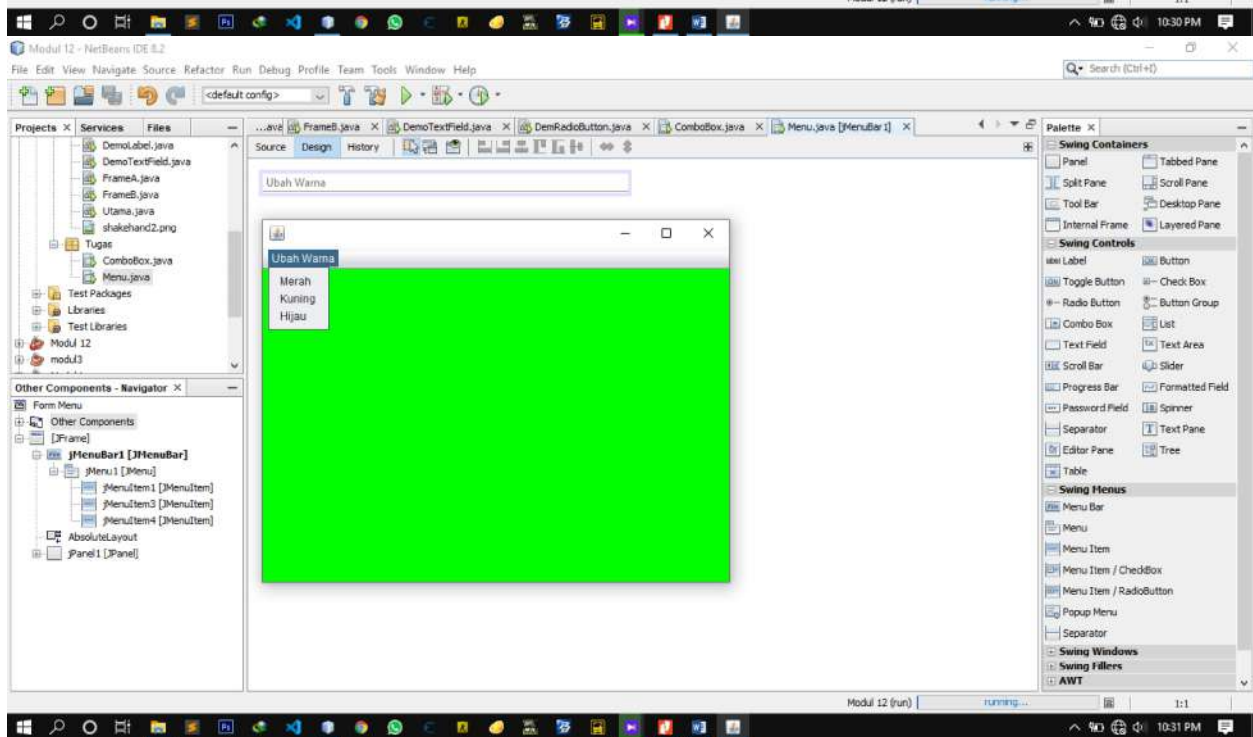
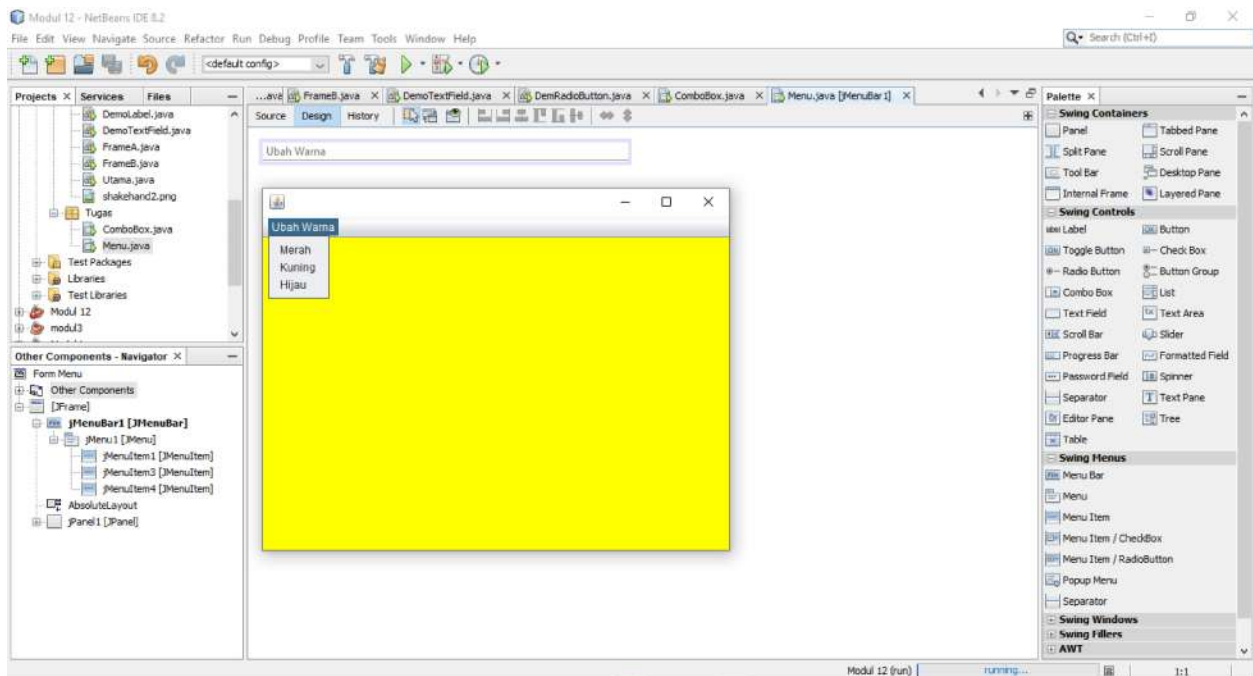
12.6. Tugas

1. JComboBox (drag and drop)



2. Menu





Code pada menu :

