**JAVA: Fundamental**

Makalah ini ditujukan untuk *myselft.*

*Jikalau saya* ***lupa*** *dan ingin* ***re-read*** *this* ***poetry****.*

Apa isi di dalam buku ini?

|  |
| --- |
| * Penjelasan fundamental/dasar-dasar java. |
| * Disertai dengan contoh program yang have fun! |

Bandung, 10 February 2023

Fachri Rizki Anugrah

Kata Pengantar

Dengan hormat,

Puji syukur kehadirat Tuhan Yang Maha Esa, karena berkat bantuannya maka penulis dapat menyelesaikan makalah yang berjudul "Java: Fundamental". Makalah ini dibuat sebagai salah satu Keinginan penulis untuk menulis makalah, jikalau penulis lupa dan ingin membaca-baca lagi atau mungkin sedang GABUT saja.

Makalah ini membahas mengenai bahasa pemrograman Java, termasuk sejarah, kelebihan dan kekurangan, serta cara penggunaannya. Java adalah salah satu bahasa pemrograman yang populer dan digunakan dalam berbagai bidang, seperti pengembangan aplikasi desktop, mobile, web, dan lain-lain.

Penulis berharap makalah ini dapat membantu pembaca untuk memahami bahasa pemrograman Java dan membantu dalam proses belajar dan mengaplikasikannya.

Akhir kata, penulis menyadari bahwa masih banyak kekurangan dalam makalah ini. Oleh karena itu, penulis sangat terbuka terhadap saran dan kritik yang membangun demi perbaikan kualitas makalah ini di masa yang akan datang.

Penulis menyadari bahwa makalah ini tidak dapat diselesaikan tanpa bantuan dan dukungan dari berbagai pihak yaitu keinginan penulis sendiri yang ingin menulis makalah ini. Oleh karena itu, penulis mengucapkan terima kasih kepada diri saya sendiri yang memberikan tekad dan ketekunan dalam membuat makalah sederhana ini, serta kepada semua pihak yang telah membantu dalam penyusunan makalah ini.

Penulis berharap makalah ini dapat bermanfaat bagi pembaca dan menjadi sumbangan yang berguna bagi perkembangan ilmu pengetahuan.

Terima kasih.

Salam, Fachri Rizki Anugrah

**BAB I PENDAHULUAN**

1. apa itu java?

Java adalah bahasa pemrograman yang dikembangkan oleh Sun Microsystems (sekarang dimiliki oleh Oracle Corporation) yang dirancang untuk membuat aplikasi yang bekerja di berbagai perangkat, termasuk komputer desktop, ponsel, dan web.

Bahasa pemrograman java salah satu bahasa pemrograman yang berorientasi objek. yang memfokuskan pada pemodelan objek-objek dunia nyata dan interaksi antar objek.

1. Sejarah Singkat

Java pertama kali dikembangkan oleh James Gosling dan timnya pada tahun 1990-an. Awalnya Java dikenal sebagai "Oak", namun nama itu kemudian diganti menjadi Java pada tahun 1995. Versi pertama Java dirilis pada tahun 1996 dan sejak itu, bahasa ini terus berkembang dan menjadi salah satu bahasa pemrograman yang paling populer dan banyak digunakan.

1. perbedaan **Jvm, jre, jdk**

JVM (Java Virtual Machine), JRE (Java Runtime Environment) dan JDK (Java Development Kit) adalah bagian dari platform Java yang berbeda-beda.

1. JVM (Java Virtual Machine) JVM adalah mesin virtual yang menjalankan kode Java. Ia memastikan bahwa kode Java dapat berjalan di berbagai sistem operasi tanpa perlu memperhatikan perbedaan sistem operasi tersebut. JVM bertanggung jawab untuk membuat kode Java menjadi kode mesin yang dapat diterima oleh sistem operasi dan mengelola memori serta tugas-tugas lainnya yang berhubungan dengan eksekusi kode Java.
2. JRE (Java Runtime Environment) JRE adalah lingkungan runtime yang berisi JVM dan komponen lain yang dibutuhkan untuk menjalankan program Java. Ini meliputi kumpulan bibliotek yang digunakan oleh program Java, antarmuka aplikasi dan komponen lain yang dibutuhkan untuk memastikan bahwa program Java berjalan dengan baik.
3. JDK (Java Development Kit) JDK adalah kit pengembangan yang berisi alat-alat dan sumber daya yang dibutuhkan oleh pengembang untuk membuat dan menguji aplikasi Java. Ini termasuk JRE, kompiler Java, debugger, dan alat-alat lain yang dibutuhkan untuk mengembangkan aplikasi Java.

Singkatnya, JVM adalah **mesin virtual** yang menjalankan kode Java, JRE adalah lingkungan **runtime** yang menjalankan program Java, dan JDK adalah kit pengembangan yang memungkinkan pengembang untuk **membuat dan menguji aplikasi** Java.

**BAB II PEMBAHASAN**

1. JAVA: konsep dasar

1. OOP (Object-Oriented Programming) Java adalah bahasa pemrograman berorientasi objek, yang berarti bahwa semua elemen dalam bahasa ini didesain berdasarkan objek-objek. Ini membuat kode lebih terstruktur dan mudah dipahami.

2. Class and Object Konsep dasar dalam OOP adalah class dan object. Class adalah template yang digunakan untuk membuat objek, sedangkan objek adalah instance dari class. Class mencakup atribut dan metode yang memodelkan perilaku dan informasi suatu objek.

3. Inheritance Inheritance adalah mekanisme dimana suatu class dapat mewarisi atribut dan metode dari class lain. Ini membuat kode lebih efisien dan memungkinkan pembuatan hierarki kelas.

4. Polymorphism Polymorphism adalah konsep dimana suatu objek dapat memiliki beberapa bentuk. Ini dapat dilakukan dengan menggunakan metode yang memiliki nama yang sama tetapi memiliki implementasi yang berbeda.

5. Encapsulation Encapsulation adalah mekanisme dimana atribut dan metode suatu objek dapat dikelompokkan bersama-sama dan dibuat tidak dapat diakses secara langsung dari luar kelas. Ini membuat kode lebih aman dan memastikan bahwa tidak ada yang dapat memodifikasi atribut atau metode yang tidak perlu.

6. Java API (Application Programming Interface) Java API adalah kumpulan class dan metode yang dapat digunakan oleh pengembang untuk membuat aplikasi. Ini membuat pengembangan aplikasi lebih mudah dan mempercepat proses pengembangan.

7. Java SE (Standard Edition) Java SE adalah paket utama dari Java yang menyediakan dasar-dasar bahasa dan API. Ini memungkinkan pengembang untuk membuat aplikasi desktop, server, dan aplikasi embedded.

Dengan memahami dasar-dasar ini, Anda dapat memulai pembelajaran Java dan membuat aplikasi sendiri.

1. JAVA: syntax dasar

1. Tipe Data Java memiliki berbagai tipe data, seperti integer, floating-point, Boolean, dan string, yang digunakan untuk menyimpan dan memproses informasi.

2. Variabel Variabel adalah nama yang digunakan untuk menyimpan informasi. Tiap variabel harus memiliki tipe data dan harus dideklarasikan sebelum digunakan.

3. Operator Java memiliki berbagai operator, seperti aritmatika, logika, dan perbandingan, yang digunakan untuk memproses informasi.

Kontrol Alir Java memiliki berbagai kontrol alir, seperti perulangan dan pemilihan, yang memungkinkan pengembang untuk menentukan bagaimana kode harus berjalan.

4. Method Method adalah blok kode yang dapat digunakan berkali-kali. Ini membuat kode lebih efisien dan memungkinkan pembuatan kode yang lebih mudah dipahami.

5. Kelas Kelas adalah unit utama dalam Java. Ini memungkinkan pengembang untuk membuat objek yang memiliki atribut dan metode.

6. Paket Paket adalah kumpulan kelas dan metode yang digunakan untuk membuat aplikasi. Ini membuat pengembangan aplikasi lebih mudah dan mempercepat proses pengembangan.

7. Komentar Komentar adalah teks yang tidak dieksekusi oleh kompiler dan digunakan untuk memberikan dokumentasi dan keterangan pada kode.

Contoh Program:

1. Tipe data:

“

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 20;

int c = a + b;

System.out.println("Hasil dari a + b adalah: " + c);

}

}

“

1. Variable:

“

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 20;

int c = a + b;

System.out.println("Hasil dari a + b adalah: " + c);

}

}

“

1. Operasi Math:

“

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 20;

int c = a + b;

System.out.println("Hasil dari a + b adalah: " + c);

}

}

“

1. Kondisi dan percabangan:

“

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 20;

if (a > b) {

System.out.println("a lebih besar dari b");

} else {

System.out.println("b lebih besar dari a");

}

}

}

“

1. Looping:

“

public class Main {

public static void main(String[] args) {

// For loop

for (int i = 1; i <= 10; i++) {

System.out.println(i);

}

// while loop

int i = 0;

while (i < 5) {

System.out.println(i);

i++;

}

// DO-WHILE

int i = 0;

do {

System.out.println(i);

i++;

} while (i < 5);

}

}

“

Java: Continue and Break;

1. Break

"Break" adalah pernyataan yang digunakan dalam Java untuk keluar dari suatu loop sebelum waktunya. Pernyataan ini sering digunakan bersama dengan pernyataan kontrol seperti "if" atau "switch" untuk mengontrol aliran program.

Sebagai contoh, jika Anda memiliki loop yang mengiterasi daftar nilai, Anda dapat menggunakan pernyataan "break" untuk keluar dari loop ketika kondisi tertentu terpenuhi. Berikut ini adalah contohnya:

for (int i = 0; i < 10; i++) {

if (i == 5) {

break;

}

System.out.println(i);

}

Dalam contoh ini, loop akan berjalan sebanyak 10 kali, tetapi akan berhenti dan keluar ketika nilai i sama dengan 5. Output dari program adalah:

OUTPUT:

0

1

2

3

4

Perlu diingat bahwa pernyataan "break" hanya akan keluar dari loop terdalam yang berada di dalamnya. Jika Anda memiliki loop bersarang, Anda perlu menggunakan label untuk menentukan loop mana yang ingin Anda keluarkan. Berikut adalah

contohnya:

outerLoop:

for (int i = 0; i < 10; i++) {

for (int j = 0; j < 10; j++) {

if (j == 5) {

break outerLoop;

}

System.out.println("i: " + i + " j: " + j);

}

}

1. Continue

Pernyataan "continue" dalam Java digunakan untuk melanjutkan iterasi loop selanjutnya tanpa menjalankan perintah di bawahnya. Ini sering digunakan dengan loop seperti "for" atau "while".

Berikut adalah contoh penggunaan pernyataan "continue" dalam loop "for":

for (int i = 0; i < 10; i++) {

if (i == 5) {

continue;

}

System.out.println(i);

}

Dalam contoh ini, ketika i bernilai 5, perintah continue akan dieksekusi dan iterasi loop selanjutnya akan dilanjutkan tanpa menjalankan perintah System.out.println(i). Hasil dari program akan menjadi:

Output:

0

1

2

3

4

6

7

8

9

Seperti pada pernyataan "break", pernyataan "continue" juga hanya akan memengaruhi loop terdalam yang berada di dalamnya. Jika Anda memiliki loop bersarang, Anda perlu menggunakan label untuk menentukan loop mana yang ingin Anda lanjutkan. Berikut adalah contohnya:

outerLoop:

for (int i = 0; i < 10; i++) {

for (int j = 0; j < 10; j++) {

if (j == 5) {

continue outerLoop;

}

System.out.println("i: " + i + " j: " + j);

}

}

Dalam contoh ini, ketika j bernilai 5, perintah continue outerLoop akan dieksekusi dan iterasi loop luar selanjutnya akan dilanjutkan tanpa menjalankan perintah di dalam loop. Hasilnya akan menjadi:

OUTPUT:

i: 0 j: 0

i: 0 j: 1

i: 0 j: 2

i: 0 j: 3

i: 0 j: 4

i: 1 j: 0

i: 1 j: 1

...

**JAVA Method/Function**

Java Method adalah blok kode yang dapat digunakan untuk menjalankan tugas tertentu. Method dapat menerima parameter dan mengembalikan nilai. Method membantu mengurangi duplikasi kode dan membuat kode lebih terstruktur dan mudah dipelihara.

Berikut adalah contoh sederhana dari metode Java:

“

public static int sum(int a, int b) {

return a + b;

}

“

Dalam contoh ini, method sum() menerima dua parameter, a dan b, dan mengembalikan nilai a + b. Anda dapat memanggil metode ini dari bagian lain dari kode Anda seperti ini:

“

int result = sum(5, 7);

System.out.println("The sum is: " + result);

“

Method dapat juga tidak mengembalikan nilai, dan dapat dideklarasikan dengan menggunakan tipe **void** seperti ini:

“

public static void printMessage(String message) {

System.out.println(message);

}

“

Anda dapat memanggil metode ini dengan mengirimkan string sebagai parameter seperti ini:

“

printMessage("Hello World!");

“

Menggunakan metode dapat membuat kode lebih terstruktur, mudah dipelihara, dan mudah dibaca. Ini juga membantu mengurangi duplikasi kode dan mempermudah proses debugging dan testing.

**JAVA RECURSION**

Rekursif Method adalah metode yang memanggil dirinya sendiri untuk menyelesaikan masalah. Ini adalah salah satu teknik pemrograman yang berguna untuk memecahkan masalah yang kompleks menjadi submasalah yang lebih sederhana.

Berikut adalah contoh sederhana dari rekursif method untuk menghitung faktorial:

public static int factorial(int n) {

if (n == 0) {

return 1;

} else {

return n \* factorial(n-1);

}

}

Dalam contoh ini, method factorial() memanggil dirinya sendiri dengan parameter n-1 setiap kali itu dipanggil. Dalam kasus ini, base case adalah ketika n bernilai 0, dan metode mengembalikan nilai 1. Dalam semua kasus lain, metode mengembalikan n dikalikan dengan hasil dari pemanggilan rekursif metode factorial dengan n-1 sebagai parameter.

Rekursif method sangat berguna untuk memecahkan masalah yang kompleks dan membuat kode lebih mudah dibaca. Namun, penting untuk memastikan bahwa Anda memiliki base case dan untuk memastikan bahwa setiap pemanggilan rekursif tidak akan menyebabkan infinite loop.

**JAVA ARRAY**

Array dalam Java adalah tipe data yang digunakan untuk menyimpan beberapa nilai dalam satu variabel. Berikut adalah dokumentasi dasar tentang array dalam Java:

Deklarasi Array: Anda dapat menentukan tipe data dan ukuran array seperti ini:

tipe\_data[] nama\_array = new tipe\_data[ukuran\_array];

Inisialisasi Array: Anda dapat mengisi nilai dalam array saat pembuatan atau setelah pembuatan.

tipe\_data[] nama\_array = {nilai1, nilai2, ...};

Mengakses Elemen Array: Anda dapat mengakses elemen array dengan menggunakan indeks. Indeks dimulai dari 0.

nama\_array[indeks] = nilai\_baru;

Looping Array: Anda dapat menggunakan loop untuk mengakses setiap elemen dalam array.

for (int i = 0; i < nama\_array.length; i++) {

System.out.println(nama\_array[i]);

}

Panjang Array: Anda dapat menggunakan properti length untuk menentukan panjang array.

int panjang = nama\_array.length;

ARRAY 2D

Array 2D adalah tipe data array di Java yang memiliki dua indeks untuk mengakses elemen-elemen di dalamnya. Ini berarti bahwa setiap elemen dalam array dapat diidentifikasi oleh baris dan kolom. Array 2D biasanya digunakan untuk menyimpan dan mengelola data tabular, seperti tabel, dengan baris dan kolom.

Berikut ini adalah contoh program sederhana untuk menunjukkan bagaimana menggunakan array 2D di Java:

public class Main {

public static void main(String[] args) {

// Deklarasi array 2D dengan 3 baris dan 4 kolom

int[][] arr = new int[3][4];

// Inisialisasi elemen-elemen dalam array 2D

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 4; j++) {

arr[i][j] = i \* 4 + j;

}

}

// Mencetak isi array 2D

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 4; j++) {

System.out.print(arr[i][j] + " ");

}

System.out.println();

}

}

}

**LATIHAN ARRAY 2D**

Berikut adalah contoh source code beberapa latihan program menggunakan array 2D di Java:

1. **Mencari elemen dari yang terbesar:**

public class Main {

public static void main(String[] args) {

int[][] arr = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};

int largest = arr[0][0];

for (int i = 0; i < arr.length; i++) {

for (int j = 0; j < arr[i].length; j++) {

if (arr[i][j] > largest) {

largest = arr[i][j];

}

}

}

System.out.println("The largest element is: " + largest);

}

}

1. **Mencari elemen dari yang terkecil:**

public class Main {

public static void main(String[] args) {

int[][] arr = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};

int smallest = arr[0][0];

for (int i = 0; i < arr.length; i++) {

for (int j = 0; j < arr[i].length; j++) {

if (arr[i][j] < smallest) {

smallest = arr[i][j];

}

}

}

System.out.println("The smallest element is: " + smallest);

}

}

1. **Menjumlahkan semua elemen:**

public class Main {

public static void main(String[] args) {

int[][] arr = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};

int sum = 0;

for (int i = 0; i < arr.length; i++) {

for (int j = 0; j < arr[i].length; j++) {

sum += arr[i][j];

}

}

System.out.println("The sum of all elements is: " + sum);

}

}

1. **Mencari rata-rata semua elements:**

public class Main {

public static void main(String[] args) {

int[][] arr = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};

int sum = 0;

int count = 0;

for (int i = 0; i < arr.length; i++) {

for (int j = 0; j < arr[i].length; j++) {

sum += arr[i][j];

count++;

}

}

double average = (double) sum / count;

System.out.println("The average of all elements is: " + average);

}

}