

A dark blue vertical bar on the left side of the page, with a blue arrow pointing right towards the title.

Implementasi Kelas Dan Objek

Pertemuan 2 KSP JAVA 2022

Several thin, curved lines in dark blue and light gray originating from the bottom left corner.

KSP JAVA 2023

INFORMATIKA 2023 | UNIVERSITAS ATMA JAYA YOGYAKARTA

Java



Java adalah salah satu bahasa pemrograman populer dan biasanya digunakan untuk mengembangkan aplikasi mobile, aplikasi web, aplikasi desktop, game, dsb.

JDK



JDK atau Java Development Kit adalah kumpulan komponen (tools & libraries) yang diperlukan untuk pengembangan aplikasi dengan bahasa pemrograman Java.

Apache NetBeans & BlueJ



Apache NetBeans & BlueJ adalah IDE yang digunakan untuk mengembangkan aplikasi dalam bahasa Java. BlueJ lebih ditujukan untuk orang yang baru mulai belajar OOP.

OOP



Salah satu konsep yang perlu dikuasai sebelum memulai pemrograman menggunakan bahasa Java adalah OOP (Object Oriented Programming) yang jika dalam bahasa Indonesia adalah PBO (Pemrograman Berbasis Objek).

OOP/PBO adalah paradigma pemrograman menggunakan objek. Nantinya sebuah objek akan memiliki atribut/state/variable dan operasi/method/perilaku/behavior.

Kelas

Kelas adalah cetakan atau blueprint untuk membuat objek. Berikut contoh kelas pada Java :

```
public class Mahasiswa {  
    private String nama;  
    private String npm;  
    private double ipk;  
  
    public Mahasiswa() {  
  
    }  
  
    public Mahasiswa(String nama, String npm, double ipk) {  
        this.nama = nama;  
        this.npm = npm;  
        this.ipk = ipk;  
    }  
  
    public void printData(){  
        System.out.println("Nama : "+nama);  
        System.out.println("NPM : "+npm);  
        System.out.println("IPK : "+ipk);  
    }  
}
```

pada contoh di atas adalah sebuah class Mahasiswa dengan 3 atribut yaitu **nama**, **npm**, dan **ipk**. Kelas tersebut juga memiliki sebuah method yaitu **printData**. Aturan penulisan kelas adalah harus diawali dengan huruf kapital dan nanti nama kelasnya harus sama dengan nama file kelasnya (Mahasiswa.java).

Objek

Objek adalah perwujudan atau instance dari sebuah kelas. Berikut contoh pembuatan Objek pada Java :

```
public static void main(String[] args) {  
    // TODO code application logic here  
    Mahasiswa M1 = new Mahasiswa();  
    Mahasiswa M2 = new Mahasiswa("Ricky", "200710700", 3.6);  
}
```

pembuatan instance baru dari sebuah kelas menggunakan keyword new. Pertama awali dengan penulisan nama kelas (Mahasiswa) kemudian nama objek (m1) lalu assignment operator (=) kemudian keyword new lalu konstruktor dan parameternya.

Method

Method adalah aksi yang bisa dilakukan oleh sebuah objek. Method bisa berupa prosedur ataupun fungsi. Nilai balikan pada fungsi bisa berupa kelas lain yang sudah dibuat. Berikut contoh method :

```
public void printData() {
    System.out.println("Nama : "+nama);
    System.out.println("NPM : "+npm);
    System.out.println("IPK : "+ipk);
}

public static void main(String[] args) {
    // TODO code application logic here
    Mahasiswa M1 = new Mahasiswa();
    Mahasiswa M2 = new Mahasiswa("Ricky", "200710700", 3.6);

    M2.printData();
}
```

pada contoh di atas kelas Mahasiswa memiliki method printData. Pada contoh tersebut printData adalah method yang digunakan untuk menampilkan data yang tersimpan dalam atribut yang dimiliki kelas Mahasiswa dan untuk mengakses dan menggunakan method tersebut bisa dengan cara `namaObjek.namaMethod()` Dengan contoh di atas berarti `M2.printData`.

Konstruktor

Konstruktor adalah sebuah method khusus yang digunakan untuk menginisialisasi objek. Konstruktor dipanggil saat pembuatan sebuah objek baru. Konstruktor harus memiliki nama yang sama dengan kelasnya, tidak boleh ada nilai balikan, dan sebuah kelas bisa memiliki lebih dari 1 konstruktor. Jika kita tidak menuliskan konstruktor secara eksplisit maka compiler akan membuat kita sebuah default konstruktor yang memiliki parameter Kosong (tidak memiliki argument). Berikut contoh konstruktor :

```
public Mahasiswa() {
}

public Mahasiswa(String nama, String npm, double ipk) {
    this.nama = nama;
    this.npm = npm;
    this.ipk = ipk;
}
```

agar menghemat penggunaan nama, maka gunakan keyword `this` pada konstruktor. Keyword `this` digunakan untuk merujuk kepada objek itu sendiri sehingga compiler tidak akan bingung jika ada variabel dengan nama yang sama.

Modifier

Modifier adalah keyword yang berguna untuk menambahkan sifat kepada kelas, atribut, atau method. Terdapat 2 jenis modifier di Java yaitu Access modifier dan Non-Access modifier. Access modifier digunakan untuk mengatur level aksesibilitas/visibilitas dari sebuah kelas, atribut, method, dan konstruktor. Beberapa contoh access modifier :

1. Public : bisa diakses oleh kelas lain
2. Protected : bisa diakses oleh kelas turunan
3. Private : tidak bisa diakses oleh kelas lain

Untuk mengakses atribut yang memiliki visibilitas private dari kelas lain maka dibutuhkan method dengan visibilitas public yang biasanya disebut accessor dan mutator. Accessor digunakan untuk mengembalikan nilai dari sebuah atribut. Mutator digunakan untuk mengubah nilai dari sebuah atribut. Contohnya sebagai berikut :

```
public Mahasiswa(String nama, String npm, double ipk) {
    this.nama = nama;
    this.npm = npm;
    this.ipk = ipk;
}

public String getNama() {
    return nama;
}

public void setNama(String nama) {
    this.nama = nama;
}

public String getNpm() {
    return npm;
}

public void setNpm(String npm) {
    this.npm = npm;
}

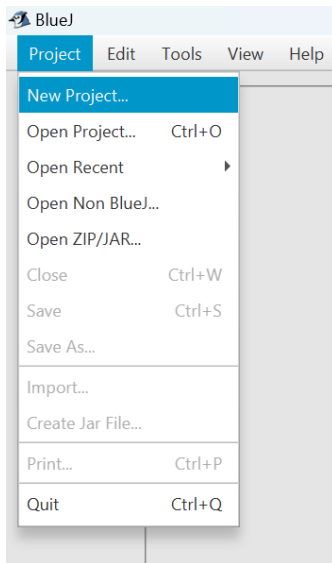
public double getIpk() {
    return ipk;
}

public void setIpk(double ipk) {
    this.ipk = ipk;
}
```

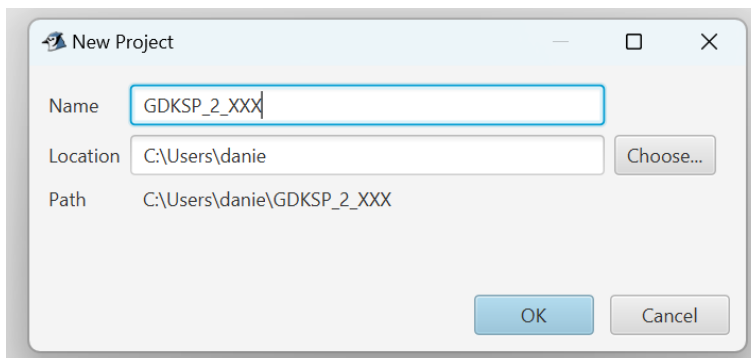
Contoh non-access modifier adalah keyword final yang artinya atribut atau method tersebut tidak bisa dioverride (tidak bisa diganti). Teman-teman bisa mencari sendiri contoh non-access modifier yang lain.

Guided BlueJ

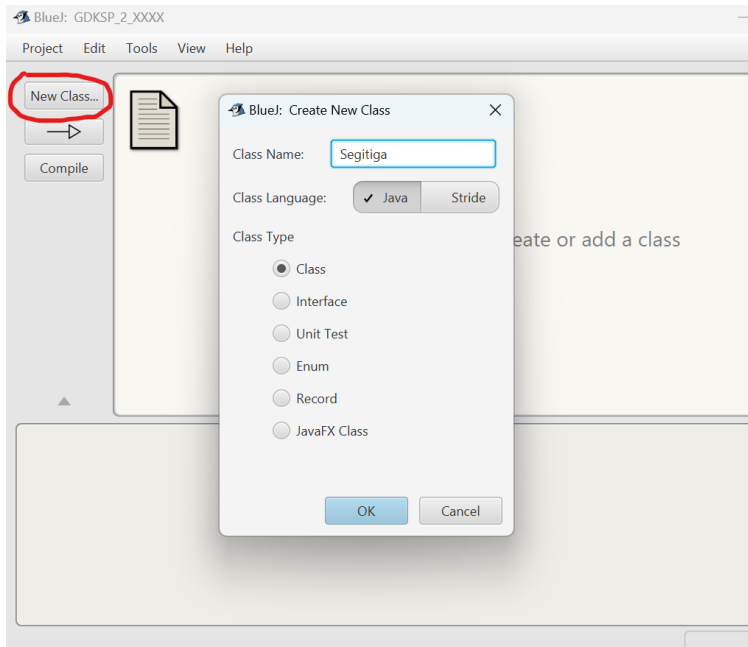
1. Pertama buka BlueJ lalu Pilih “Project” lalu “New Project”



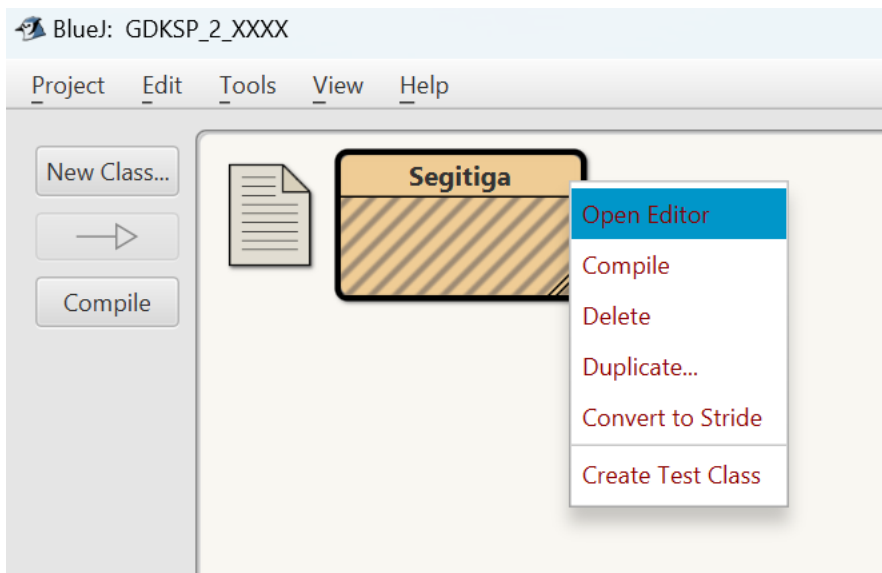
2. Lalu beri nama sesuai dengan gambar dan XXXX adalah 4 digit terakhir npm



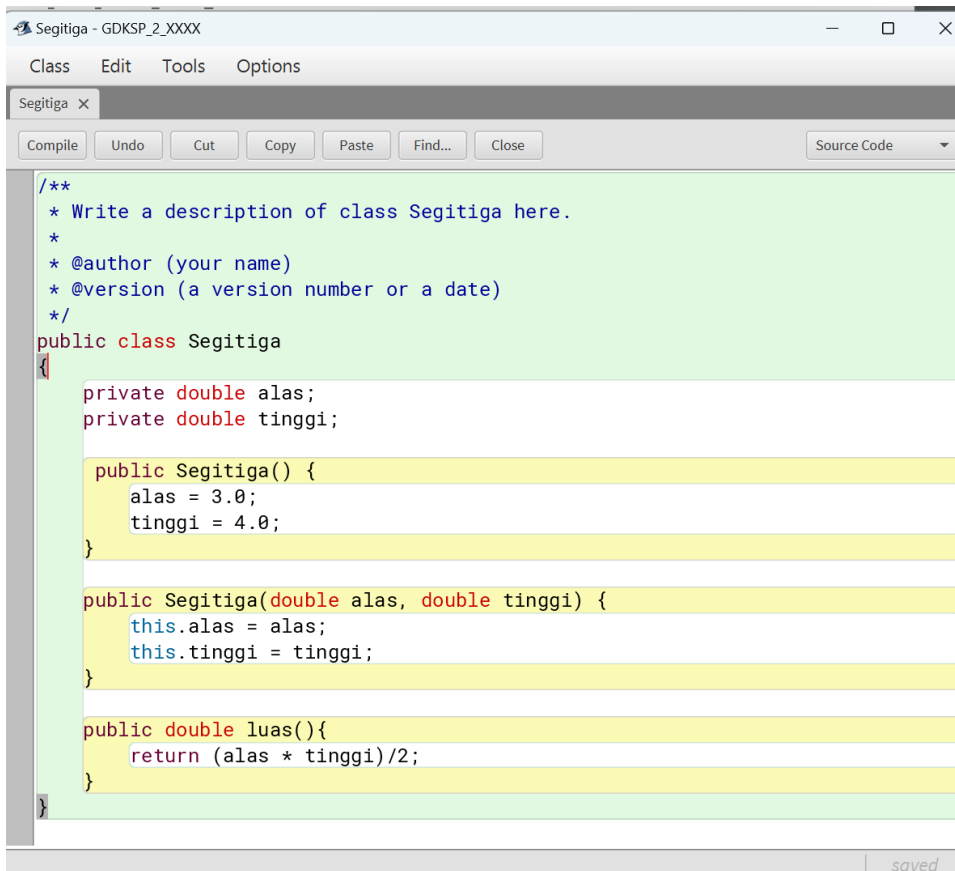
3. Setelah itu buat kelas baru dengan cara klik button “New Class” dan beri nama kelas barunya “Segitiga”. Jika sudah klik “OK”



4. Setelah itu klik kanan kelas baru dibuat tadi dan pilih open editor untuk mulai menuliskan code untuk kelas Segitiga

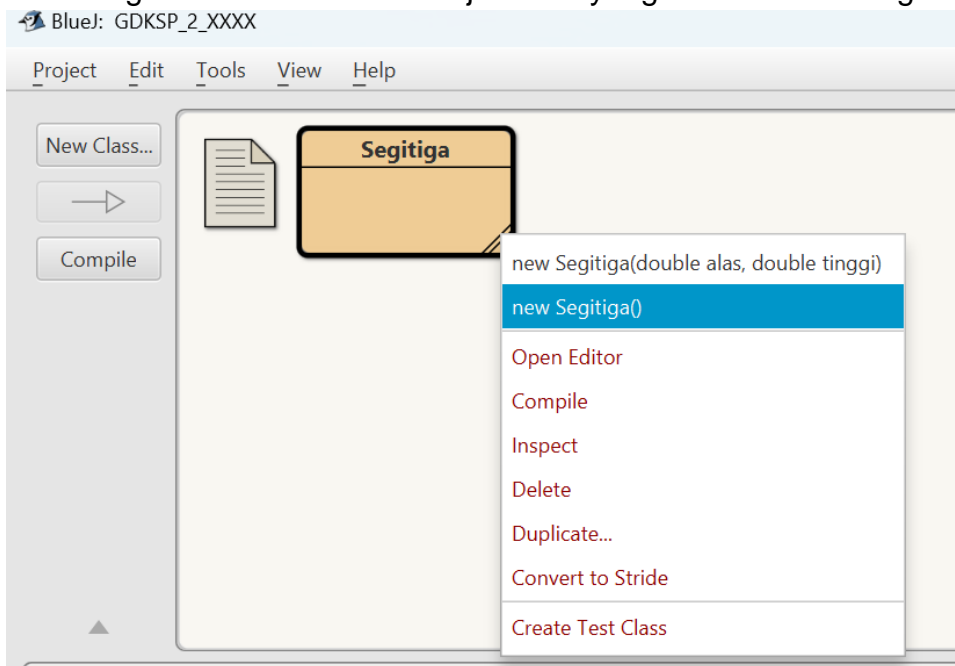


5. Lalu tulislah code berikut:

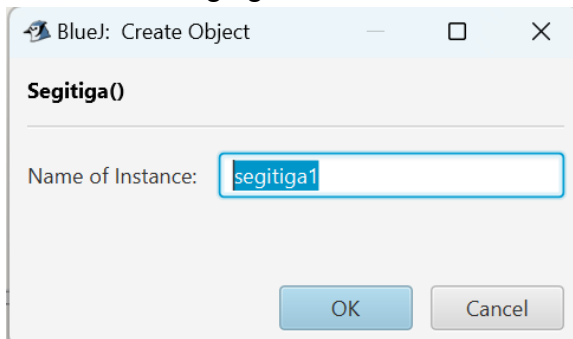


Jika sudah selesai ditulis klik tombol "Compile"

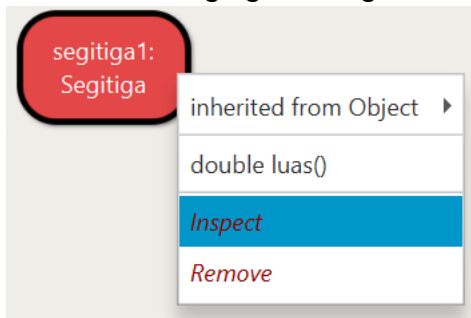
6. Sekarang kita akan membuat objek baru yang tidak memiliki argument.



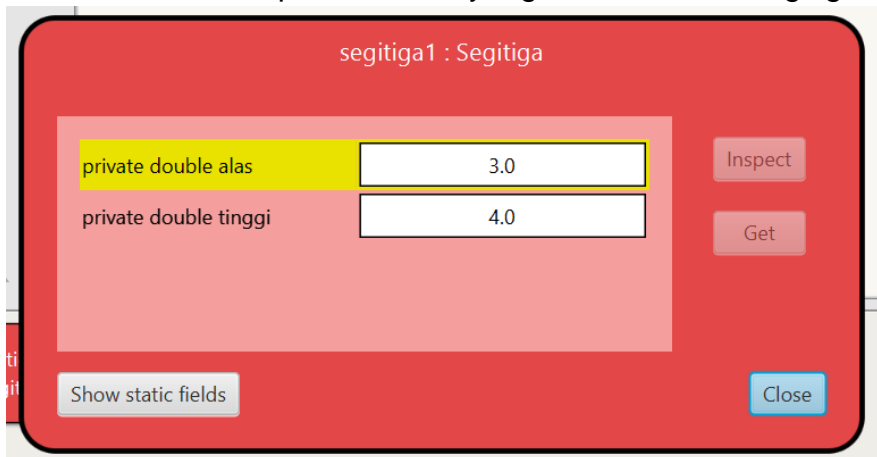
7. Beri nama “segitiga1”



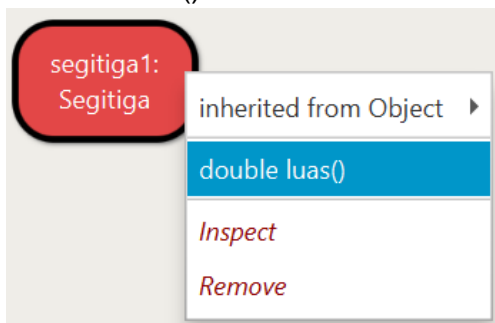
8. Cek atribut segitiga1 dengan klik kanan object lalu “Inspect”



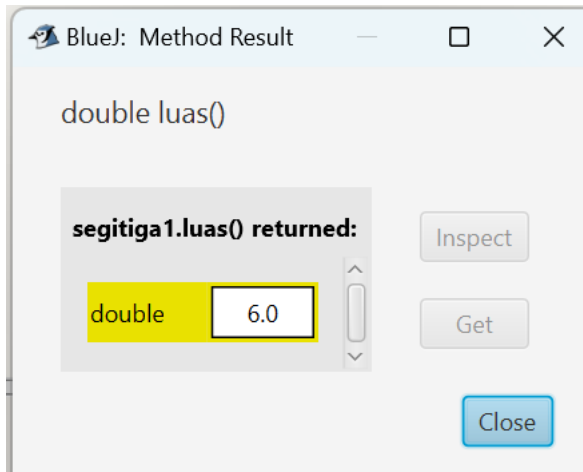
Maka akan menampilkan atribut yang dimiliki atribut segitiga1



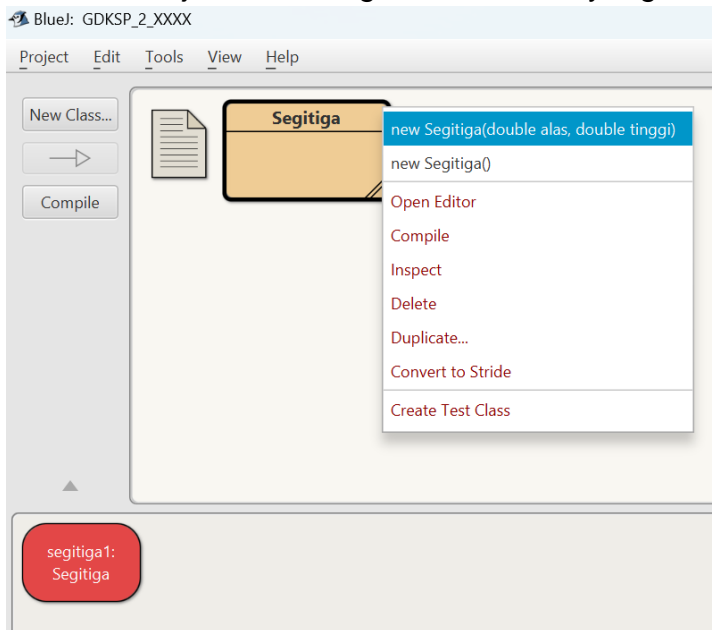
9. Sekarang kita akan mengecek luas Segitiga dengan klik kanan objek lalu pilih “double luas()”



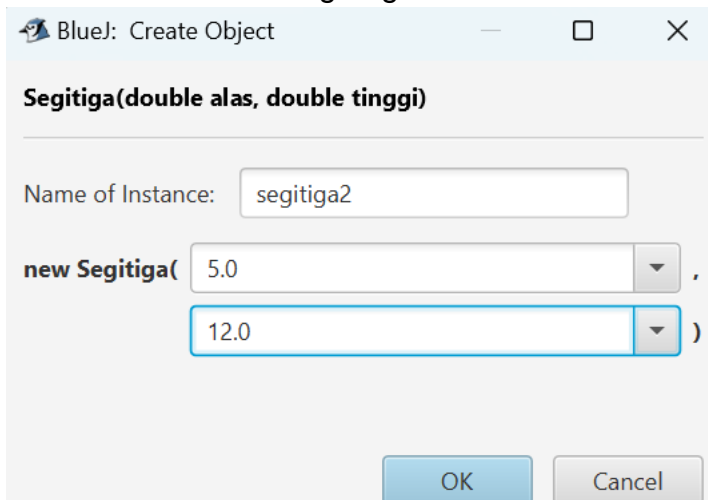
Maka akan menampilkan nilai yang direturnkan oleh method double luas()



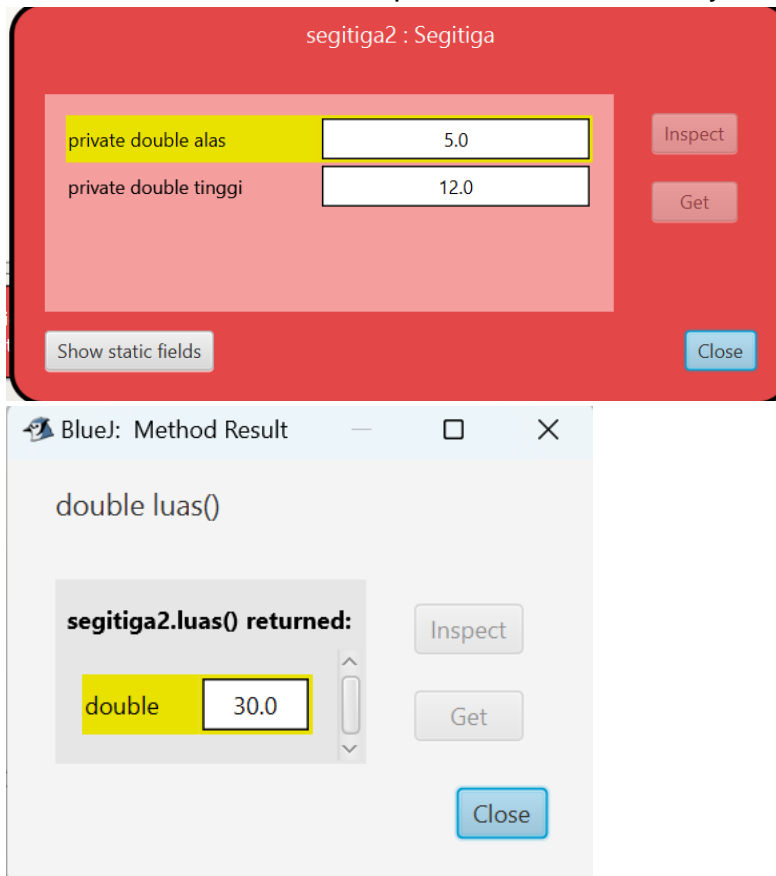
10. Lalu buat objek baru dengan konstruktor yang memiliki argument



Dan isikan sesuai dengan gambar ini:



11. lalu cek atribut dan luas seperti contoh sebelumnya.



12. Setelah itu Kembali ke open editor dan tambahkan accessor dan mutator untuk atribut alas dan tinggi

```
public class Segitiga
{
    private double alas;
    private double tinggi;

    public Segitiga() {
        alas = 3.0;
        tinggi = 4.0;
    }

    public Segitiga(double alas, double tinggi) {
        this.alas = alas;
        this.tinggi = tinggi;
    }

    public double getAlas() {
        return alas;
    }

    public void setAlas(double alas) {
        this.alas = alas;
    }

    public double getTinggi() {
        return tinggi;
    }

    public void setTinggi(double tinggi) {
        this.tinggi = tinggi;
    }

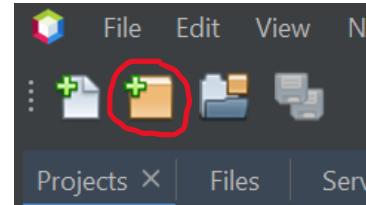
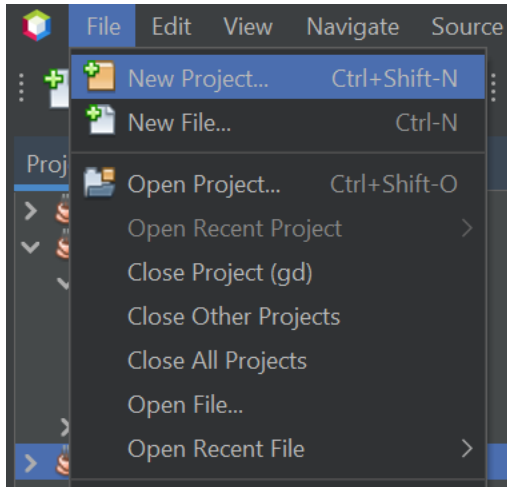
    public double luas(){
        return (alas * tinggi)/2;
    }
}
```

Jika sudah klik “Compile”

13. Setelah itu coba accessor dan mutator dengan membuat objek baru seperti yang sudah dicontohkan tadi. Accessor digunakan untuk mengembalikan nilai atribut dan mutator digunakan untuk mengubah nilai atribut.

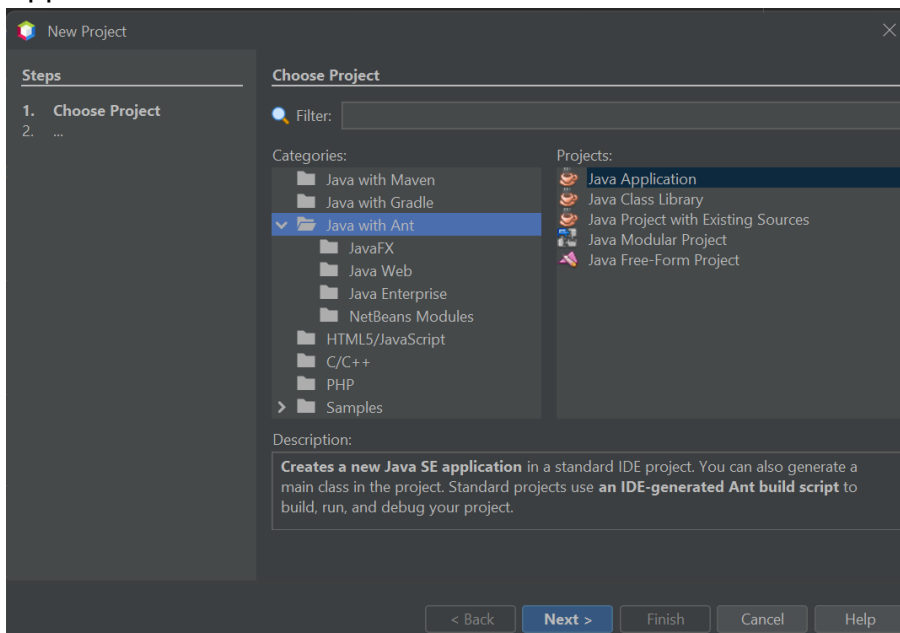
Guided NetBean

1. Buka NetBean setelah itu pilih “file” dan klik “New Project”

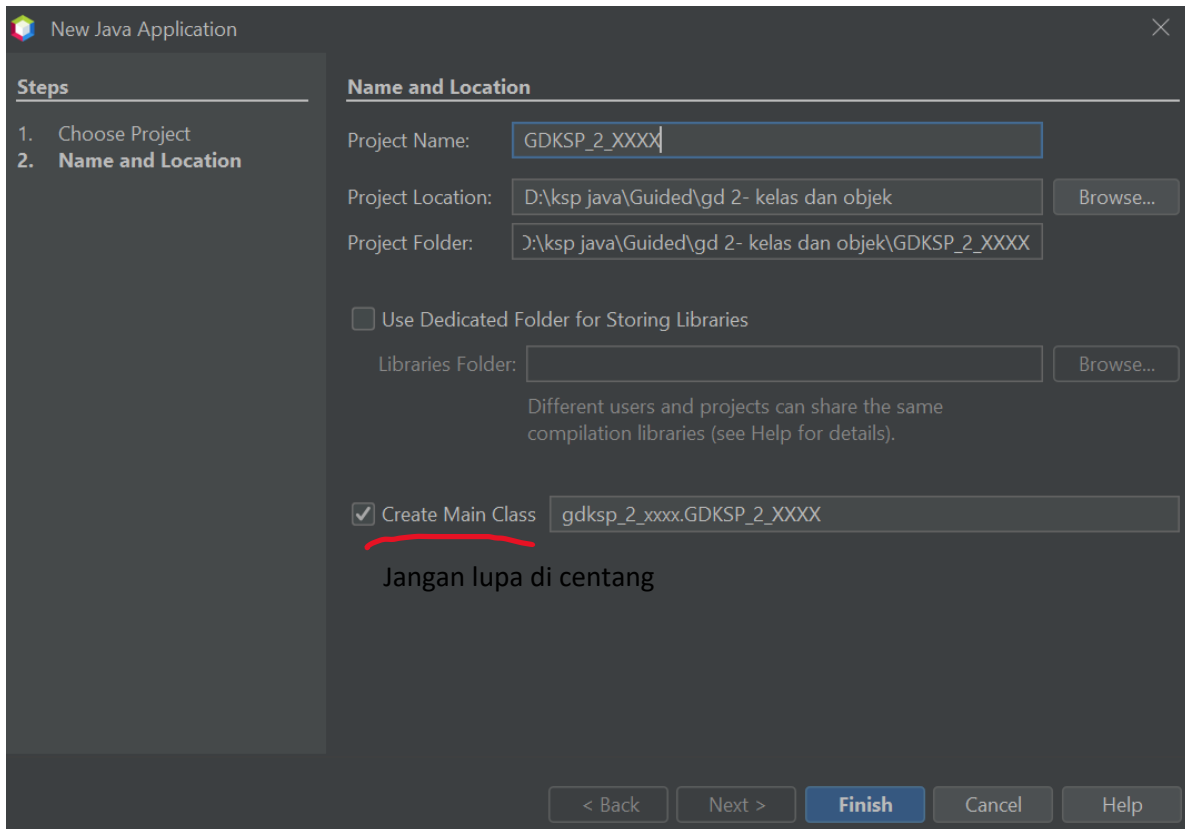


Atau klik tombol tersebut

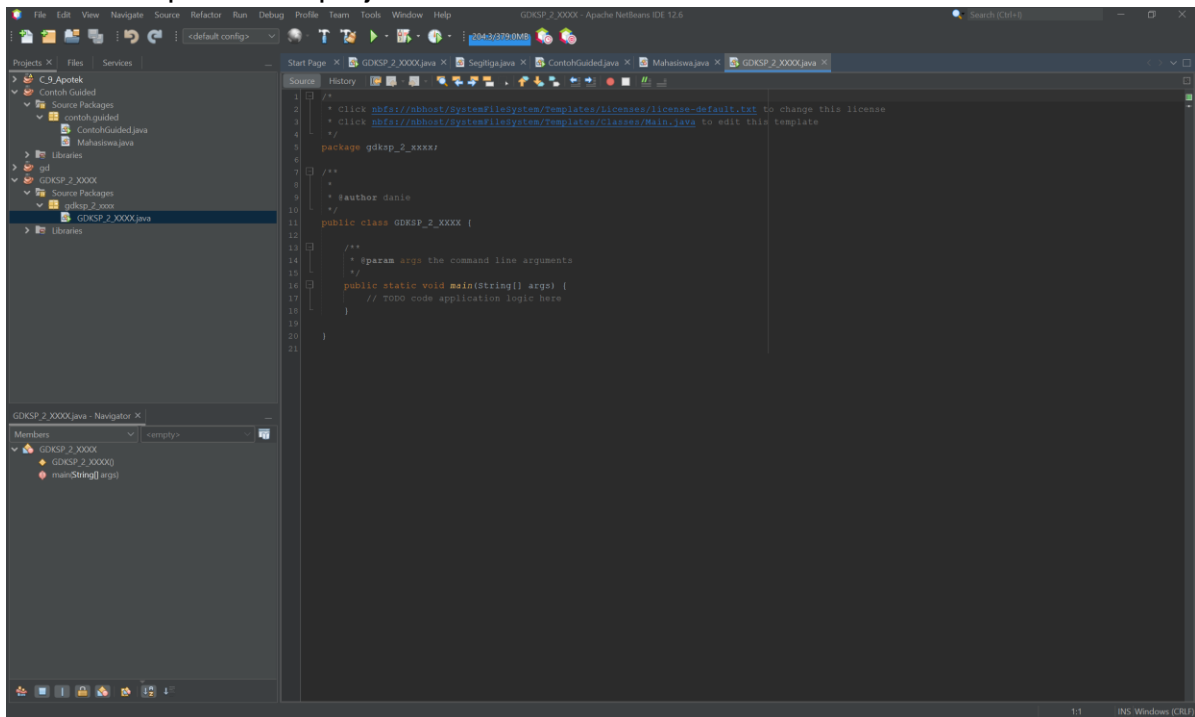
2. Pada bagian “Categories” pilih “Java with Ant” lalu pada bagian “Projects” pilih “Java Application”. Jika sudah klik “Next”



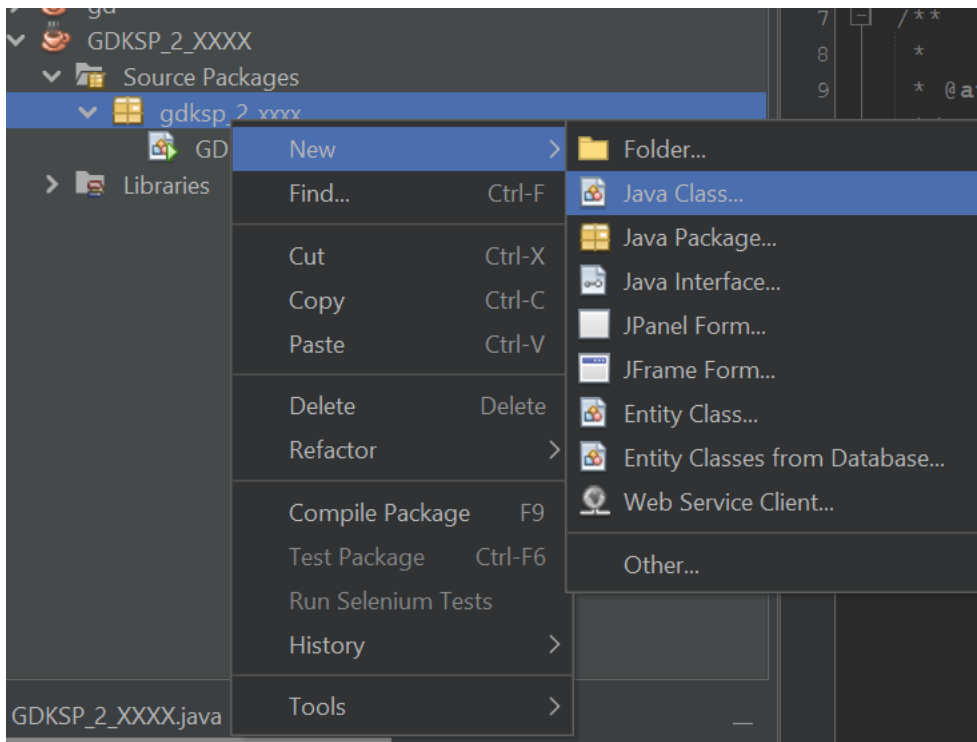
3. Beri nama project GDKSP_2_XXXX (XXXX adalah 4 digit terakhir npm). Pastikan checkbox “Create Main Class” sudah tercentang. Jika sudah klik “Finish”.



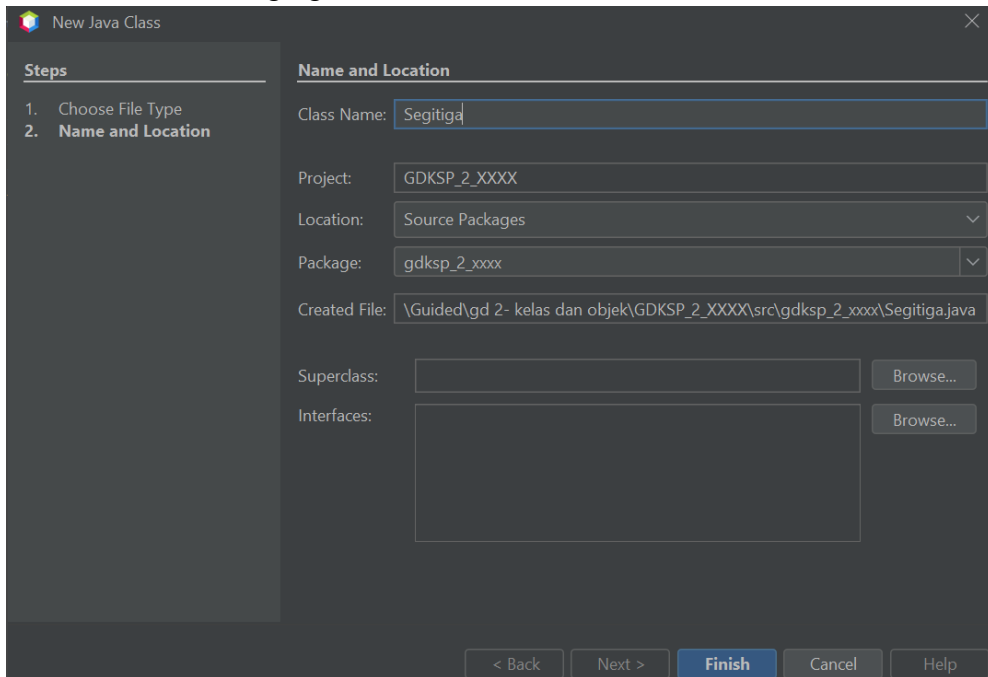
Berikut tampilan awal project:



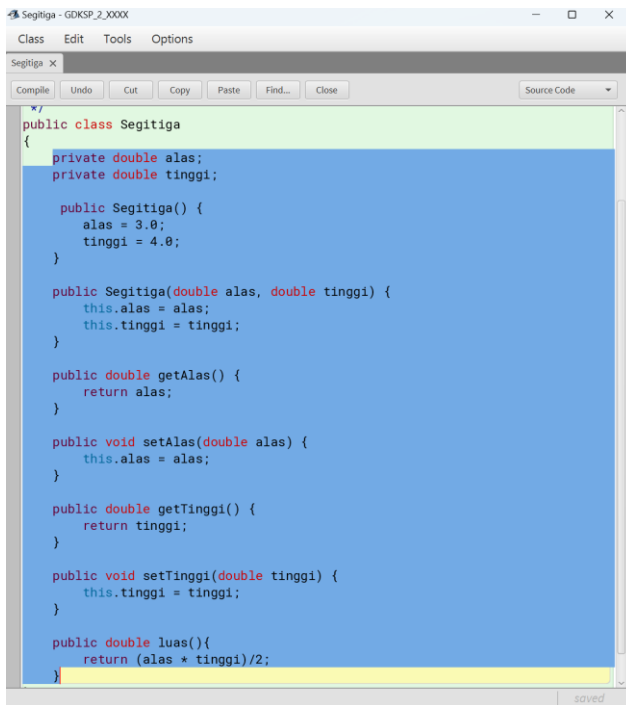
4. Buat kelas baru dengan cara right click pada package gdksp_2_xxxx> New > Java Class



5. Lalu beri nama Segitiga. Jika sudah klik “Finish”



6. Copy kode kelas Segitiga dari guided BlueJ tadi



```
public class Segitiga
{
    private double alas;
    private double tinggi;

    public Segitiga() {
        alas = 3.0;
        tinggi = 4.0;
    }

    public Segitiga(double alas, double tinggi) {
        this.alas = alas;
        this.tinggi = tinggi;
    }

    public double getAlas() {
        return alas;
    }

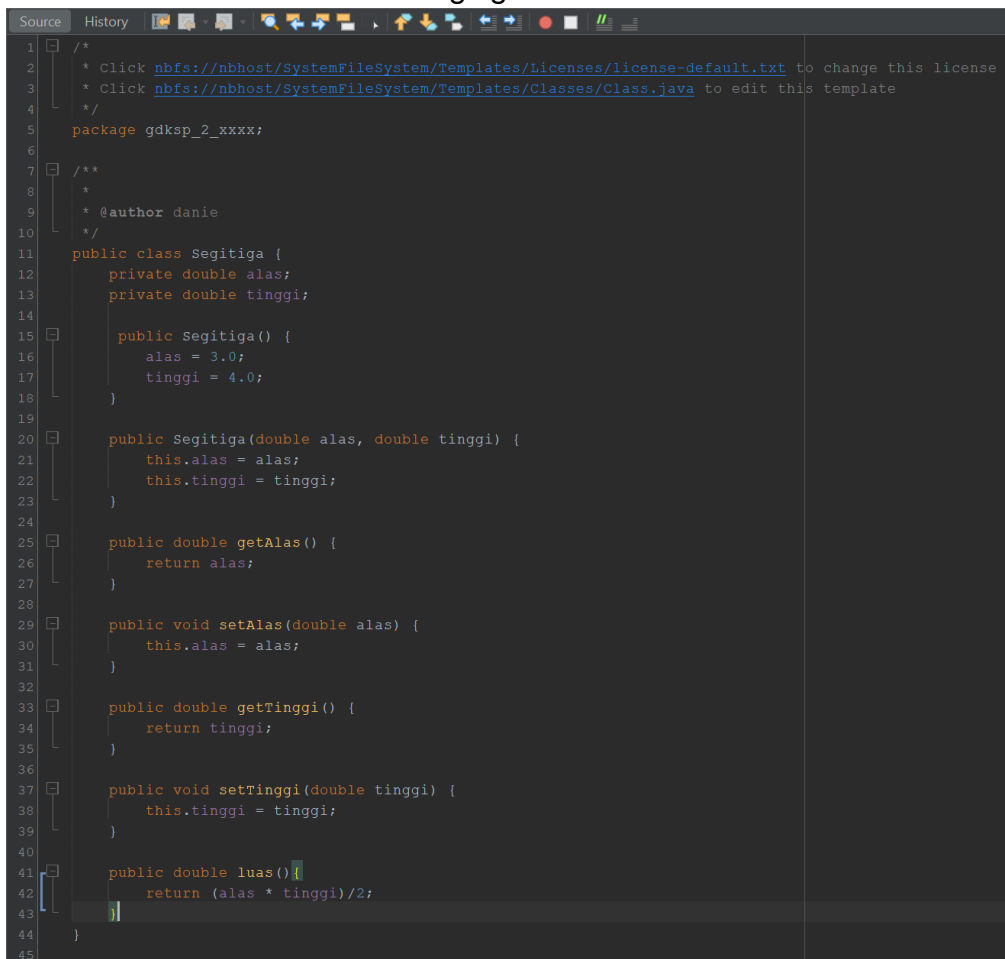
    public void setAlas(double alas) {
        this.alas = alas;
    }

    public double getTinggi() {
        return tinggi;
    }

    public void setTinggi(double tinggi) {
        this.tinggi = tinggi;
    }

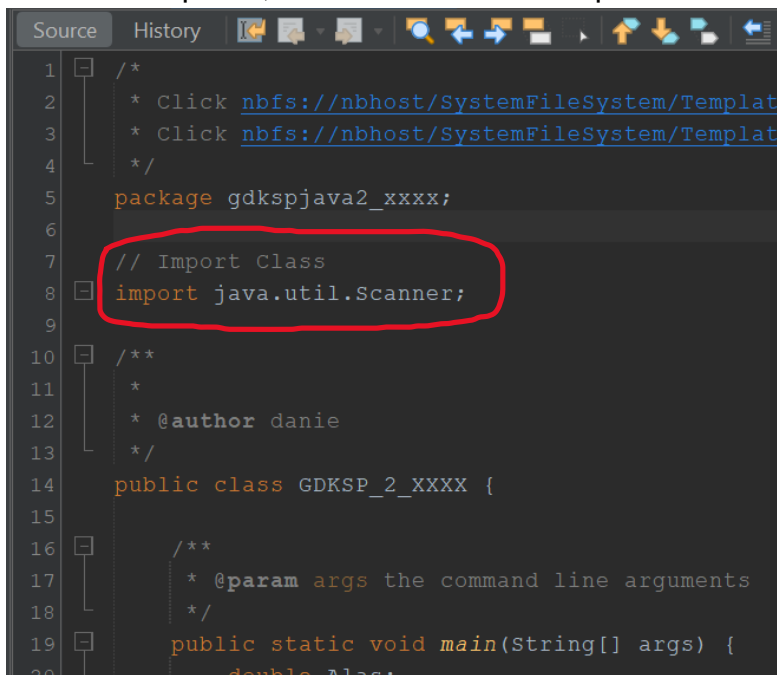
    public double luas(){
        return (alas * tinggi)/2;
    }
}
```

Dan masukan kedalam kelas segitiga di netbean



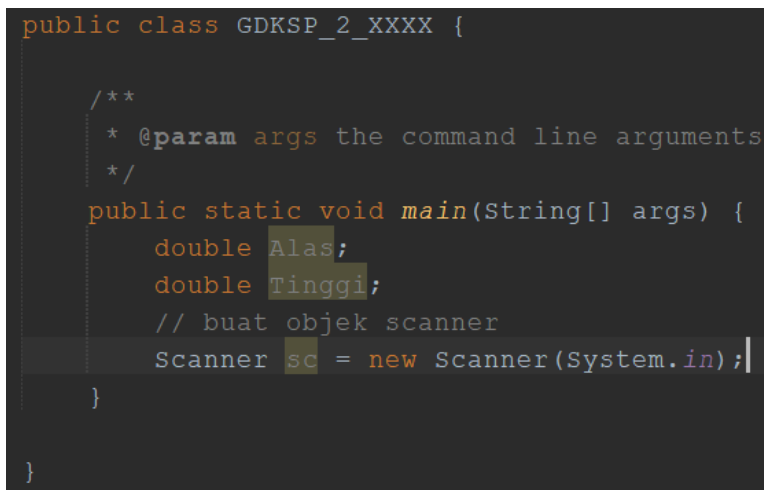
```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5   package gdksp_2_XXXX;
6
7   /**
8    *
9    * @author daniel
10   */
11  public class Segitiga {
12      private double alas;
13      private double tinggi;
14
15      public Segitiga() {
16          alas = 3.0;
17          tinggi = 4.0;
18      }
19
20      public Segitiga(double alas, double tinggi) {
21          this.alas = alas;
22          this.tinggi = tinggi;
23      }
24
25      public double getAlas() {
26          return alas;
27      }
28
29      public void setAlas(double alas) {
30          this.alas = alas;
31      }
32
33      public double getTinggi() {
34          return tinggi;
35      }
36
37      public void setTinggi(double tinggi) {
38          this.tinggi = tinggi;
39      }
40
41      public double luas(){
42          return (alas * tinggi)/2;
43      }
44  }
45
```


7. Masuk ke kelas GDKSP_2_XXXX, pertama import kelas Scanner yang nantinya akan digunakan untuk menerima inputan pada program Java kita. Banyak cara untuk menerima inputan, teman-teman bisa eksplor sendiri



```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates
3   * Click nbfs://nbhost/SystemFileSystem/Templates
4   */
5  package gdkspjava2_xxxx;
6
7  // Import Class
8  import java.util.Scanner;
9
10 /**
11  *
12  * @author daniel
13  */
14 public class GDKSP_2_XXXX {
15
16     /**
17     * @param args the command line arguments
18     */
19     public static void main(String[] args) {
20         double Alas;
```

8. Pada main buat variabel alas dan tinggi. Jika sudah buat objek baru dari kelas Scanner.



```
public class GDKSP_2_XXXX {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        double Alas;
        double Tinggi;
        // buat objek scanner
        Scanner sc = new Scanner(System.in);
    }
}
```

9. Buat 2 objek baru dari kelas Segitiga dengan nama segitiga1 dan segitiga2

```
public static void main(String[] args) {  
    double Alas;  
    double Tinggi;  
    // buat objek scanner  
    Scanner sc = new Scanner(System.in);  
  
    // buat objek segitiga1  
    Segitiga segitiga1 = new Segitiga();  
  
    // buat objek segitiga2  
    Segitiga segitiga2 = new Segitiga(5.0, 12.0);  
}
```

10. Tambahkan code berikut untuk menampilkan data objek segitiga1 dan segitiga2

```
//menampilkan objek segitiga1  
System.out.println("\n[segitiga1]");  
System.out.println("\nAlas : "+ segitiga1.getAlas());  
System.out.println("\nTinggi: "+segitiga1.getTinggi());  
System.out.println("\nLuas: "+segitiga1.luas());  
  
//menampilkan objek segitiga2  
System.out.println("\n[segitiga2]");  
System.out.println("\nAlas : "+ segitiga2.getAlas());  
System.out.println("\nTinggi: "+segitiga2.getTinggi());  
System.out.println("\nLuas: "+segitiga2.luas());
```

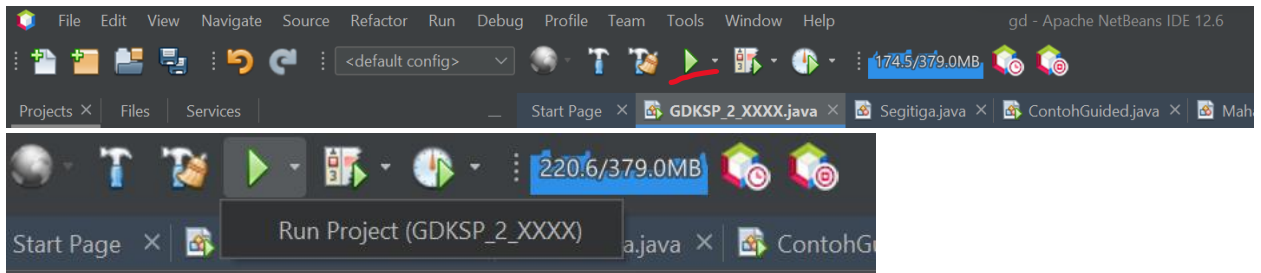
11. Lalu di bawah code tadi tambahkan code berikut sehingga program meminta user untuk menginputkan alas dan tinggi baru untuk segitiga1

```
// meminta user untuk melakukan input alas dan tinggi  
System.out.println("\n\n[Input alas dan tinggi baru segitiga1]");  
  
// input alas  
System.out.println("\nAlas : "); Alas = sc.nextDouble();  
System.out.println("\nTinggi: "); Tinggi = sc.nextDouble();  
  
segitiga1.setAlas(Alas);  
segitiga1.setTinggi(Tinggi);
```

12. Tambahkan code berikut untuk mengeset alas dan tinggi segitiga1 dari inputan yang sudah diterima sebelumnya lalu tampilkan kembali data objek segitiga1

```
//menampilkan objek segitiga1  
System.out.println("\n[segitiga1]");  
System.out.println("\nAlas : "+ segitiga1.getAlas());  
System.out.println("\nTinggi: "+segitiga1.getTinggi());  
System.out.println("\nLuas: "+segitiga1.luas());
```

13. Untuk run project bisa dengan cara klik button dengan icon segitiga hijau atau F6 pada keyboard



14. Beginilah hasil outputnya:

```
Output - GDKSP_2_XXXX (run)

run:

[segitiga1]

Alas : 3.0

Tinggi: 4.0

Luas: 6.0

[segitiga2]

Alas : 5.0

Tinggi: 12.0

Luas: 30.0

[Input alas dan tinggi baru segitiga1]

Alas :
```

15. Coba masukkan nilai untuk alas dan tinggi segitiga1

```
[Input alas dan tinggi baru segitiga1]

Alas :
7

Tinggi:
24

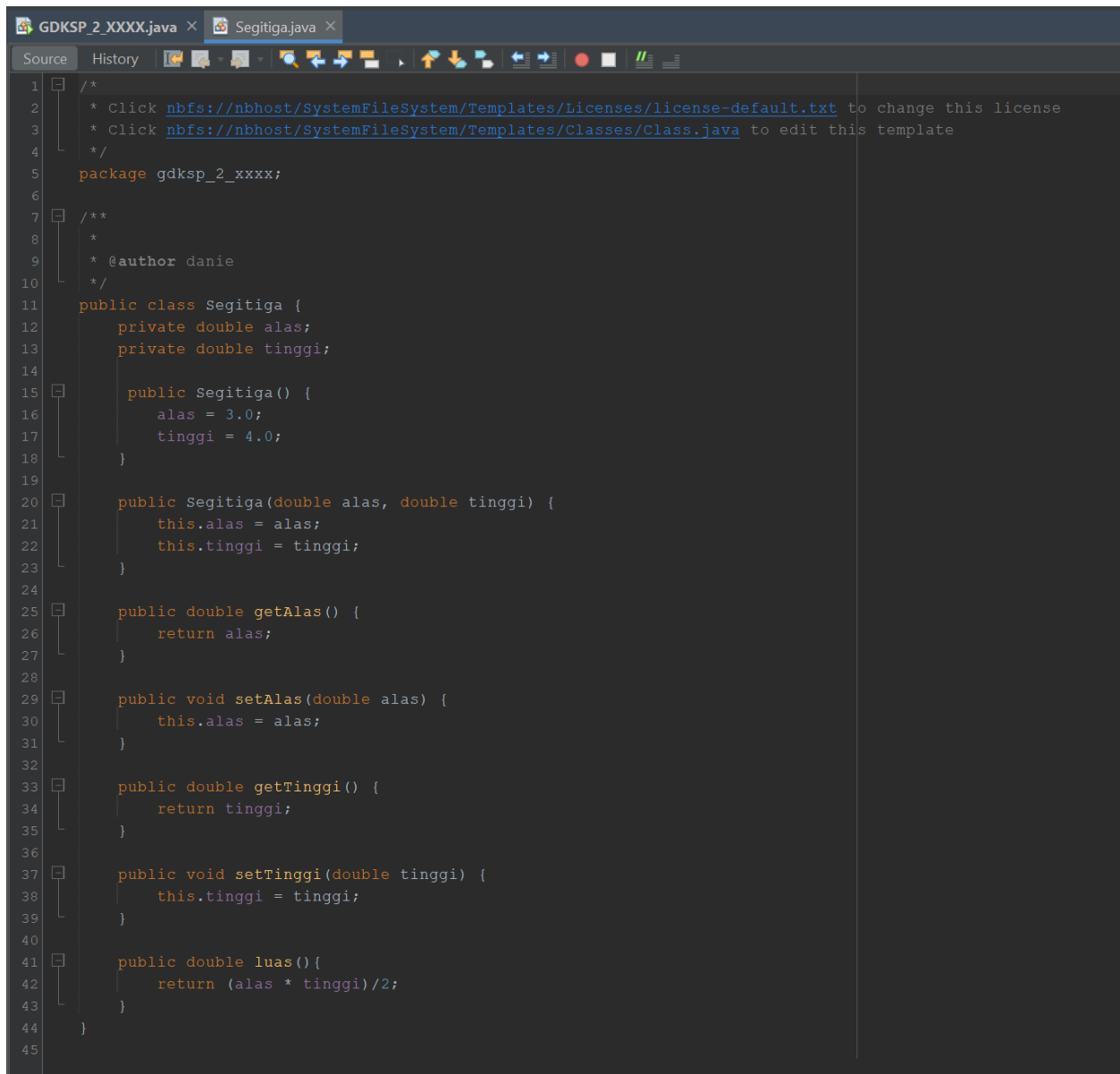
[segitiga1]

Alas : 7.0

Tinggi: 24.0

Luas: 84.0
```

Segitiga.java



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5   package gdksp_2_xxxx;
6
7   /**
8    *
9    * @author daniel
10   */
11   public class Segitiga {
12       private double alas;
13       private double tinggi;
14
15       public Segitiga() {
16           alas = 3.0;
17           tinggi = 4.0;
18       }
19
20       public Segitiga(double alas, double tinggi) {
21           this.alas = alas;
22           this.tinggi = tinggi;
23       }
24
25       public double getAlas() {
26           return alas;
27       }
28
29       public void setAlas(double alas) {
30           this.alas = alas;
31       }
32
33       public double getTinggi() {
34           return tinggi;
35       }
36
37       public void setTinggi(double tinggi) {
38           this.tinggi = tinggi;
39       }
40
41       public double luas(){
42           return (alas * tinggi)/2;
43       }
44   }
45
```

GDKSP_2_XXXX

```
GDKSP_2_XXXX.java x Segitiga.java x
Source History
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
4   */
5   package gdksp_2_XXXX;
6
7   // Import Class
8   import java.util.Scanner;
9   /**
10    *
11    * @author daniel
12    */
13   public class GDKSP_2_XXXX {
14       /**
15        * @param args the command line arguments
16        */
17       public static void main(String[] args) {
18           double Alas;
19           double Tinggi;
20           // buat objek scanner
21           Scanner sc = new Scanner(System.in);
22
23           // buat objek segitiga1
24           Segitiga segitiga1 = new Segitiga();
25
26           // buat objek segitiga2
27           Segitiga segitiga2 = new Segitiga(5.0, 12.0);
28
29           //menampilkan objek segitiga1
30           System.out.println("\n[segitiga1]");
31           System.out.println("\nAlas : "+ segitiga1.getAlas());
32           System.out.println("\nTinggi: "+segitiga1.getTinggi());
33           System.out.println("\nLuas: "+segitiga1.luas());
34
35           //menampilkan objek segitiga2
36           System.out.println("\n[segitiga2]");
37           System.out.println("\nAlas : "+ segitiga2.getAlas());
38           System.out.println("\nTinggi: "+segitiga2.getTinggi());
39           System.out.println("\nLuas: "+segitiga2.luas());
40
41           // meminta user untuk melakukan input alas dan tinggi
42           System.out.println("\n\n[Input alas dan tinggi baru segitiga1]");
43
44           // input alas
45           System.out.println("\nAlas : "); Alas = sc.nextDouble();
46           System.out.println("\nTinggi: "); Tinggi = sc.nextDouble();
47
48           segitiga1.setAlas(Alas);
49           segitiga1.setTinggi(Tinggi);
50
51           //menampilkan objek segitiga1
52           System.out.println("\n[segitiga1]");
53           System.out.println("\nAlas : "+ segitiga1.getAlas());
54           System.out.println("\nTinggi: "+segitiga1.getTinggi());
55           System.out.println("\nLuas: "+segitiga1.luas());
56
57       }
58   }
59
60 }
```