**HADIAH 1**

**Pemrograman Berorientasi Objek**



Fakhita Nur Aziza

51019007

**PRODI SISTEM INFORMASI**

**STMIK KHARISMA MAKASSAR**

**2021**

**Soal**

1. 50 pesan error di java (error tentang apa berikan penjelasan).
2. 30 file header java (import java.io. ? ), (file header itu fungsinya untuk apa ?).
3. Format penulisan Bahasa pemrograman java.

**Jawaban**

1. 50 pesan error di java :
2. ***Main method not found in class TestApel, please define the main method as:  
   public static void main (String[] args)***: kesalahan penulisan pada head method
3. ***BOOLEAN cannot be resolved to a type*** *:* menulis keyword dengan huruf kapital
4. ***Syntax error on token “,”, delete this token* *Type mismatch: cannot convert from int[][] to int[]*** *:* mengasumsikan array 2 dimensi langsung diimplementasikan pada java.
5. ***The constructor Test (String, int) is undefined*** *:* terbalik dalam mengisi parameter konstruktor.
6. ***The constructor Test (double) is undefined*** *:* menggunakan konstruktor yang tidak ada
7. ***The method size () is undefined for the type String*** *:* menggunakan method yang tidak dimiliki class
8. ***The method hitungJarak (int, double) in the type Test is not applicable for the arguments (double, int)*** *:* terbalik ketika mengisi method)
9. ***The local variable x may not have been initialized*** *:* menggunakan variable sebelum diberi nilai.
10. ***The field Apel.berat is not visible*** *:* mengakses variable instance private melalui class lain.
11. ***Syntax error, insert “}” to complete ClassBody*** *:* lupa memberikan tanda } pada akhir class body.
12. ***The method printWarnaDanBerat() is undefined for the type TestApel*** *:* kesalahan menulis format untuk method class.
13. **Variabel dengan nama panjang, tidak sama dengan Panjang**. Jadi penulisan variable harus benar-benar diperhatikan. kesalahan dalam penulisan variable.
14. ***String can not be resolved to a type*** *:* kesalahan dalam penulisan class ketika mendeklarasikan variable.
15. ***Cannot invoke abs() on the primitive type int*** *:* menggunakan method static seperti method objek.
16. ***Scanner cannot be resolved to a type*** *:* lupa untuk mengimport package.
17. *String literal is not properly closed by a double-quote :* menulis string pada baris yang baru.
18. ***Return type for the method is missing*** *:* tidak menggunakan keyword void untuk method yang tidak mengembalikan nilai.
19. ***This method must return a result of type double*** *:* lupa menggunakan keyword return pada value returning method.
20. ***Incompatible types : String cannot be converted to int*** *:* tipe data string tidak diizinkan untuk di convert ke integer.
21. ***Expected*** *:* ada symbol yang kurang atau tertinggal.
22. ***;' excepted" dan "incompatible types : int cannot be converted to String*** *:* tidak ada simbol semicolon atau penutup baris setiap kode program dan konvert dari int ke string, tetapi ini masalah berbeda dari yang sebelumnya.
23. ***Arithmetic Exception*** : untuk perhitungan aritmatika, misalkan sebuah angka di bagi0
24. ***Null Pointer Exception*** : bila objek yang ingin diakses masih NULL, maka tidak bisa akses method dari objek tersebut.
25. ***Number Format Exception*** : bila ingin melakukan casting tipe data string ke int maka akan terjadi kesalahan ini.
26. ***IO Exception*** : bila yang diminta adalah angka, tapi diinputkan berupa string.
27. ***Illegal Argument Exception*** : mengirimkan parameter yang tidak sesuai.
28. ***Runtime Exception*** : program sedang berjalan dan tidak memberikan compilation.
29. ***Illegal State Exception*** : mencoba memanggil method di waktu yang salah.
30. ***No Such Method Exception*** : mencoba memanggil method yang tidak ada.
31. ***Class Cast Exception :*** mencoba mengubah sebuah class menjadi class lain yang bukan “instance-nya”.
32. **Parse Exception** : mencoba mengubah string menjadi data tertentu tapi string tersebut tidak terformat dengan benar.
33. ***Invocation Target Exception*** : method atau konstruktor yang kita panggil menyebabkan suatu exception terpanggil.
34. ***Index Out Of Brounds Exception*** : beberapa jenis indeks diluar batas.
35. ***Negative Array Size Exception*** : array diciptakan dengan ukuran negatif.
36. ***Array Store Exception*** : penyimpanan array dengan tipe data tidak sesuai.
37. **Security Exception** : aturan security yang dilanggar.
38. ***…class… is public, it should be declared in a file named ….java :* nama class tidak sesuai dengan nama file nya.**
39. ***Javac : no such file or directory (untuk lingkungan linux)*** : ada tiga kemungkinan: JDK belum ter-install , jika sudah menginstallnya, kemungkinan variabel PATH sistem belum ter-setting dengan benar, jika tidak ada permasalahan dengan variabel PATH, kemungkinan mengalami permasalahan instalasi (misalnya: konflik dengan program lain)
40. ***Unclosed string literal*** *:* biasanya muncul apabila kita lupa menutup string (“).
41. ***Cannot find symbol*** *:* biasanya karena perbedaan penulisan pada pendeklarasian variabel dengan penulisan kode.
42. ***Missing return statement*** *:* akan muncul apabila metode tidak mengembalikan nilai apapun.
43. ***Possible loss of precision*** *:* pesan akan muncul apabila informasi yang masuk lebih banyak daripada yang dapat disimpan. Biasanya juga dikarenakan penginputan bilangan riil pada variabel dengan tipe integer.
44. ***Unreachable statement*** *:* apabila sebuah perintah yang kita tulis berada di tempat yang tidak bisa dijangkau untuk di eksekusi, biasanya karena penempatannya setelah break atau pengembalian statement.
45. ***Array Index Out Of Bounds Exception*** : Ini adalah pesan galat runtime yang terjadi saat kode mencoba mengakses indeks array yang tidak berada dalam nilai. Indeks array dimulai dari nol dan berakhir pada satu kurang dari panjang array. Seringkali diperbaiki dengan menggunakan “<” daripada “<=” saat mendefinisikan batas indeks array.
46. ***String Index Out Of Bounds Exception*** : terjadi saat kode mencoba mengakses bagian dari string yang tidak berada dalam batas string. Biasanya, ini terjadi ketika kode mencoba membuat substring dari string yang panjangnya tidak sama dengan parameter.
47. ***Bad magic number*** *:* Pesan kesalahan perangkat lunak Java ini berarti ada yang salah dengan file definisi kelas di jaringan.
48. ***Broken pipe*** *:* Pesan kesalahan ini mengacu pada aliran data dari file atau soket jaringan yang telah berhenti bekerja atau ditutup dari ujung lainnya.
49. ***Class file contains wrong class*** *:* Masalah "file kelas berisi kelas yang salah" terjadi ketika kode Java mencoba menemukan file kelas di direktori yang salah, menghasilkan pesan kesalahan.
50. ***NoInitial Context Exception*** : Kesalahan "NoInitialContextException" terjadi ketika aplikasi Java ingin melakukan operasi penamaan tetapi tidak dapat membuat koneksi.
51. ***SSL Exception*** : Pesan kesalahan perangkat lunak Java ini terjadi ketika ada kegagalan dalam operasi terkait SSL.
52. 30 file header java :

* Interface Summary

1. ***Closeable*** : Closeable adalah sumber atau destinasi dari data yang dapat di tutup
2. ***DataInput*** : Interfacenya menyediakan pembacaan byte dari aliran biner dan merekonstruksi dari data dalam semua tipe primitive Java
3. ***DataOutput*** : Interfacenya menyediakan pengkonversian data dari semua tipe primitive Java ke kumpulan seri dari byte dan menuliskannya ke dalam aliran biner
4. ***Externalizable*** : Externalizable hanya menyimpan identitas kelas dan menuliskannya ke dalam serialization stream, dan merupakan tanggung jawab dari kelas untuk menyimpan dan memulihkan konten dari contohnya
5. ***FileFilter*** : Sebuah filter untuk pathname yang abstrak
6. ***FilenameFilter*** : Contoh kelas yang mengimplementasikan antarmuka ini digunakan untuk memfilter nama file.
7. ***Flushable*** : Merupakan destinasi dari data yang dapat di-flush
8. ***Object Input*** : ObjectInput memperluas interface dari DataInput untuk menyertakan pembacaan objek
9. ***ObjectInputValidation*** : Panggil balik interface yang memperbolehkan validasi dari objek didalam grafik
10. ***ObjectOutput*** : ObjectOutput memperluas interface dari DataOutput untuk menyertakan penulisan objek
11. ***ObjectStreamConstants*** : Konstanta dituliskan ke dalam Object Serialization Stream
12. ***Serializable*** : Serialisasi dari sebuah kelas diaktfkan oleh kelas yang mengimplementasikan interface Serializable java.io

* Class Summary

1. ***BufferedInputStream*** : Menambahkan fungsionalitas ke aliran input lain -yaitu, kemampuan untuk menyangga input dan mendukung metode penandaan dan reset.
2. ***BufferedOutputStream*** : Mengimplementasikan sebuah aliran output buffer.
3. ***BufferedReader*** : Membaca teks dari aliran output karakter, menyangga karakter untuk menyediakan pembacaan karakter, array, dan lines yang efisien.
4. ***BufferedWriter*** : Menulis teks ke aliran output karakter, menyangga karakter untuk menyediakan penulisan karakter, array, dan lines yang efisien.
5. ***CharArrayReader*** : Mengimplementasikan sebuah aliran output dimana datanya dapat ditulis ke dalam byte array.
6. ***CharArrayWriter*** : Mengimplementasikan penyanggaan karakter yang dapat digunakan sebagai Writer.
7. ***Console*** : Metode untuk mengakses console berbasis karakter, jika bisa, diasosiasikan dengan mesin virtual Java saat ini.
8. ***File*** : Sebuah representasi abstrak dari pathname file dan direktori.
9. ***Writer*** : Kelas abstrak untuk menulis kedalam aliran karakter.
10. ***Reader*** : Kelas abstrak untuk membaca aliran karakter.

* Exception Summary

1. ***CharConversionException*** : Kelas dasar untuk pengecualian konversi karakter.
2. ***EOFException*** : Memberikan signal jika akhir file atau aliran telah dicapai secara tak terduga ketika input.
3. ***FileNotFoundException*** : Memberikan signal bahwa sebuah percobaan untuk membuka file yang telah ditentukan pathname-nya gagal.
4. ***IOException*** : Menandakan bahwa pengecualian I/O dari beberapa jenis telah terjadi.
5. ***NotActiveException*** : Muncul ketika serialisasi atau deserialisasi tidak aktif.
6. ***NotSerializableException*** : Muncul ketika sebuah contoh memerlukan interface serialisasi.
7. ***UnsupportedEncodingException*** : Enkoding karakter tidak didukung.

* Error Summary

1. **IOError** : Muncul ketika ada error I/O yang serius telah terjadi.
2. Format penulisan bahasa pemrograman Java:
3. Harus disimpan berekstensi \*.java
4. Nama file harus sama dengan nama class public, misalnya nama filenya myHello.java, maka nama class public juga harus myHello

Contoh:

public class myHello {

//harus sama dengan nama file (myHello.java)

}

1. Comment sebaiknya sebaiknya ditulis untuk menjelaskan sebuah class atau method. Comment ditulis untuk tujuan dokumentasi.

Contoh:

public static void main(String[]args){

//untuk menampilkan kata myfirs java

System.out.println(“Hello this is my first java”);

}

1. Java statement adalah suatu baris yang diakhiri dengan titik koma.

Contoh:

System.out.println(“Hello this is my first java”);

1. Block adalah satu atau beberapa statement yang berada diantara tanda kurung kurawal { dan diakhiri dengan kurung kurawal }.

Contoh:

public static void main(String[]args){

System.out.println(“Hello this is my first java”);

System.out.println(“Java is very good”);

}

1. Deklarasi package merupakan sebuah folder yang berisi sekumpulan program Java. Deklarasi package biasanya dilakukan saat membuat program atau aplikasi besar.

Contoh deklarasi package:

package com.petanikode.program;

1. Bagian import Pada bagian ini, kita melakukan impor library yang dibutuhkan pada program. Library merupakan sekumpulan class dan fungsi yang bisa kita gunakan dalam membuat program. Contoh impor library:

import java.util.Scanner;

1. Bagian class Java merupakan bahasa pemrograman yang menggunakan paradigma OOP (Object Oriented Programming). Setiap program harus dibungkus di dalam class agar nanti bisa dibuat menjadi objek. Kalau kamu belum paham apa itu OOP? Cukup pahami class sebagai deklarasi nama program.

class NamaProgram {

public static void main(String args[]){

System.out.println("Hello World");

}

}

Ini adalah blok class. Blok class dibuka dengan tanda kurung kurawal { kemudian ditutup atau diakhiri dengan }. Di dalam blok class, kita dapat mengisinya dengan method atau fungsi-fungsi dan juga variabel. Pada contoh di atas, terdapat method main().

1. Method main Method main() atau fungsi main() merupakan blok program yang akan dieksekusi pertama kali. Ini adalah entri point dari program.Method main() wajib kita buat. Kalau tidak, maka programnya tidak akan bisa dieksekusi. Contoh method main().

public static void main(String args[]){

System.out.println("Hello World");

}

Penulisannya harus seperti ini… Method main() memiliki parameter args[]. Parameter ini nanti akan menyimpan sebuah nilai dari argumen di command line.

Lalu di dalam method main(), terdapat statement atau fungsi:

System.out.println("Hello World");

Ini adalah fungsi untuk menampilkan teks ke layar monitor.

1. Statement dan Ekspresi pada Java

Statement dan eksrepsi adalah bagian terkecil dalam program. Setiap statement dan ekspresi di Java, harus diakhiri dengan titik koma (;).

Contoh statemen dan ekspresi:

System.out.println("Hello World");

System.out.println("Apa kabar?");

var x = 3;

var y = 8;

var z = x + y;

Statemen dan ekspresi akan menjadi instruksi yang akan dikerjakan oleh komputer.

Pada contoh di atas, kita menyuruh komputer untuk menampilkan teks "Hello World", dan "Apa kabar?". Lalu kita menyuruhnya untuk menghitung nilai x + y.

1. Blok Program Java

Blok program merupakan kumpulan dari statement dan ekspresi yang dibungkus menjadi satu.

Blok program selalu dibuka dengan kurung kurawal { dan ditutup dengan }.

Contoh blok program:

// blok program main

public static void main(String args[]){

System.out.println("Hello World");

System.out.println("Hello Kode");

// blok program if

if( true ){

System.out.println('True');

}

// blok program for

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

System.out.println("Perulangan ke"+i);

}

}

Intinya: jika kamu menemukan kurung { dan }, maka itu adalah sebauh blok program. Blok program dapat juga berisi blok program yang lain (nested). Pada contoh di atas, blok program main() berisi blok if dan for.

1. Penulisan String dan Karakter

String merupakan kumpulan dari karakter. Kita sering mengenalnya dengan teks.

Contoh string: "Hello world"

Aturan penulisan string pada Java, harus diapit dengan tanda petik ganda seperti pada contoh di atas.

Apabila diapit dengan tanda petik tunggal, maka akan menjadi sebuah karakter.

Contoh: 'Hello world'.

Jadi harap dibedakan:

Tanda petik ganda ("...") untuk membuat string;

Sedangkan tanda petik tunggal ('...') untuk membuat karakter.

1. Case Sensitive

Java bersifat Case Sensitive, artinya huruf besar atau kapital dan huruf kecil dibedakan.

Contoh:

String nama = "Petani Kode";

String Nama = "petanikode";

String NAMA = "Petanikode.com";

System.out.println(nama);

System.out.println(Nama);

System.out.println(NAMA);

Tiga variabel tersebut merupakan tiga variabel yang bebeda, mesikipun sama-sama bernam nama. Banyak pemula yang sering salah pada hal ini. Karena tidak bisa membedakan mana variabel yang menggunakan huruf besar dan mana yang menggunakan huruf kecil. Apabila kita membuat variabel seperti ini:

String jenisKelamin = "Laki-laki";

Maka kita harus memanggilnya seperti ini:

System.out.println(jenisKelamin);

Bukan seperti ini:

System.out.println(jeniskelamin);

Perhatikan, huruf K adalah huruf kapital.

1. Gaya Penulisan Case

Gaya penulisan case (case style) yang digunakan oleh Java adalah: camelCase, PascalCase, dan ALL UPPER.

Gaya penulisan camelCase digunakan pada nama variabel, nama objek, dan nama method.

Contoh:

String namaSaya = "Dian";

Lalu untuk PascalCase digunakan pada penulisan nama class.

Contoh:

class HelloWOrld {

//...

}

Perhatikan nama class-nya, kita menggunakan huruf kapital di awal, dan huruf kapital pada huruf W untuk memisahkan dua suku kata.

Sedangkan camelCase huruf depannya menggunakan huruf kecil, dan awalan suku kata berikutnya menggunakan huruf besar.

// ini camelCase

belajarJava

// ini PascalCase

BelajarJava

Lalu, penulisan ALL UPPER atau semaunya kapital digunakan pada pembuatan nama konstanta.

Contoh:

public final String DB\_NAME = "petanikode";

Untuk penulisan dua suku kata atau lebih, ALL UPPER dipisah dengan garus bawah atau underscore (\_).

**Sumber Referensi :**

<https://stackify.com/top-java-software-errors/>

<https://bahasajava.com/20-syntax-error-pada-pemrograman-java-pemula/>

<http://5bab5.blogspot.com/2017/09/pengertian-dan-fungsi-import-pada-java.html>

<https://www.petanikode.com/java-sintaks/>

<https://amicowo.wordpress.com/2011/07/18/cara-penulisan-program-java/>

<https://shomad22.wordpress.com/2016/01/09/input-pada-java/>