MAKALAH

PEWARISAN DALAM BAHASA JAVA

****

Oleh :

Diki Candra

NIM 2022903430010

D4 TEKNOLOGI REKAYASA KOMPUTER JARINGAN

POLITEKNIK NEGERI LHOKSEMAWE

2022/2023

**KATA PENGANTAR**

Alhamdulillah kami panjatkan puji syukur kami ke hadirat Allah SWT. Sehingga penulis telah menyelesaikan makalah polimorfismeini dengan tepat waktu.

Salah satu tujuan penulis dalam menulis Makalah polimorfisme ini adalah sebagai dokumentasi dan juga bentuk evaluasi kegiatan polimorfisme. Makalah yang penulis buat ini berdasarkan data-data yang valid yang telah dikumpulkan dalam berbagai metode.

Penulis menyadari atas ketidaksempurnaan penyusunan Makalah polimorfisme ini. namun penulis tetap berharap laporan ini akan memberikan manfaat bagi para pembaca. Demi kemajuan penulis, penulis juga mengharapkan adanya masukan berupa kritik atau saran yang berguna. Terima kasih

**DAFTAR ISI**

**KATA PENGANTAR ii**

**DAFTAR ISI iii**

**BAB I PENDAHULUAN**

1.1 Latar Belakang 1

1.2 Rumusan Masalah 2

1.3 Tujuan Penelitian 2

**BAB II PEMBAHASAN**

2.1 Pengertian Polimorfisme 3

2.2 Jenis Polimorfisme Pada Java 3

2.3 Metode Overloading Dan Overriding 4

2.4 Kelebihan Operator 9

2.5 Polimorfisme Dinamis 10

2.6 Polimorfisme Runtime Dengan Anggota Data 13

2.7 Konsep Polimorfisme Pada Java 14

2.8 Implementasi Polimorfisme 17

2.9 Perbedaan Jenis Polimorfisme 22

2.10 Keuntungan Menggunakan Polimorfisme 27

**BAB III PENUTUP**

3.1 kesimpulan 28

**DAFTAR PUSTAKA**

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Dalam Pemrograman Berbasis Objeck (PBO), kita harus memperhatikan dan memahami konsep-konsep dasar pada program itu sendiri, khususnya pada program java yang akan kita pakai. Pada makalah ini ditekankan untuk lebih memahami konsep dasar suatu sifat pada Java, yaitu polymorphisme, bagaimana fungsinya dan cara penulisannya. Oleh karena itu kami membahasnya dalam makalah ini.

Polimorfisme terbagi menjadi dua suku kata yaitu, *Poly* yang berarti banyak dan *Morfisme* yang berarti bentuk. Dalam ilmu sains, *Polimorfisme (polymorphism)* adalah sebuah prinsip dalam biologi di mana organisme atau spesies memiliki banyak bentuk serta tahapan (stages). Prinsip tersebut diterapkan juga pada bahasa Java.

Polimorfisme dalam OOP merupakan sebuah konsep OOP di mana class memiliki banyak “bentuk” method yang berbeda, meskipun namanya sama. Maksud dari “bentuk” adalah isinya yang berbeda, namun tipe data dan parameternya berbeda.

Polimorfisme juga dapat diartikan sebagai teknik programming yang mengarahkan kamu untuk memprogram secara general daripada secara spesifik. Contohnya kita memiliki tiga class yang berbeda yaitu: “Kelinci”, “Kucing”, dan “Sapi”. Di mana ketiga class tersebut merupakan turunan dari class “Hewan”.

Polimorfisme pada Java memiliki 2 macam yaitu diantaranya:

1. Static Polymorphism (Polimorfisme statis).
2. Dynamic Polymorphism (Polimorfisme dinamis).

Perbedaan keduanya terletak pada cara membuat polimorfisme. **Polimorfisme statis menggunakan method overloading, sedangkan polimorfisme dinamis menggunakan method overriding.**

Untuk mendapatkan operasi polimorfisme dari suatu method, method baru ini harus merupakan member dari kelas induk. Dengan kata lain, Anda harus mendefinisikan method yang akan dipanggil ini dalam kelas. Beberapa syarat yang harus dipenuhi agar operasi polimorfisme dapat berlangsung adalah:

* Method dari kelas turunan yang akan dieksekusi, dipanggil oleh objek kelas induk
* Method ini harus merupakan member dari kelas induk
* Nama method di kelas induk harus sama dengan yang ada di kelas turunan
  1. **Rumusan Masalah**

1. Apa Pengertian Polimorfisme?
2. Bagaimana Konsep Polimorfisme?
3. Bagaimana konsep method overloading dan overriding?
4. Bagaimana Implementasi Polimorfisme?
5. Apa Saja Keuntungan Menggunakan Polimorfisme?
6. Bagaimana Jenis Polimorfisme Pada Java?
   1. **Tujuan Penelitian**
7. Untuk dapat mengetahui apa itu polimorfisme
8. Untuk dapat mengetahui bagaimana konsep polimorfisme
9. Untuk dapat mengetahui Bagaimana konsep method overloading dan overriding
10. Untuk dapat mengetahui Bagaimana Implementasi Polimorfisme
11. Untuk dapat mengetahui Apa Saja Keuntungan Menggunakan Polimorfisme
12. Untuk dapat mengetahui Bagaimana Jenis Polimorfisme Pada Java

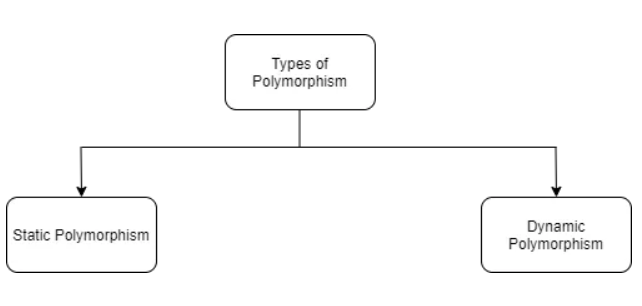
**BAB II**

**PEMBAHASAN**

* 1. **Pengertian Polimorfisme**

Dalam pemrograman berorientasi object kita mengenal suatu istilah yaitu “ Polimorfisme ”, yaitu suatu teknik dalam programming yang lebih mengarahkan kita untuk memprogram secara general dari pada secara spesifik. Sebagai contoh seumpama kita memiliki 3 class berbeda yaitu: “Motor”, “Mobil” dan “Pesawat”. Dimana ketiga class tersebut adalah turunan dari class “Kendaraan”. Dalam kelas kedaraan terdapat suatu method yaitu “Move()”, akan tetapi apakah method “Move()” akan memberikan hasil yang sama pada tiga class tersebut. Dimana jika method “Move()” dipanggil pada class “Motor” akan menghasilkan perpindahan 30Km, jika pada class “Mobil” akan menghasilkan perpindahan 70Km dan jika dipanggil pada class “Pesawat” akan menghasilkan perpindahan 300Km. Hal inilah yang kita sebut sebagai Polimorfisme , yaitu penggunaan method yang sama, yang akan menghasilkan suatu hasil yang berbeda tergantung class yang menggunakan method tersebut.

* 1. **Jenis Polimorfisme Pada Java**



* **Polimorfisme Statis**

Saat kami membebani metode statis, kami menyebutnya sebagai polimorfisme statis. Karena ini menyelesaikan polimorfisme selama waktu kompilasi, kita dapat menyebutnya juga sebagai **kompilasi-waktu** polimorfisme. Kita dapat mengimplementasikan polimorfisme jenis ini di java menggunakan salah satunya **metode overloading** or [operator](https://www.tutorialcup.com/id/Jawa/operator-di-java.htm)**kelebihan beban.** Dalam polimorfisme statis, selama waktu kompilasi, metode ini mengidentifikasi metode mana yang akan dipanggil berdasarkan parameter yang kita lewati.

* 1. **Metode Overloading Dan Overriding**

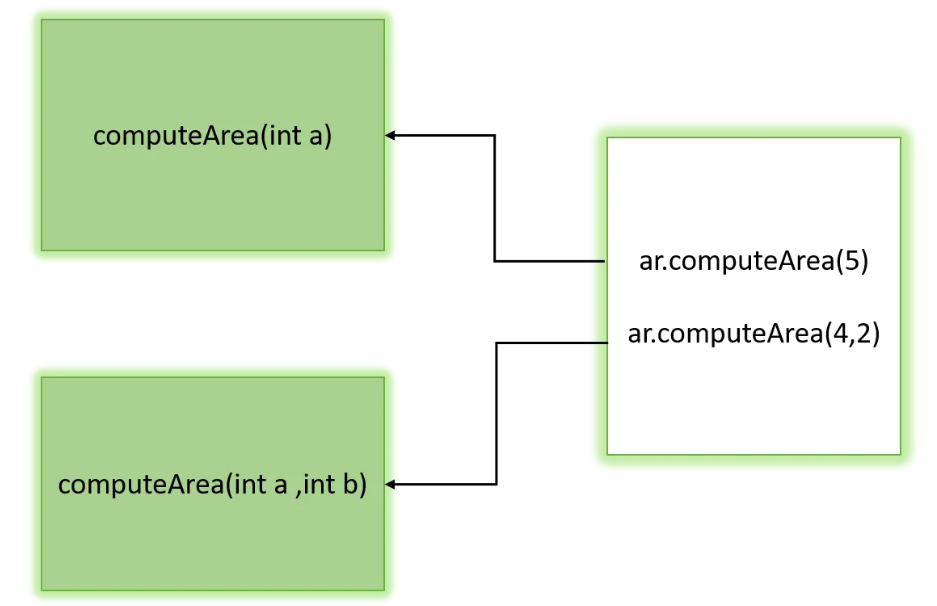
**Aturan Method Overloading**

Jika ada banyak metode dengan nama yang sama tetapi penerapannya berbeda, kami menyebutnya metode overloading. Kita dapat mengimplementasikan metode overloading dengan cara yang berbeda:

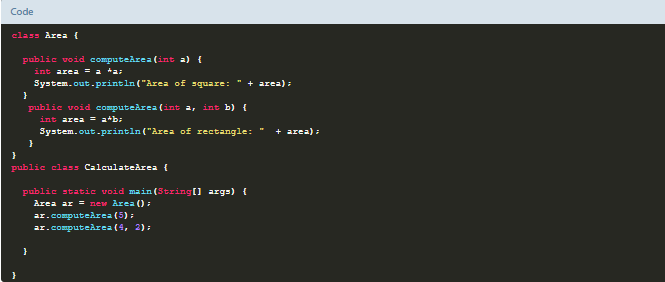
Aturan Method Overloading

* Nama method harus sama dengan method lainnya.
* Parameter haruslah berbeda.
* Return boleh sama, juga boleh berbeda.

Pertama, mari kita lihat contoh metode overloading dengan file **jumlah parameter yang berbeda.**Dalam contoh ini, kami telah membuat kelas **Daerah**yang memiliki 2 metode **computeArea (int) dan computeArea (int, int).**Perbedaan kedua metode tersebut adalah yang satu memiliki 1 parameter dan yang lainnya memiliki 2 parameter meskipun nama metodenya sama. Jadi ketika kita memanggil metode menggunakan 1 parameter, itu memanggil computeArea (int) dan ketika kita memanggil metode dengan 2 parameter, itu memanggil computeArea (int, int).



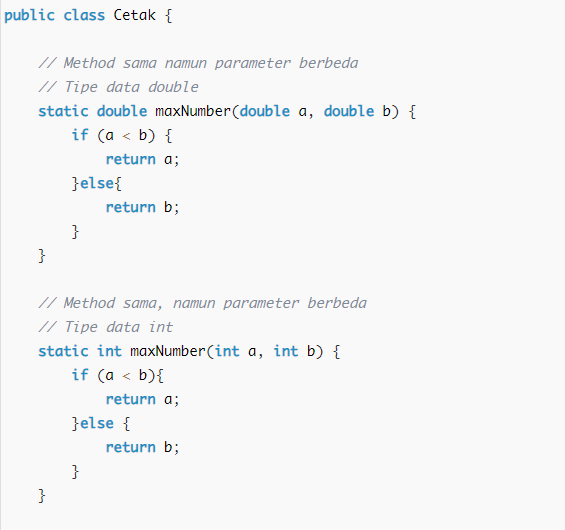
**Source Code :**

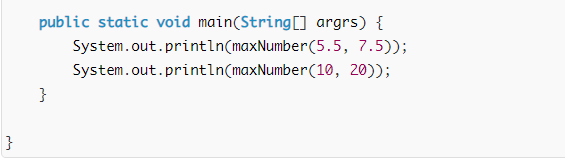


**Output :**



Adapun contoh lain lagi agar lebih paham.





Maka hasil outputnya adalah

5.5  
10

Pada class Cetak.java memiliki 2 method yang sama yaitu maxNumber(). Tapi parameter dan tipenya berbeda, yaitu:

static double maxNumber(double a, double b)

static int maxNumber(int a, int b)

Yang pertama memiliki parameter dan tipe data double, sedangkan satunya lagi memiliki parameter dan tipe data int. Hal ini jelas berbeda.

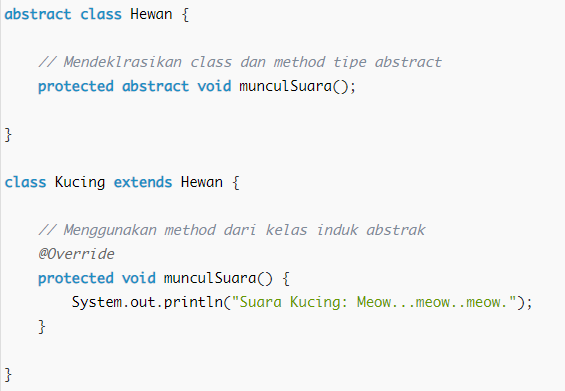
Interface adalah class kosong yang berisi nama-nama method dan nantinya harus diimplementasikan pada class lain. Dalam pengimplementasiannya, tiap-tiap class akan mengimplementasikan secara berbeda dengan nama method yang sama.

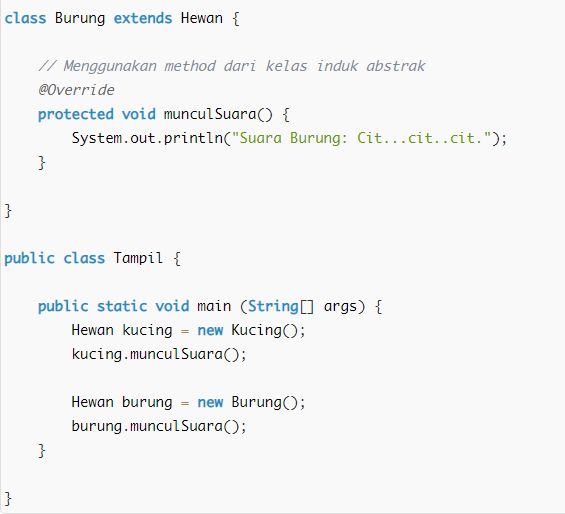
Abstrak adalah class yang masih dalam bentuk bayangan. Ia tidak bisa dibuat langsung menjadi objek karena bentuknya masih bayangan atau abstrak. Tentunya abstrak ini induk dan jika ini konkrit, kamu mesti mengimplementasikan method-method tersebut. Ini bisa kamu lakukan dengan melakukan teknik pewarisan (inheritance).

**Aturan Method Overriding**

* Mode akses overriding method harus sama atau lebih luas daripada override method.
* Subclass hanya dapat dan boleh meng-override method superclass satu kali saja. Tidak boleh ada lebih dari satu method yang sama pada kelas.
* Soal aturan hak akses, setiap subclass tidak boleh mempunyai hak akses method overriding yang ketat dibandingkan dengan hak akses method pada superclass ataupun parent class.

Berikut contoh program method overriding





**Maka Outputnya:**

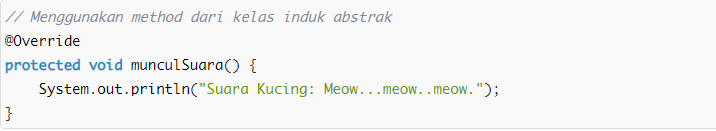
Suara Kucing: Meow…meow..meow.

Suara Burung: Cit…cit..cit.

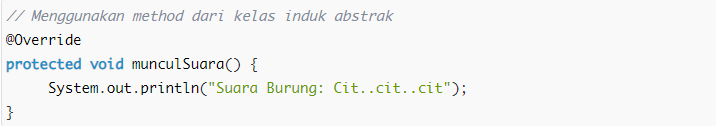
**Telusuri Kode**

Class Hewan.java memiliki method utama yaitu munculSuara(). Kita menambahkan beberapa subclass anak kelas dari Hewan.java yakni Kucing.java dan Burung.java. Mereka memiliki method yang sama meskipun menampilkan statements argumen yang berbeda.

**Kucing.java**



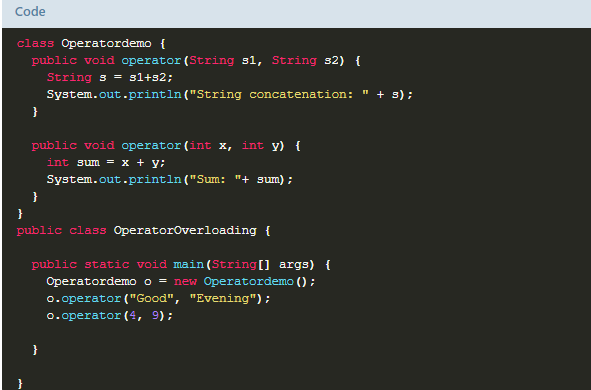
**Burung.java**



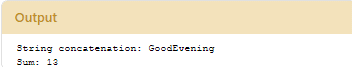
* 1. **Kelebihan Operator**

Mirip dengan metode overloading, kita juga bisa mengganti operator yang juga merupakan contoh dari [statis](https://www.tutorialcup.com/id/Jawa/kata-kunci-statis-java.htm" \t "_blank) polimorfisme di jawa. Tapi kita hanya bisa membebani "**+**Operator di Jawa dan tidak mendukung overloading operator lain. Saat kami menggunakan Tali sebagai operan antara + itu menghasilkan Tali rangkaian. Saat kami menggunakan **nomor** sebagai operan antara **+** itu menghasilkan **tambahan**dari dua angka. Di bawah ini adalah contoh dari **+** kelebihan beban operator.

**Source Code :**



**Output :**



* 1. **Polimorfisme Dinamis**

Ketika polimorfisme terpecahkan selama runtime, kami menyebutnya **dinamis** or **runtime**polimorfisme. Dalam jenis polimorfisme di java ini, ia menggunakan variabel referensi dari superclass untuk memanggil metode yang diganti. Artinya, berdasarkan objek yang dirujuk oleh variabel referensi, ia memanggil metode yang diganti dari kelas yang sesuai. Kita bisa menggunakan **metode menimpa**konsep untuk menerapkan polimorfisme dinamis.

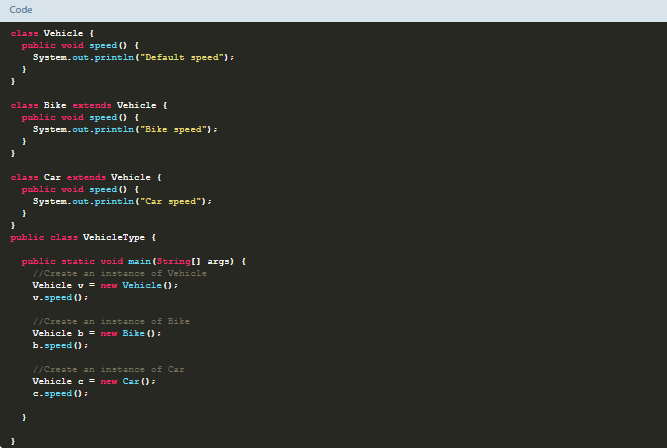
* **Mengganti Metode**

Ketika subclass memiliki metode yang sama dengan kelas dasar, kami menyebutnya metode penimpaan yang berarti bahwa subkelas tersebut telah menimpa metode kelas dasar. Berdasarkan jenis objek yang kita buat, ia memanggil metode kelas yang sesuai. Artinya, jika kita membuat sebuah objek dari superclass dan mereferensikannya menggunakan subclass, maka metode dari subclass tersebut akan dipanggil. Karena ini menghitung ini selama runtime, kami menyebutnya sebagai polimorfisme runtime di java.

Anda akan dapat memahami konsep ini dengan jelas dengan contoh di bawah ini. Kami telah membuat kelas induk **Kendaraan**dan 2 subclass **Sepeda dan Mobil.**Kelas induk memiliki metode **kecepatan**dan kedua subkelas telah mengganti metode kelas dasar **kecepatan.**

Jika kita membuat objek instance kelas Vehicle dan memanggil metode kecepatan (v.speed) itu memanggil metode kelas induk. Ketika kita memanggil metode kecepatan menggunakan objek yang dibuat dari kelas Sepeda contoh (b.speed), itu memanggil metode kelas Sepeda. Demikian pula, ketika kita memanggil metode menggunakan objek yang dibuat dari instance kelas Car (c.speed), ia memanggil metode kelas Car.

**Source Code :**



**Output :**



Di bawah ini adalah contoh lain dari polimorfisme runtime di mana kita dapat menggunakan nama objek yang sama untuk membuat beberapa instance kelas. Kami pertama-tama mendeklarasikan sebuah objek **f**kelas **Buah-buahan** Kemudian gunakan **baru**kata kunci kita instantiate variabel objek untuk kelas Buah. Saat kita menggunakan objek ini untuk memanggil metode rasa, itu memanggil metode rasa dari kelas Buah. Selanjutnya, ketika kita memberi contoh variabel objek untuk kelas Apple dan memanggil metode rasa menggunakan objek ini, itu memanggil metode kelas Apple. Begitu pula yang terjadi pada kelas Nanas.

**Source Code :**

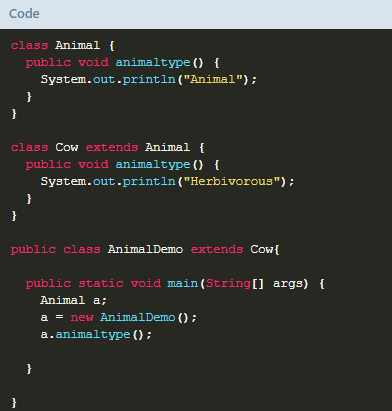


**Output :**



Mari pertimbangkan contoh lain di bawah ini di mana kita memanggil metode subkelas yang tidak memiliki metode yang diganti. Dalam kasus seperti itu, itu memanggil metode kelas orang tua. Demo Hewan kelas tidak memiliki metode jenis binatang. Oleh karena itu disebut metode Sapi kelas sejak kelas AnimalDemo memperluas kelas Sapi.

**Source Code :**



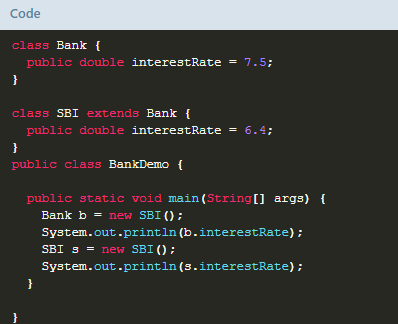
**Output:**



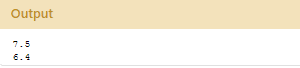
* 1. **Polimorfisme Runtime Dengan Anggota Data**

Dalam polimorfisme runtime, hanya metode yang diganti dan bukan anggota data atau variabel. Dari contoh di bawah ini, kita dapat dengan jelas memahami perbedaan ini. Saat kita membuat sebuah objek **b**Misalnya kelas SBI dari **Bank** kelas, itu memanggil variabel superclass karena anggota data tidak diganti. Untuk mengakses variabel subclass, kita perlu membuat objek dari subclass ie **SBI s = SBI baru ().**

**Source Code :**



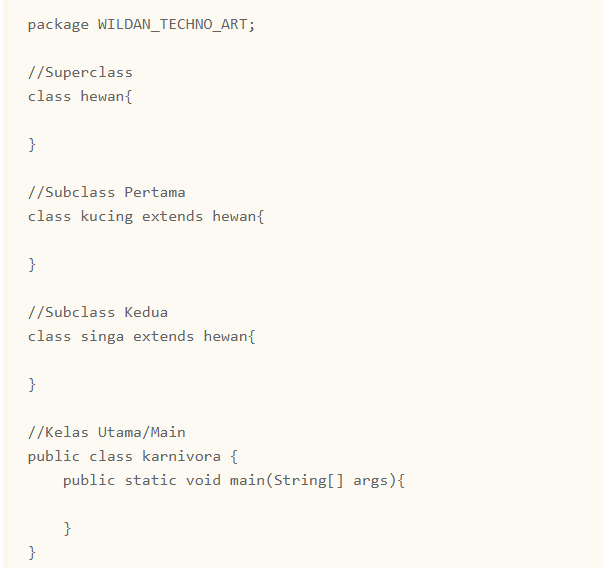
**Output :**



* 1. **Konsep Polimorfisme Pada Java**

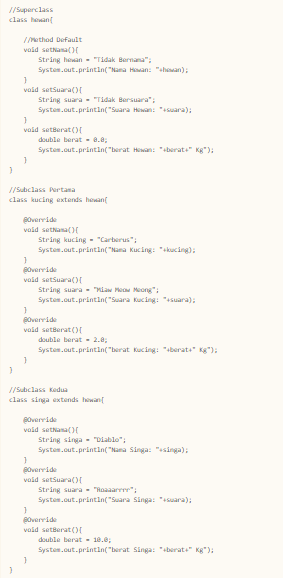
Pada program yang kita buat, kita akan menggunakan 4 class sekaligus, dimana terdapat 1 superclass dan beberapa subclass, masing-masing class tersebut akan kita berinama hewan (superclass), kucing (subclass), singa (subclass) dan karnivora (class utama/main). Diamana semua data-data dari masing-masing subclass akan dijalnkan pada class karnivora, dan jangan lupa pada class kucing dan singa harus mewarisi/mengextends sifat-sifat dari class hewan, untuk persiapan, buatlah seperti berikut ini :

**Source Code :**



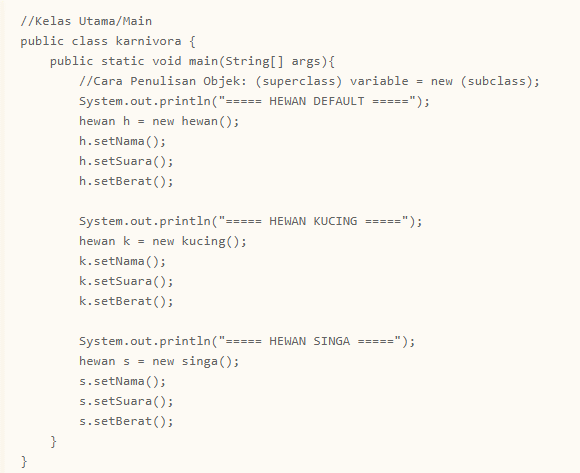
Selanjutnya pada class hewan akan kita tambahkan beberapa atibut seperti variable dan method, dan pada masing-masing subclass akan menggunakan nama method yang sama dengan kelas induk mereka tetapi memiliki fungsi yang berbeda.

**Source Code :**



Terakhir, kita akan menjalankannya pada class karnivora, sebelumnya kita harus membuat instance/objek dari masing-masing class tersebut di class karnivora, supaya kita bisa memanggil method dari class-class tersebut dan menjalankannya.

**Source Code :**



**Output :**



* 1. **Implementasi Dari Polimorfisme**

Implementasi dari polimorfime sendiri adalah Override, Overloading, dan Polimorfisme Runtime.

* **Override**

Teknik ini telah dijelaskan pada bagian sebelumnya teknik ini adalah teknik menempa atau menulis ulang method superclass pada subclass. Penggunaanya adalah dengan menulis ulang nama method pada bagian subclass.

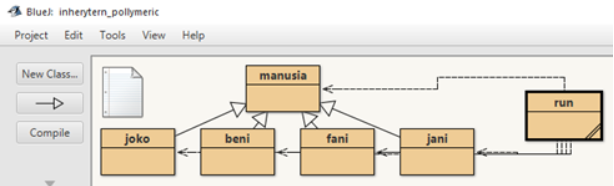
* **Overloading**

Overloading adalah kemampuan sebuah class memiliki beberapa method dengan nama yang sama, overloading bisa juga disebut Polimorfisme static. Teknik ini dapat digunakan apabila variablenya berbeda, teknik ini akan dibahas lebih lanjut nanti.

* **Polimorfisme Runtime**

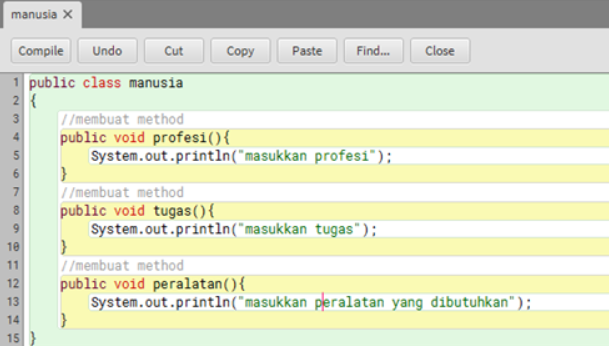
Polimorfisme Runtime seperti namanya polimorfisme ini terjadi ketika program dijalankan, dalam implementasinya menggunakan metode inheritence dan override. Dalam membuat object biasa “manusia a = new manusia” manusia yang pertama berfungsi sebagai variable dan yang berikutnya adalah object, namun sekarang object dibuat dengan variable dan object yang berberda “makhluk\_hidup a = new manusia” makhluk hidup disini adalah superclass sedangkan manusia adalah subclassnya.

**Contoh Program** :



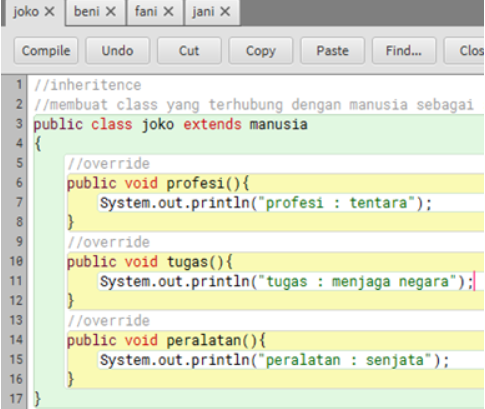
* Class manusia sebagai superclass dari class joko, beni, fani dan jani
* Run membuat object daris semua subclass manusia

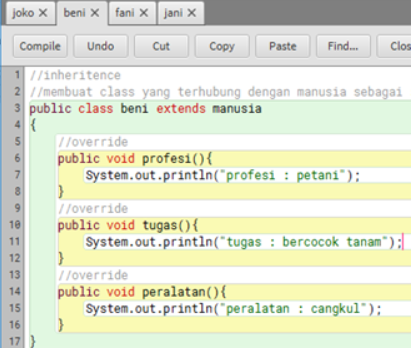
**Class Manusia :**

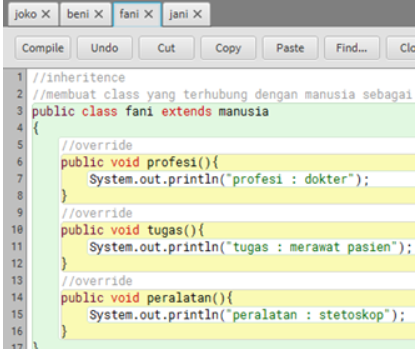


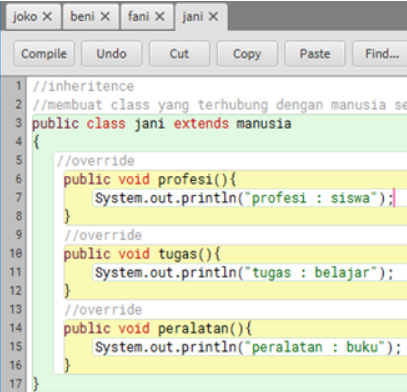
Class ini berisikan 3 method yang nantinya akan diturunkan pada subclassnya.

**Class joko, beni, fani, dan jani :**



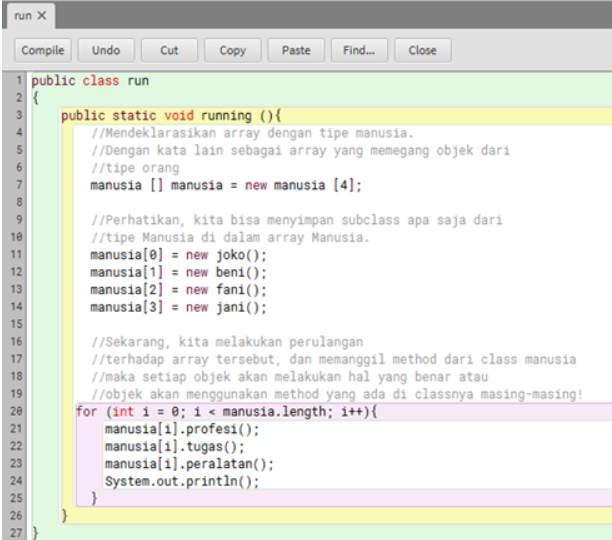






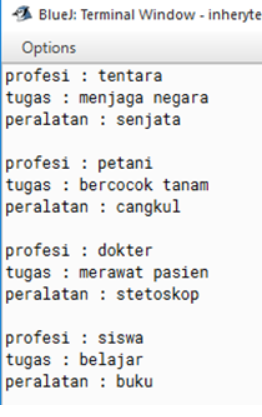
Pada ketiga class diatas semuanya adalah subclass dari class manusia namun semuanya dilakukan proses inheritence dan disesuaikan dengan kebutuhannya masing-masing.

**Class Run :**



Class ini digunakan untuk menjalankan program awalnya membuat array dengan jumlah subclass yang akan ditampilkan. Selanjutnya pembuatan object pada array yang menggunakan metode Polimorfisme Runtime. Kemudian melakukan perulangan sebanyak jumlah array dari manusia, didalam array itu kita panggil semua method yang ada pada superclass manusia.

**Running Program :**



Dapat menampilkan semua anggota subclass dari class manusia dengan kaidah polimorfisme.

* 1. **Perbedaan Diantara Jenis Polimorfisme**

Polimorfisme terbagi menjadi dua suku kata yaitu, Poly yang berarti banyak dan Morfisme yang berarti bentuk. Dalam ilmu sains, Polimorfisme (polymorphism) adalah sebuah prinsip dalam biologi di mana organisme atau spesies memiliki banyak bentuk serta tahapan (stages). Prinsip tersebut diterapkan juga pada bahasa Java. Polimorfisme dalam OOP merupakan sebuah konsep OOP di mana *class*memiliki banyak “bentuk” *method*yang berbeda, meskipun namanya sama. Maksud dari “bentuk” adalah isinya yang berbeda, namun tipe data dan parameternya berbeda.

Polimorfisme juga dapat diartikan sebagai teknik *programming* yang mengarahkan kamu untuk memprogram secara general daripada secara spesifik. Contohnya kita memiliki tiga class yang berbeda yaitu: “Kelinci”, “Kucing”, dan “Sapi”. Di mana ketiga *class* tersebut merupakan turunan dari *class* “Hewan”. Sejalan dengan contoh yang diberikan, kamu diharapkan dapat mengerti dan memahami konsep polimorfisme itu sendiri.

Polimorfisme pada Java memiliki 2 macam yaitu diantaranya:

* Static Polymorphism (Polimorfisme statis).
* Dynamic Polymorphism (Polimorfisme dinamis).

Perbedaan keduanya terletak pada cara membuat polimorfisme. Polimorfisme statis menggunakan *method overloading,* sedangkan polimorfisme dinamis menggunakan *method overriding*. Polymorphism sering dikaitkan dengan penggunaan lebih dari satu metoda dengan nama sama. Penggunaan metoda dengan nama sama dapat diterapkan dengan method overloading dan method overriding. Peran polymorphism sebenarnya tidak terbatas hanya pada hal tersebut. Ada keterkaitan antara polymorphism dan inheritance (turunan).

Dalam konsep turunan, saat obyek dari subclass dikonstruksi, obyek dari superclass juga ikut dikonstruksi. Jadi setiap instance dari subclass adalah juga instance dari superclass. Apabila Anda mendeklarasikan metoda dengan parameter dari tipe superclass, Anda diperbolehkan untuk memberi argumen berupa obyek subclass yang merupakan turunan dari superclass tersebut. Berikut ini adalah contoh program yang dapat memberikan gambaran berkaitan dengan konsep polymorphism. Perlu dipahami dan dimengerti bahwa kelas Object merupakan akar dari semua kelas Java dan menduduki puncak tertinggi dalam hirarkhi. Program akan mendefinisikan kelas yang berkaitan dengan bidang datar secara sederhana termasuk beberapa kelas turunannya (kelas PersegiPanjang dan Balok) dan membatasi hanya pada penerapan method overriding.

A picture containing text

Description automatically generated

Polymorphism menggunakan method tersebut untuk melakukan berbagai tugas. Polymorphism memungkinkan kita untuk melakukan satu tindakan dengan berbagai cara berbeda. Polymorphism yang akan dibahas pada artikel ini lebih dikenal dengan konsep method overriding.

Misalnya, Superclass bernama Hewan yang memiliki method yang disebut suaraHewan(). Subclass Hewan dapat berupa Tikus, Kucing, Anjing, Burung. Dan mereka juga memiliki implementasi suara masing masing (Tikus “citcit”, dan kucing “Weong”, dsb). Contoh :

**Source Code :**



Ingat pada tutorial **inheritance** bahwa kita menggunakan keyword extends untuk mewarisi atribut dan method dari class lain.Sekarang, kita telah membuat object Tikus dan Kucing dan Kucing memanggil method suaraHewan() untuk keduanya. Contoh:

**Source Code :**



**Output :**

Hewan bersuara Tikus bersuara : cit cit Kucing bersuara : weong

* **Mengapa dan Kapan Menggunakan Inheritance dan Polymorphisme?**

Dua konsep ini sangat berguna untuk menggunakan kembali kode yang sudah kita buat pada suatu class atau menggunakan kembali atribut dan method dari sebuah class yang telah ada ketika membuat sebuah class baru. Konsep polymorphism yang dibahas dalam artikel ini lebih dikenal dengan nama konsep method overriding.

**2.10 Keuntungan menggunakan polimorfisme**

* Dapat menggunakan kelas-kelas yang kita buat (sebagai super kelas) dan membuat kelas kelas baru berdasar superkelas tersebut dengan karakteristik yang lebih khusus dari behaviour umum yang dimiliki superkelas.
* Dapat membuat super kelas yang hanya mendefinisikan behaviuor namun tidak memberikan implementasidari metode-metode yang ada.Hal ini berguna jika ingin membuat template kelas, kelas ini disebut kelas abstrak karena behaviournya masih abstrak dan belum diimplementasikan. Subkelas-subkelas dari kelas ini yang disebut kelas konkret, mengimplementasikan behaviuor abstrak tersebut sesuai dengan kebutuhan masing-masing.
* Menghindari duplikasi object, yang dapat menciptakan class baru dari class yang sudah ada, sehingga tidak perlu menuliskan code dari nol ataupun mengulangnya, namun tetap bisa menambahkan attribute dan atau method unik dari class itu sendiri. Dalam konsep yang lebih umum sering kali polymorphism disebut dalam istilah satu interface banyak aksi.
* Polymorphism dapat berarti banyak bentuk, maksudnya yaitu dapat menimpa (override), suatu method, yang berasal dari parent class (super class) dimana object tersebut diturunkan, sehingga memiliki kelakuan yang berbeda.

**BAB III**

**PENUTUP**

* 1. **Kesimpulan**

Polimorfisme erat kaitannya dengan pewarisan. Polimorfisme adalah pemikiran bahwa objek dinamis suatu kelas dasar dapat berperilaku seperti kelas turunan. Ketika objek tersebut menunjuk kelas dasar , objek tersebut berperilaku seperti kelas dasar , tetapi ketika objek tersebut menunjuk kelas turunan objek tersebut berperilaku seperti kelas turunan .Dalam hal ini objek dapat memiliki beberapa bentuk , tergantung pada saat itu kelas mana yang ditunjuk. Yang perlu menjadi catatan, bahwa perubahan perilaku ini dari kelas dasar kepada kelas turunan , tidak dapat objek turunan menunjuk kelas dasar.

Istilah pewarisan berarti suatu kelas bisa mewariskan sebagian atau keseluruhan struktur dan perilaku kelas lain. Jika kelas B adalah kelas turunan dari kelas A, maka kita bisa juga menyebut kelas A adalah kelas super dari kelas B. Kelas turunan bisa memiliki struktur atau perilaku tambahan dari kelas supernya. Atau bahkan kelas turunan bisa mengubah atau mengganti perilaku kelas supernya. Hubungan antara kelas turunan dan kelas super sering dilukiskan dalam bentuk diagram di mana kelas turunan digambarkan di bawah kelas supernya, dan dihubungkan dengan garis penghubung dengan tanda segitiga yang diletakkan di dekat kelas supernya.

**DAFTAR PUSTAKA**

1. <https://www.tutorialcup.com/id/java/polymorphism-in-java.htm>
2. <https://nextgen.co.id/contoh-polimorfisme-java/#:~:text=Java%20Polymorphism,lainnya%20melalui%20inheritance%20(pewarisan)>
3. <https://sis.binus.ac.id/2017/08/30/polymorphism/>
4. <https://www.dicoding.com/blog/pengertian-polimorfisme-dalam-pemrograman-java/>