**LAPORAN TUGAS**

**STRUKTUR DATA**

**PENGENALAN GENERICS**



**Dosen Pengampu:**

**Khairul Umam, S.Pd, M.Kom.**

**Oleh:**

**Nama : Andri Wijaksono**

**NIM : 362155401206**

**Kelas : 1G**

**PROGRAM STUDI DIPLOMA III**

**TEKNIK INFORMATIKA**

**POLITEKNIK NEGERI BANYUWANGI**

**2021**

Buatlah *resume* terkait konsep **generics** pada Java:

1. Jelaskan apa yang dimaksud dengan **generics**pada Java!
2. Bagaimana mekanisme penerapan konsep generics pada ***class***?
3. Bagaimana mekanisme penerapan konsep generics pada ***method***?
4. Apa pengaruh penggunaan operator **extends**pada generics *class*dan *method*?
5. Apa pengaruh penggunaan operator **super**pada generics *class*dan *method*?
6. Apa kegunaan operator **wildcard ?** pada konsep generics?

Resume

1. sebuah kode yang dapat digunakan oleh beberapa objek tipe data seperti string, integer dan float. dengan kata lain dapat menampung sebuah tipe data tersebut
2. Pendeklarasian Generic Class ditandai dengan simbol yang diawali tanda (<) dan diakhiri tanda (>), setelah nama Kelas :

***class Koleksi<T>***

Dari kode diatas, berarti kita membuat simbol generic dengan simbol T, sehingga T dianggap tipe data dalam lingkup kelas tersebut. Saat pendeklarasian objek Koleksi, maka kita harus menentukan tipe T tersebut :

***Koleksi<Integer> a = new Koleksi<Integer>();***

Kode diatas berarti kita mengganti simbol T dengan tipe data integer.

Tipe data harus diimplementasikan dalam objek, misal int menjadi Integer, double menjadi Double, string menjadi String.

1. Generic Programming dalam sebuah Metode, hanya berlaku untuk metode tersebut, tidak berlaku untuk metode yang lain dalam kelas yang sama.

**public class GenericMethodTest**

**{**

**// generic method printArray**

**public static < E > void printArray( E[] inputArray )**

**{ for ( E element : inputArray )**

**System.out.printf( "%s ", element );**

**System.out.println();**

**}**

**public static void main( String args[] )**

**// create arrays of Integer, Double and Character**

**Integer[] intArray = { 1, 2, 3, 4, 5 };**

**Double[] doubleArray = { 1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7 };**

**Character[] charArray = { 'H', 'E', 'L', 'L', 'O' };**

**printArray( integerArray ); // pass an Integer array**

**printArray( doubleArray ); // pass a Double array**

**printArray( characterArray ); // pass a Character array**

**}**

**}**

Perhatikan bahwa tipe data harus diimplementasikan dalam objek,

misal

int menjadi Integer

double menjadi Double

char menjadi Character.

1. Kata kunci extends digunakan untuk memperluas kelas (menunjukkan bahwa kelas diwarisi dari kelas lain).

Di Java, dimungkinkan untuk mewarisi atribut dan metode dari satu kelas ke kelas lainnya. Kami mengelompokkan "konsep pewarisan" ke dalam dua kategori:

subclass (anak) - kelas yang mewarisi dari superclass(kelas lain)

superclass (induk) - kelas yang diwarisi dari subclass

Untuk mewarisi dari kelas, gunakan kata kunci extends.

**class Vehicle {**

**protected String brand = "Ford"; // Vehicle attribute**

**public void honk() { // Vehicle method**

**System.out.println("Tuut, tuut!");**

**}**

**}**

**class Car extends Vehicle {**

**private String modelName = "Mustang"; // Car attribute**

**public static void main(String[] args)**

Kelas Car (subclass) mewarisi atribut dari metode kelas Vehicle (superclass):

1. Kata kunci super mengacu pada objek superclass (induk).

Ini digunakan untuk memanggil metode superclass, dan untuk mengakses konstruktor superclass.

Penggunaan paling umum dari kata kunci super adalah untuk menghilangkan kebingungan antara superclass dan subclass yang memiliki metode dengan nama yang sama.

**class Animal { // Superclass (induk)**

**public void animalSound() {**

**System.out.println("The animal makes a sound");**

**}**

**}**

**class Dog extends Animal { // Subclass (anak)**

**public void animalSound() {**

**super.animalSound(); // memanggil metode superclass**

**System.out.println("The dog says: bow wow");**

**}**

**}**

1. Tanda tanya (?) dikenal sebagai wildcard dalam pemrograman generik. Ini mewakili jenis yang tidak diketahui. Wildcard dapat digunakan dalam berbagai situasi seperti jenis parameter, bidang, atau variabel lokal; terkadang sebagai tipe pengembalian. Tidak seperti array, instantiasi yang berbeda dari tipe generik tidak kompatibel satu sama lain, bahkan secara eksplisit. Ketidakcocokan ini dapat dilunakkan oleh wildcard jika ? digunakan sebagai parameter tipe aktual

1. Wildcard Batas Atas:

ini dapat digunakan saat Anda ingin melonggarkan batasan pada variabel. Misalnya, Anda ingin menulis metode yang berfungsi pada Daftar < Bilangan Bulat >, Daftar < Ganda >, dan Daftar < Angka >, Anda dapat melakukannya menggunakan wildcard berbatas atas.Untuk mendeklarasikan wildcard batas atas, gunakan karakter wildcard ('?'), diikuti oleh kata kunci extends, diikuti dengan batas atasnya.

**// Program Java untuk mendemonstrasikan Wildcard dengan Batas Atas**

**import java.util.Arrays;**

**import java.util.List;**

**class WildcardDemo {**

**public static void main(String[] args)**

**{**

**// Daftar Bilangan Bulat Berbatas Atas**

**List<Integer> list1 = Arrays.asList(4, 5, 6, 7);**

**// mencetak jumlah elemen dalam daftar**

**System.out.println("Total sum is:" + sum(list1));**

**// Daftar ganda**

**List<Double> list2 = Arrays.asList(4.1, 5.1, 6.1);**

**// mencetak jumlah elemen dalam daftar**

**System.out.print("Total sum is:" + sum(list2));**

**}**

**private static double sum(List<? extends Number> list)**

**{**

**double sum = 0.0;**

**for (Number i : list) {**

**sum += i.doubleValue();**

**}**

**return sum;**

**}**

**}**

Output:

Total sum is:22.0

Total sum is:15.299999999999999

2. Wildcard dengan Batas Bawah

Itu dinyatakan menggunakan karakter wildcard ('?'), diikuti oleh kata kunci super, diikuti dengan batas bawahnya: <? super A>.

**// Program Java untuk mendemonstrasikan Wildcard dengan Batas Atas**

**import java.util.Arrays;**

**import java.util.List;**

**class WildcardDemo {**

**public static void main(String[] args)**

**{**

**// Daftar Bilangan Bulat Berbatas Atas**

**List<Integer> list1 = Arrays.asList(4, 5, 6, 7);**

**// Objek daftar bilangan bulat sedang diteruskan**

**printOnlyIntegerClassorSuperClass(list1);**

**// Daftar nomor**

**List<Number> list2 = Arrays.asList(4, 5, 6, 7);**

**// Objek daftar bilangan bulat sedang diteruskan printOnlyIntegerClassorSuperClass(list2);**

**}**

**public static void printOnlyIntegerClassorSuperClass(**

**List<? super Integer> list)**

**{**

**System.out.println(list);**

**}**

**}**

Output:

[4, 5, 6, 7]

[4, 5, 6, 7]

3. Wildcard Tanpa Batas:

Jenis wildcard ini ditentukan menggunakan karakter wildcard (?), misalnya, Daftar. Ini disebut daftar jenis yang tidak diketahui. Ini berguna dalam kasus berikut -

Saat menulis metode yang dapat digunakan menggunakan fungsionalitas yang disediakan di kelas Object.

Ketika kode menggunakan metode di kelas generik yang tidak bergantung pada parameter tipe

**// Program Java untuk mendemonstrasikan wildcard tanpa batas**

**import java.util.Arrays;**

**import java.util.List;**

**class unboundedwildcardemo {**

**public static void main(String[] args)**

**{**

**// Daftar Bilangan Bulat**

**List<Integer> list1 = Arrays.asList(1, 2, 3);**

**// Daftar Ganda**

**List<Double> list2 = Arrays.asList(1.1, 2.2, 3.3);**

**printlist(list1);**

**printlist(list2);**

**}**

**private static void printlist(List<?> list)**

**{**

**System.out.println(list);**

**}**

**}**

Output:

[1, 2, 3]

[1.1, 2.2, 3.3]