Pemrograman Berorientasi Objek
Tugas 2

Alvin Pratama Thejakusuma
51019002

PRODI SISTEM INFORMASI
STMIK KHARISMA MAKASSAR

Soal

1. 50 pesan error di java (error tentang apa berikan penjelasannya.
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:
    1) "… expected" = pesan error ini akan muncul apabila ada sesuatu yang kurang pada kode. Yang paling sering adalah lupa dalam menempatkan (;) dan menutup kurung.
    2) "unclosed string literal" = biasanya muncul apabila kita lupa menutup string (").
    3) "illegal start of an expression" = biasanya muncul dikarenakan kode yang buruk.
    4) "cannot find symbol" = biasanya karena perbedaan penulisan pada pendeklarasian variabel dengan penulisan kode.
    5) "public class XXX should be in file" = pesan ini muncul apabila penulisan class berbeda dengan nama file program java. Pastikan kapital nya juga sama.
    6) "incompatible types" = muncul apabila terdapat perbedaan tipe. Biasanya terjadi apabila kode berusaha untuk merubah text string ke integer.
    7) "invalid method declaration; return type required" = muncul apabila kita lupa untuk menuliskan tipe.
    8) "method <X> in class <Y> cannot be applied to given types" = error ini menjelaskan apabila kode kita memanggil parameter yang salah.
    9) "missing return statement" = akan muncul apabila metode tidak mengembalikan nilai apapun.
    10) "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.
    11) "reached end of file while parsing" = biasanya muncul apabila kode program tidak ditutup dengan (})
    12) "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.
    13) "variable <X> might not have been initialized" = Ini terjadi ketika variabel lokal yang dideklarasikan dalam suatu metode belum diinisialisasi.
    14) "Operator .. cannot be applied to <X>" = Masalah ini terjadi saat operator digunakan untuk jenis, bukan dalam definisinya. Ini sering terjadi ketika kode Java mencoba menggunakan string tipe dalam perhitungan. Untuk memperbaikinya, string perlu dikonversi ke integer atau float.
    15) "inconvertible types" = terjadi ketika kode Java mencoba melakukan konversi ilegal. Misalnya, boolean tidak dapat dikonversi ke bilangan bulat.

16) “missing return value” = akan muncul apabila saat pernyataan pengembalian menyertakan jenis yang salah.
17) “cannot return a value from method whose result type is void” = terjadi saat metode void mencoba mengembalikan nilai apa pun, ini bisa diperbaiki dengan mengubah metode agar sesuai dengan tipe dalam pernyataan pengembalian.
18) “non-static variable . . . cannot be referenced from a static context” = Kesalahan ini terjadi ketika kompiler mencoba mengakses variabel non-statis dari metode statis.
19) “non-static method . . . cannot be referenced from a static context” = Masalah ini akan muncul ketika kode Java mencoba memanggil metode non-statis di kelas non-statis.
20) “(array) <X> not initialized” = Anda akan mendapatkan pesan “(array) <X> not initialized” ketika sebuah array telah dideklarasikan tetapi tidak diinisialisasi. Array tetap panjangnya sehingga setiap array perlu diinisialisasi dengan panjang yang diinginkan.
21) “ArrayIndexOutOfBoundsException” = 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.
22) “StringIndexOutOfBoundsException” = 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.
23) “NullPointerException” = terjadi ketika program mencoba menggunakan referensi objek yang tidak memiliki nilai yang ditetapkan padanya
24) “NoClassDefFoundError” = terjadi ketika penerjemah tidak dapat menemukan file yang berisi kelas dengan metode utama. Biasanya dikarenakan file tidak berada pada direktori yang tepat / nama file berbeda dengan nama kelas
25) “NoSuchMethodFoundError” = Pesan kesalahan ini akan muncul ketika perangkat lunak Java mencoba memanggil metode dari suatu kelas dan metode tersebut tidak lagi memiliki definisi.
26) “NoSuchProviderException” = error ini akan muncul ketika penyedia keamanan diminta yang tidak tersedia.
27) "AccessControlException" = menunjukkan bahwa akses yang diminta ke sumber daya sistem seperti sistem file atau jaringan ditolak
28) “ArrayStoreException” = muncul ketika aturan casting elemen dalam array Java dilanggar. Berhati-hatilah dengan nilai apa yang Anda tempatkan di dalam array.
29) “bad magic number” = Pesan kesalahan perangkat lunak Java ini berarti ada yang salah dengan file definisi kelas di jaringan.

30) “broken pipe” = Pesan kesalahan ini mengacu pada aliran data dari file atau soket jaringan yang telah berhenti bekerja atau ditutup dari ujung lainnya
31) “could not create Java Virtual Machine” = Pesan kesalahan Java ini biasanya terjadi ketika kode mencoba memanggil Java dengan argumen yang salah. Ini sering disebabkan oleh kesalahan dalam deklarasi dalam kode atau mengalokasikan jumlah memori yang tepat untuk itu.
32) “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
33) “ClassCastException” = Pesan "ClassCastException" menunjukkan kode Java mencoba untuk melemparkan objek ke kelas yang salah.
34) “ClassFormatError” = Pesan "ClassFormatError" menunjukkan kesalahan tautan dan terjadi ketika file kelas tidak dapat dibaca atau ditafsirkan sebagai file kelas.
35) “ClassNotFoundException” = terjadi saat run time — artinya kelas yang ada di sana selama kompilasi hilang saat run time. Ini adalah kesalahan tautan.
36) “ExceptionInInitializerError” = Masalah Java ini akan terjadi ketika ada yang tidak beres dengan inisialisasi statis. Ketika kode Java nanti menggunakan kelas, kesalahan "NoClassDefFoundError" akan terjadi.
37) “IllegalBlockSizeException” = "IllegalBlockSizeException" akan terjadi selama dekripsi ketika panjang pesan bukan kelipatan 8 byte.
38) “BadPaddingException” = "BadPaddingException" akan terjadi selama dekripsi saat padding digunakan untuk membuat pesan yang dapat diukur dengan kelipatan 8 byte.
39) “IncompatibleClassChangeError” = "IncompatibleClassChangeError" adalah bentuk LinkageError yang dapat terjadi ketika kelas dasar berubah setelah kompilasi kelas anak.
40) “FileNotFoundException” = Pesan kesalahan perangkat lunak Java ini ditampilkan ketika file dengan nama path yang ditentukan tidak ada. Selain file yang tidak menunjukkan nama path yang ditentukan, ini bisa berarti file yang ada tidak dapat diakses.
41) “EOFException” = Sebuah "EOFException" dilemparkan ketika akhir file atau akhir aliran telah dicapai secara tidak terduga selama input.
42) “UnsupportedEncodingException” = Pesan kesalahan perangkat lunak Java ini ditampilkan ketika pengkodean karakter tidak didukung.
43) “SocketException” = Pesan "SocketException" menunjukkan ada kesalahan saat membuat atau mengakses soket
44) “SSLException” = Pesan kesalahan perangkat lunak Java ini terjadi ketika ada kegagalan dalam operasi terkait SSL.

45) “MissingResourceException” = "MissingResourceException" terjadi ketika sumber daya tidak ada. Jika sumber daya berada di classpath yang benar, ini biasanya karena file properti tidak dikonfigurasi dengan benar.
46) “NoInitialContextException” = Kesalahan "NoInitialContextException" terjadi ketika aplikasi Java ingin melakukan operasi penamaan tetapi tidak dapat membuat koneksi
47) “NoSuchElementException” = Kesalahan "NoSuchElementException" terjadi ketika iterasi (seperti loop "untuk") mencoba mengakses elemen berikutnya ketika tidak ada.
48) “NoSuchFieldError” = Pesan kesalahan perangkat lunak Java ini dilempar ketika aplikasi mencoba mengakses bidang dalam suatu objek tetapi bidang yang ditentukan tidak lagi ada di objek
49) “NumberFormatException” = Pesan kesalahan perangkat lunak Java ini terjadi saat aplikasi mencoba mengonversi string ke tipe numerik, tetapi nomor tersebut bukan string digit yang valid
50) “TimeoutException” = Pesan kesalahan perangkat lunak Java ini terjadi saat operasi pemblokiran habis.

2. Java file header:

- 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

13. BufferedInputStream = Menambahkan fungsionalitas ke aliran input lain -yaitu, kemampuan untuk menyangga input dan mendukung metode penandaan dan reset.
14. BufferedOutputStream = Mengimplementasikan sebuah aliran output buffer
15. BufferedReader = Membaca teks dari aliran output karakter, menyangga karakter untuk menyediakan pembacaan karakter, array, dan lines yang efisien
16. BufferedWriter = Menulis teks ke aliran output karakter, menyangga karakter untuk menyediakan penulisan karakter, array, dan lines yang efisien
17. CharArrayReader = Mengimplementasikan sebuah aliran output dimana datanya dapat ditulis ke dalam byte array
18. CharArrayWriter = Mengimplementasikan penyanggaan karakter yang dapat digunakan sebagai Writer
19. Console = Metode untuk mengakses console berbasis karakter, jika bisa, diasosiasikan dengan mesin virtual Java saat ini
20. File = Sebuah representasi abstrak dari pathname file dan direktori
21. Writer = Kelas abstrak untuk menulis kedalam aliran karakter
22. Reader = Kelas abstrak untuk membaca aliran karakter

- Exception Summary

23. CharConversionException = Kelas dasar untuk pengecualian konversi karakter
24. EOFException = Memberikan signal jika akhir file atau aliran telah dicapai secara tak terduga ketika input
25. FileNotFoundException = Memberikan signal bahwa sebuah percobaan untuk membuka file yang telah ditentukan pathname-nya gagal
26. IOException = Menandakan bahwa pengecualian I/O dari beberapa jenis telah terjadi
27. NotActiveException = Muncul ketika serialisasi atau deserialisasi tidak aktif
28. NotSerializableException = Muncul ketika sebuah contoh memerlukan interface serialisasi
29. UnsupportedEncodingException = Enkoding karakter tidak didukung

Error Summary

30. IOError = Muncul ketika ada error I/O yang serius telah terjadi

3. Format penulisan Bahasa pemrograman java
    1) Harus disimpan berekstensi *.java

    2) 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)
       }

    3) 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");
       }

    4) Java statement adalah suatu baris yang diakhiri dengan titik koma.
       Contoh:
       System.out.println("Hello this is my first java");

    5) 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");
       }

    6) 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;

7) 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;

8) 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().

9) 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.

10) 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.

11) 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.

12) 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.

13) 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.

14) 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 (_).