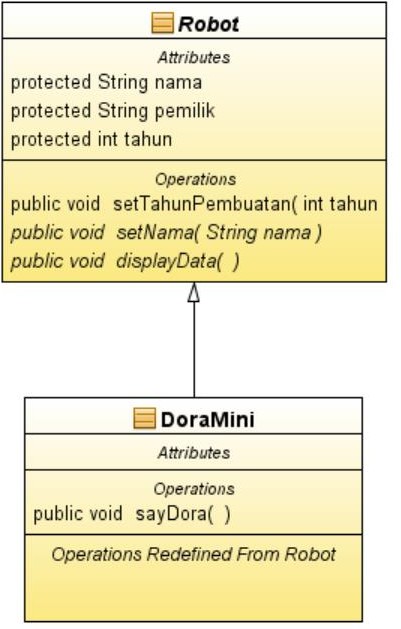
Nama :Amalia Fandaning Tyas

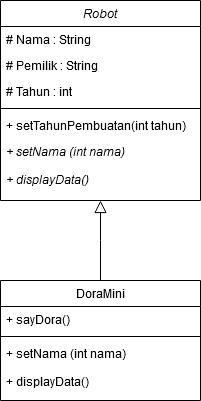
NIM : 19051397012

Prodi : D4 MI 19 A

Praktek PBO Abstract dan Interface

Tugas:

1. Ubahlah class diagram di atas ke dalam bentuk standar UML class diagram Jawab :



1. Buatlah 2 kelas berdasarkan diagram kelas di atas (beserta kelas Main). Keterangan tambahan:
2. Kelas Robot adalah kelas abstrak
3. Method “sayDora” akan menampilkan “Halo, Saya Dora Mini” di layar
4. Method displayData akan menampilkan setiap nilai dari atribut yang dimiliki ke layar
5. Method setNama dan displayData adalah method abstract Jawab :

abstract class Robot

{

String nama; String pemilik; int tahun;

void setTahunpembuatan(int *tahun*)

{

System.out.println("Saya dibuat tahun " + *tahun*);

}

abstract void setNama(String *nama*); abstract void displayData();

}

class DoraMini extends Robot

{

void sayDora()

{

System.out.println("Hi, saya Dora Mini");

}

@Override

void setNama(String *nama*)

{

System.out.println("Nama panggilan saya " + *nama*);

}

@Override

void displayData()

{

sayDora(); setNama("Dora"); setTahunpembuatan(2020);

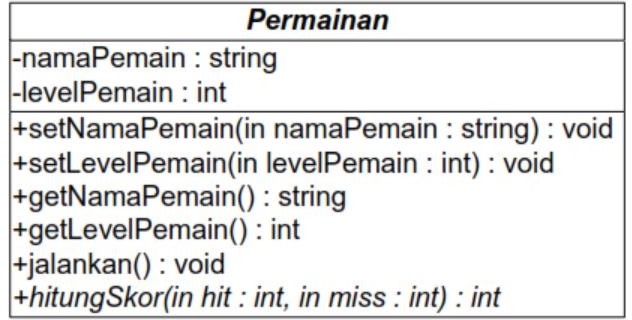
}

public static void main(String[] *args*) { DoraMini DoraMini = new DoraMini(); DoraMini.displayData();

}

}

1. Seorang analyst membuat aplikasi simulasi permainan. Analyst membuat class diagram dengan abstract class sebagai berikut:



Deskripsi:

1. Atribut namaPemain dan levelPemain menyimpan nama dan level pemain.
2. Nilai levelPemain berkisar 1-100 dengan ketentuan: a. 1-20: normal b. 21-80: medium c. 81- 100: hard
3. Method jalankan() akan menjalankan skenario permainan (set nama dan level pemain, mengeluarkan data tersebut, dan menghitung skor pemain).
4. Method hitungSkor merupakan abstract method.

Tugas :

1. Buatlah kode berdasarkan abstract class di atas.
2. Buatlah classes lain yang menggunakan abstract class di atas dengan deskripsi:
   1. “PermainanArcade” dengan aturan hitung skor: jumlah hit x 3 – jumlah miss x 1;
   2. “PermainanStrategy” dengan aturan hitung skor: jumlah hit x 5; Perhatikan bahwa algoritma hitungSkor ditentukan oleh subclass, bukan superclass.
3. Buktikan bahwa abstract method memastikan bahwa method tersebut di-override oleh subclass.

c) Buktikan bahwa objek dari abstract class tidak dapat dibentuk.

Jawab : Tidak dapat dibentuk karna memiliki 2 jenis implementasi class yaitu Permainan Arcade dan Permainan Strategi. Dapat dibentuk, jika dibedakan antar class.

import java.util.\*;

abstract class Permainan

{

Scanner inputUser = new Scanner(System.in); String namaPemain;

int levelPemain;

void setNamaPemain(String *namaPemain*)

{

System.out.print("Masukkan nama anda : ");

*namaPemain* = inputUser.nextLine();

}

void setLevelPemain(int *levelPemain*)

{

System.out.print("Pilih level anda : ");

*levelPemain* = inputUser.nextInt();

if (*levelPemain* >= 1 && *levelPemain* <=20 )

{

System.out.println("Level normal");

}

else if (*levelPemain* >= 21 && *levelPemain* <= 80)

{

System.out.println("Level medium");

}

else if (*levelPemain* >= 81 && *levelPemain* <= 100)

{

System.out.println("Level hard");

}

else

{

System.out.println("Level tidak tersedia!"); System.out.println("Level terendah : 1"); System.out.println("Level tertinggi : 100");

}

}

String getNamaPemain()

{

return namaPemain;

}

int getLevelPemain()

{

return levelPemain;

}

void jalankan()

{

setNamaPemain(namaPemain); setLevelPemain(levelPemain);

}

abstract int hitungSkor(int *hit*, int *miss*);

}

class PermainanArcade extends Permainan

{

@Override

int hitungSkor(int *hit*, int *miss*)

{

int total = (*hit*\*3) - (*miss*\*1); System.out.println("Maka, skor perolehan : " + total); System.out.println("\n");

return total;

}

}

class PermainanStrategy extends Permainan

{

@Override

int hitungSkor(int *hit*, int *miss*)

{

int total = *hit*\*5;

System.out.println("Maka, skor perolehan : " + total); System.out.println("\n");

return total;

}

}

public class Main\_Permainan {

public static void main(String[] *args*) { Scanner inputUser = new Scanner(System.in); Permainan Arcade = new PermainanArcade();

Permainan Strategy = new PermainanStrategy();

System.out.println("--MODE PERMAINAN--"); System.out.println("1. Permainan Arcade"); System.out.println("2. Permainan Strategy"); System.out.print("Masukkan pilihan anda : "); int pilmode = inputUser.nextInt();

if (pilmode == 1) { Arcade.jalankan();

System.out.println("\n--PERMAINAN ARCADE--"); Arcade.hitungSkor(5, 3);

}

else if (pilmode == 2)

{

Strategy.jalankan();

System.out.println("\n--PERMAINAN STRATEGY--");

Strategy.hitungSkor(10, 5);

}

else

{

System.out.println("\nMode tidak tersedia!");

}

}

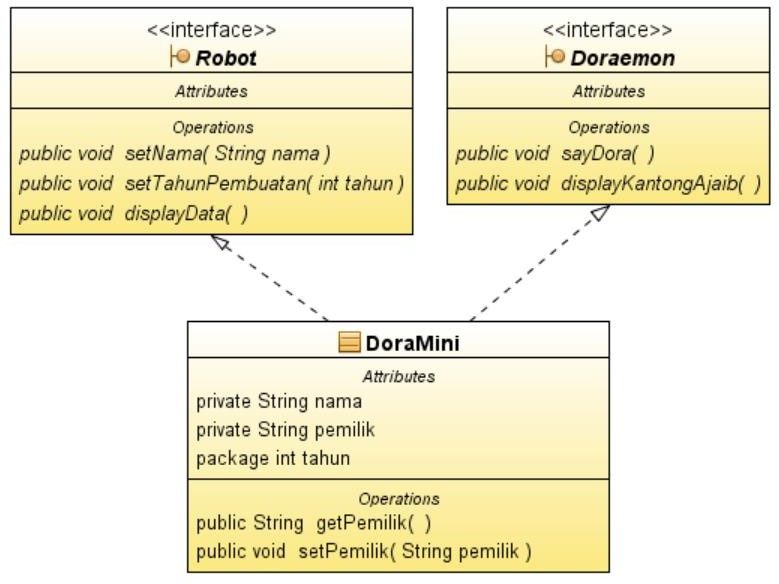
}

1. Tulislah dan jelaskan perbedaan antara abstract class dan interface Jawab :

Interface untuk mendefinisikan method standar yang sama dalam class-class yang berbeda. Sekali kita telah membuat kumpulan definisi method standar(abstrak), kita dapat menulis method tunggal(spesifik) untuk memanipulasi semua class-class yang mengimplementasikan interface tsb.

Abstract class untuk mendefinisikan secara luas sifat-sifat dari class tertinggi pada hirarki OOP, dan gunakan subclassnya (turunannya/child class) untuk melengkapi deskripsi method dari class abstract.

4.



Keterangan:

1. Method “sayDora” akan menampilkan “Halo, Saya Dora Mini” di layar
2. Method “dispKantungAjaib” akan menampilkan “Saya juga seperti Doraemon yang memiliki kantung ajaib”
3. Method displayData akan menampilkan setiap nilai dari atribut yang dimiliki ke layar

Tugas : Buatlah 3 kelas berdasarkan diagram kelas di atas (dan juga class Main untuk membentuk objeknya).

interface Robot

{

void setNama(String *nama*); void setTahun(int *tahun*); void displayData();

}

interface Doraemon

{

void sayDora();

void displayKantongAjaib();

}

class DoraMini implements Robot , Doraemon

{

String pemilik; String nama; int tahun;

String getPemilik()

{

return pemilik;

}

void setPemilik(String *pemilik*)

{

System.out.println("Pemilik Dora Mini : " + *pemilik*);

}

@Override

public void setNama(String *nama*)

{

System.out.println("Nama : " + *nama*);

}

@Override

public void setTahun(int *tahun*)

{

System.out.println("Dibuat tahun : " + *tahun*);

}

@Override

public void displayData()

{

setNama("DoraMini"); setTahun(2017); setPemilik("Nur Wulan C.\n");

System.out.println("--DoraMini Berkata--"); sayDora();

displayKantongAjaib();

}

@Override

public void sayDora()

{

System.out.println("Halo, Saya Dora Mini");

}

@Override

public void displayKantongAjaib()

{

System.out.println("Saya juga seperti Doraemon yang memiliki kantung ajaib\n");

}

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

{

DoraMini DoraMini = new DoraMini(); DoraMini.displayData();

}

}