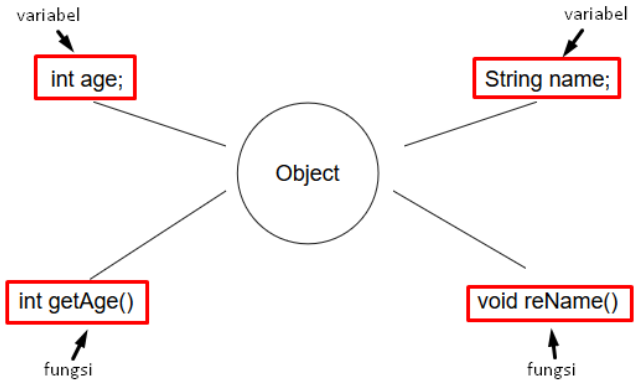
1. Java merupakan bahasa pemrograman yang multi platform dan multi device yang sudah ada sejak tahun 1990-an. Saat kita menuliskan program dengan Java, kita dapat menjalankannya hampir di semua komputer dan perangkat lain yang support Java.

2. OOP adalah teknik pemrograman yang berorientasi pada objek. Fungsi dan variabel dibungkus dalam sebuah objek/class yang dapat saling berorientasi, sehingga membentuk sebuah program.



3. Class dan Object
   - Class adalah definisi variabel dan fungsi yang menggambarkan sebuah object.
     - Berfungsi untuk mengumpulkan prosedur / fungsi dan variabel dalam satu tempat.
     - ex : Kita ingin membuat game sederhana, di dalamnya ada kucing dan makanan.

• **Prosedural**	• **OOP**

`var CatMood = "happy";`
`var CatEnergy = 80;`
`var Makanan = "Fish";`
 	
`catMove();`
`catPlay();`
`catJump();`
`catEat();`
	`class Kucing {`
	`  var mood;`
	`  var energi;`
	
	`  lari()`
	`  loncat()`
	`  makan()`
	`}`
	
	`class Makanan {`
	`  var nama;`
	`  var rasa;`
	
	`  hide()`
	`}`

   - Pembentukan class ini nanti yang akan dipakai untuk membuat object.

- Object adalah variabel yang instance / perwujudan dari class.
  - Dalam OOP :
    - Variabel : atribut / properti
    - Fungsi : method
  - ex :

```
class NamaClass {
    String atribut1;
    String atribut2;

    void namaMethod() {
    }
    void namaMethodLain() {
    }
}
```

    - Object-nya bisa dibuat :

```
NamaClass namaObj = new NamaClass();
```

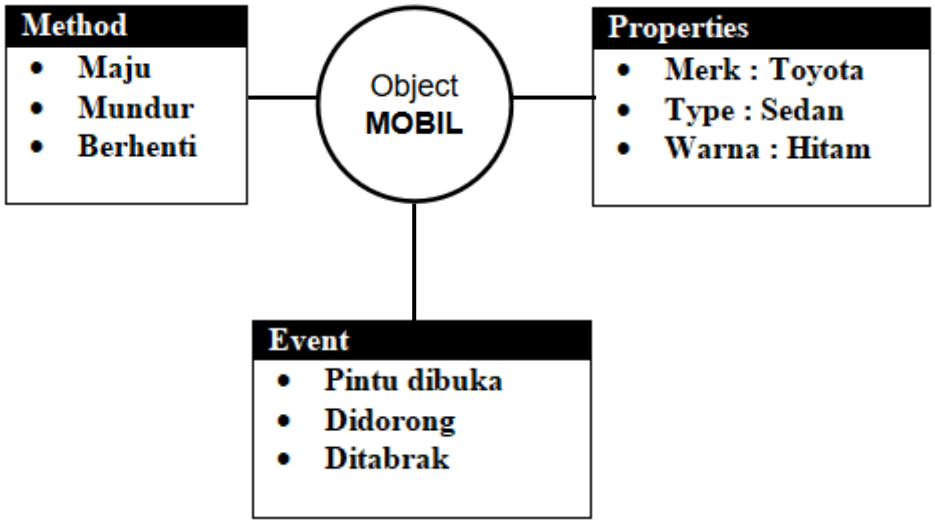
      new berfungsi untuk membuat object baru dari class tertentu.
  - Setelah membuat object, kita dapat mengakses atribut dan method dari object tersebut.
  - Ex :

```
namaObj.namaMethod();
namaObj.atribut1;
```

    - Tanda titik (.) berfungsi untuk mengakses atribut dan method yang telah dibuat.

4. Method, Properties, dan Event.
   - Properties adalah karakteristik yang dimiliki oleh suatu object.
   - Method adalah aksi yang dapat dilakukan oleh suatu object.
   - Event adalah kejadian yang dapat dialami oleh suatu object.

     ex :



5. JKT48

- Method :

Method
• **Menari** • **Menyanyi**

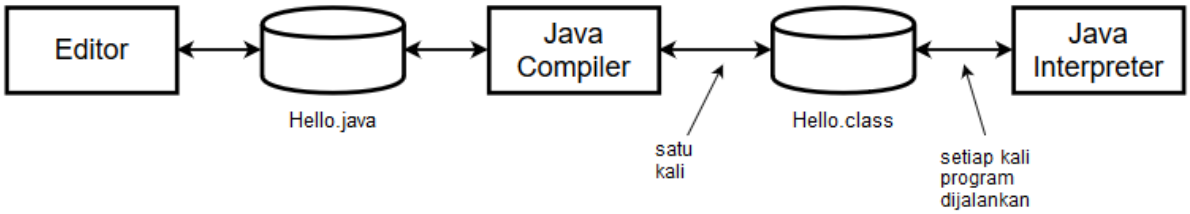
- Properties :

Properties
• **Rambut : Panjang** • **Tinggi : >150cm**

- Event :

Event
•

6. Bagaimana sebuah kode dieksekusi di Java?



- Langkah pertama yaitu menuliskan kode program pada text editor.
- Selanjutnya kode program yang dibuat kemudian tersimpan dalam sebuah berkas berekstensi .java .
- Setelah membuat dan menyiman kode program, kompilasi file yang berisi kode program tersebut dengan menggunakan Java compiler / kompilasi Java.
- Hasil dari kompilasi berupa berkas bytecode dengan ekstensi .class .
- Berkas yang mengandung bytecode tersebut kemudian akan dikonversikan oleh Java Interpreter menjadi bahasa mesin sesuai dengan jenis dan platform yang digunakan

Proses	Tools	Hasil
• Menulis kode program • Kompilasi program • Menjalankan program	• Text Editor • Java Compiler • Java Interpreter	• Berkas berekstensi .java • Berkas berekstensi .class • Program output

- Kompilasi java berada pada bagian Java Compiler di mana file source java asli yang mempunyai nama file .java di-compile menjadi .class yang nantinya akan diproses oleh Java Virtual Machine (JVM).

7. Java Virtual Machine (JVM) merupakan aplikasi yang harus di-install agar sebuah komputer bisa menjalankan program yang ditulis dalam bahasa Java.

8. Constructor dan Modifier
   - Constructor merupakan method khusus yang berfungsi menginisiasi sebuah object saat object pertama kali dibenttuk.
     - Nama constructor = Nama class

   - Modifier merupakan sifat yang dimiliki setiap atribut, method maupun kelas dalam Java. Modifier akses merupakan modifier yang selalu digunakan, modifier akses terdiri dari :
     - Private : method dan variabel hanya dapat diakses oleh **class yang sama**.
     - Pubic : method dan variabel dapat diakses pada **clas mana saja**.
     - Protected : method dan variabel dapat diakses di dalam package / di luar package, namun **hanya melalui inheritance** (pewarisan yang merupakan salah satu konsep pemrograman Java).
     - **Jika tidak menggunakan modifier apapun**, maka ini adalah **default**, di mana modifier ini hanya dapat diakses pada package yang sama.

9. Encapsulation, Polymorphism, Inheritance, Abstraction.
   - **Encapsulation** : pembungkusan class dan menjaga apa saja yang ada di dalam class tersebut, baik method ataupun atribut, agar tidak dapat diakses oleh class lain.
   - **Polymorphism** : method yang mempunyai banyak bentuk; kita dapat menimpa (override) suatu method yang berasal dari parent class (super class) di mana object tersebut diturunkan, sehingga memiliki kelakuan yang berbeda.
   - **Inheritance** : sebuah class yang dapat menurunkan property dan method yang dimilikinya pada class lain. Konsep ini digunakanuntuk memanfaatkan fitur ‘code reuse’ guna menghindari duplikasi kode program.
   - **Abstraction** : kelas yang masih berbentuk abstrak. Karena bentuknya masih abstrak, maka dia tidak bisa dibuat langsung menjadi object.

10. Instansiasi merupakan suatu **perwujudan**; digunakan untuk objek-objek yang menginisiasi dari sebuah class, sedangkan sebuah class dapat mempunyai banyak object.
    - ex :

```
public static void main (String[] args){
    Date tgl = new Date (); // tgl merupakan objek baru dari Date
    System.out.println ("Tanggal Sekarang adalah : " + tgl );
}
```

    - Ciri – ciri penulisan instansiasi = **new**
      - Membutuhkan operator **new** untuk mempersiapkan memory sesuai dengan isi kelas.

11. Inheritance merupakan sebuah class yang dapat **menurunkan property dan method** yang dimilikinya pada class lain. Class tersebut dapat mengakses data – data dari class utama yang bertindak sebagai **parent**.

12. 5 class standar dalam Java beserta penjelasan singkat dan penggunaannya.
    - Mencetak output berupa String.

```
public class ClassOne {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Output :

```
Hello World
```

    - Mencetak output berupa String dan melakukan **pemanggilan variable** namaLengkap.

```
public class ClassTwo {
    public static void main(String[] args) {
        String namaDepan = "Mentari ";
        String namaBelakang = "Rama";
        String namaLengkap = namaDepan + namaBelakang;
        System.out.println(namaLengkap);
    }
}
```

Ouput :

```
Mentari
Rama
```

    - Mencetak output berupa int dengan **pengoperasian operator tertentu**.

```
public class ClassThree {
    public static void main(String[] args) {
        int jml1 = 100 + 50;
        int jml2 = jml1 + 250;
        int jml3 = jml2 + jml2;

        System.out.println(jml1);
        System.out.println(jml2);
        System.out.println(jml3);
    }
}
```

Output :

```
150
400
800
```

- Mencetak output berupa String yang menyimpan **barisan karakter**.

```
public class ClassFour {
    public static void main(String[] args) {
        String txt = "Hello World";

        System.out.println(txt.toUpperCase());
        System.out.println(txt.toLowerCase());
    }
}
```

Output :

```
HELLO WORLD
hello world
```

- Mencetak output berupa String dengan inputan **static** dan menggunakan **if** (kondisi) pada program.

```
public class ClassFive {
    public static void main(String[] args) {
        if (20 > 18) {
            System.out.println("20 lebih besar dari 18"); //mutlak
        }
    }
}
```

Output :

```
20 lebih besar dari 18
```

Penggunaan if-else :

Mencetak output berupa String dengan inputan **static** dan menggunakan **if-else** pada program.

```
public class ClassFive {
    public static void main(String[] args) {
        int umur = 20;
        if (umur < 18) {
            System.out.println("Tidak boleh masuk.");
        } else {
            System.out.println("Silahkan masuk.");
        }
    }
}
```

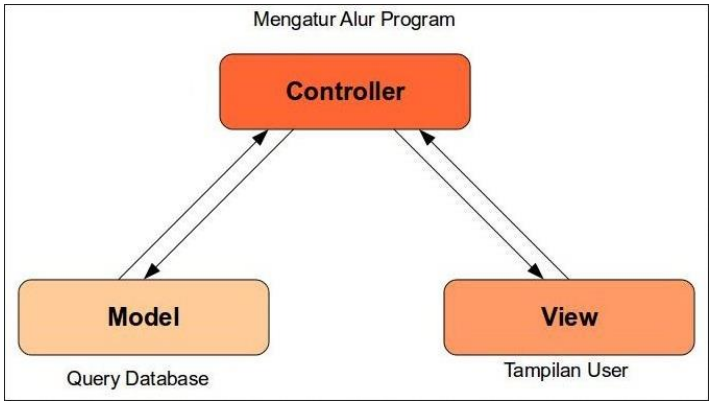
Output :

```
Silahkan masuk.
```

13. Garbage Collector adalah salah satu mekanisme dan fitur dari Java Virtual Machine (JVM) untuk meningkatkan Memory Management yang akan menghapus object secara otomatis pada memory jika tidak dibutuhkan lagi.

14. Model - View – Controller
    - Model – View – Controller (MVC) merupakan sebuah metode untuk membuat sebuah aplikasi dengan **memisahkan** data (Model) dari tampilan (View) dan cara bagaimana

memprosesnya (Controller). Dalam implementasinya kebanyakan kerangka kerja (framework) dalam aplikasi web adalah berbasis arsitektur MVC. MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, antarmuka pengguna, dan bagian yang menjadi kontrol dalam sebuah aplikasi web.



15. Sistem yang dapat dibangun dengan Java di antaranya ialah :
    - Banking
    - Perpustakaan
    - Informasi Akademik
    - Laundry
    -

16. Framework merupakan **kerangka kerja** dalam suatu software untuk memudahkan para programmer membuat sistem yang berisi fungsi, plug-in, serta konsepnya sehingga membentk suatu sistem yang utuh. Dengan menggunakan framework, sebuah sistem akan **lebih tersusun dan terstruktur dengan rapih**. Namun, penggunaan framework bukan berarti kita terbebas dari pengkodean. Kita sebagai programmer menggunakan variabel & fungsi – fungsi tertentu yang terdapat di framework tersebut. Jadi, pekerjaan kita **lebih efektif** karena tidak harus membuat fungsi – fungsi lagi.

17. Java Spring merupakan framework(springframework) open source berbasis Java yang menyediakan infrastruktur yang komprehensif dalam mengembangkan aplikasi Java dengan **mudah dan cepat**. Spring akan membantu programmer dalam pengembangan aplikasi dengan build yang sederhana, portable, cepat, dan sistem berbasis JVM yang lebih fleksibel. Hal Ini **memudahkan bagi para developer pemula** untuk mempelajarinya. Jika mempelajari Spring, para developer berevolusi menjadi developer yang lebih baik. Hal ini karena framework Spring mendorong untuk membuat kode program yang modular dan independen. Hasilnya, kode program yang dibuat akan **lebih rapi, mudah dites, dan terstruktur dengan baik.**

18. Java Spring dan Spring Boot
    - Java Spring merupakan framework(springframework) open source berbasis Java yang menyediakan infrastruktur yang komprehensif dalam mengembangkan aplikasi Java dengan **mudah dan cepat** sedangkan Spring Boot merupakan salah satu jenis adalah salah satu varian dari **springframework**, yang mempermudah proses web development di Java. Kita tidak perlu capek-capek nulis file konfigurasi xml atau pusing dengan cara deploy di application server.

19. Java dibangun dengan bahasa pemrograman lain; Java mengadopsi sintaks – sintaks pada **bahasa C dan C++**, dengan sintaks yang lebih sederhana. Java juga merupakan bahasa pemrograman yang bersifat umum / non – spesifikasi dan secara khusus di-design untuk memanfaatkan implementasi seminimal mungkin.

20. Yang membuat Java menjadi powerful dibanding Procedural Programming karna konsep pemrograman OOP **lebih fokus pada masalah yang ditangani dengan menggunakan komputer** sedangkan Procedural Programming lebih fokus pada bagaimana komputer menangani masalah, dan OOP biasanya digunakan untuk pembuatan sistem yang lebih kompleks karna cara berfikirnya bisa seperti manusia.