public interface IRequest{

public void execute();

}

public class HelloRequest implements IRequest{

private String name;

public HelloRequest(String name){

this.name = name;

}

public void execute(){

System.out.printf("Hello! %s!%n", name);

}

}

public class WelcomeRequest implements IRequest{

private String place;

public WelcomeRequest(String place){

this.place = place;

}

public void execute(){

System.out.printf("Welcome to %s!%n", place);

}

}

public class Test{

public static void main(String[] args){

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

int n = (int)(Math.random()\* 10)% 2;

switch (n){

case 0:

doRequest(

new HelloRequest("caterpillar"));

break;

case 1:

doRequest(new WelcomeRequest("PmWiki"));

}

}

}

public static void doRequest(IRequest request){

request.execute();

}

}

class Abstract01{//class無abstract修飾子

int a = 0, b = 10;

abstract void absMethod01(int x);//抽象方法

}

abstract class Abstract02{

int a = 0, b = 10;

abstract void absMethod02(int x){//抽象方法含處理內容

a = x;

if (a == b) System.out.println("absMethod02：a等於b");

else System.out.println("absMethod02：a不等於b");

}

}

abstract class Abstract03{

int a = 0, b = 10;

abstract void absMethod03(int x);

}

abstract class Abstract04{

int a = 0, b = 10;

abstract void absMethod04a(int x);

abstract void absMethod04b();

}

abstract class Abstract05{

int a = 0, b = 10;

abstract void absMethod05a(int x);

void absMethod05b(){

System.out.println("absMethod05b：非abstract方法");

}

}

abstract class Abstract06{

int a = 0, b = 10;

void absMethod06(){

System.out.println("absMethod06：非abstract方法");

}

}

abstract class Abstract07{

int a = 0, b = 10;

abstract void absMethod07(int x);

}

class AbsMain07{

public static void main(String[] args){

Abstract07 A07 = new Abstract07();

}

}

abstract class Abstract08{

int a = 0, b = 10;

abstract void absMethod08(int x);

}

class AbsSon08 extends Abstract08{};

class AbsMain08{

public static void main(String[] args){

AbsSon08 A08 = new AbsSon08();

}

}

abstract class Abstract09{

int a = 0, b = 10;

abstract void absMethod09(int x);

}

class AbsSon09 extends Abstract09{

void absMethod09(int x){

a = x;

if (a == b)

System.out.println("absMethod09：a等於b");

else

System.out.println("absMethod09：a不等於b");

}

}

class AbsMain09{

public static void main(String[] args){

AbsSon09 A09 = new AbsSon09();

A09. absMethod09(10);

A09. absMethod09(9);

}

}

abstract class Abstract10{

int a = 0, b = 10;

void absMethod10(){

System.out.println("absMethod10：非abstract方法");

}

}

class AbsSon10 extends Abstract10{ }

class AbsMain10{

public static void main(String[] args){

AbsSon10 A10 = new AbsSon10();

A10.absMethod10();

}

}

interface Iffat{//定義介面

int x = 100;

void ifmd ();

}

public class Ifmain{

public static void main(String args[]){

Iffat aa = new Iffat();//建立物件，錯誤

System.out.println("x=" + aa.x);

}

}

interface Iffat1{//定義介面

int x = 100;//宣告欄位，具有final屬性

void ifmd1();//宣告無內容之方法，具有public屬性

}

class Ifson1 implements Iffat1{//實作介面

public void ifmd1(){//覆蓋方法並置入內容，並須指定public

System.out.println("這是實作單一介面的例子");

}

}

public class Ifmain1{

public static void main(String args[]){

Ifson1 aa = new Ifson1();

System.out.println("x=" + aa.x);

aa.ifmd1();

}

}

interface Iffat2a{//定義介面

int x = 100;//宣告欄位

void ifmd2a();//宣告無內容之方法

}

interface Iffat2b{//定義介面

int y = 200;//宣告欄位

void ifmd2b();//宣告無內容之方法

}

class Ifson2 implements Iffat2a, Iffat2b{//實作多個介面

public void ifmd2a(){//覆蓋方法並置入內容

System.out.println("這是實作複數介面的例子Iffat2a");

}

public void ifmd2b(){//覆蓋方法並置入內容

System.out.println("這是實作複數介面的例子Iffat2b");

}

}

public class Ifmain2{

public static void main(String args[]){

Ifson2a aa = new Ifson2a();

System.out.println("x=" + aa.x);

System.out.println("y=" + aa.y);

aa.ifmd2a();

aa.ifmd2b();

}

}

interface Iffat3{//定義介面

int x = 100;//宣告欄位

void ifmd3();//宣告無內容之方法

}

class Clfat3{//定義類別

int z = 300;//宣告欄位

public void clmd3(){

System.out.println("實作介面及繼承類別的例子Clfat3");

}//類別可宣告有內容之方法

}

class Ifson3 extends Clfat3 implements Iffat3{//實作介面並繼承類別

public void ifmd3(){//覆蓋方法並置入內容

System.out.println("實作介面及繼承類別的例子Iffat3");

}

}

public class Ifmain3{

public static void main(String args[]){

Ifson3 aa = new Ifson3();

//aa.x = aa.x + 100;介面定義之欄位具final屬性，不可改變其值

System.out.println("x=" + aa.x);

aa.z = aa.z + 200;//類別定義之欄位可改變其值

System.out.println("z=" + aa.z);

aa.ifmd3();

aa.clmd3();

}

}

interface Iffat4{

int x = 100;

void ifmd4(){

System.out.println("介面方法具abstract屬性，不能有執行碼");

}

}

interface Iffat5{

int x = 500;

void ifmd5();//方法具有public屬性，不須宣告

}

class Ifson5 implements Iffat5{//實作介面

void ifmd5(){//覆蓋方法並置入內容，但無指定public

System.out.println("沒指定public之錯誤範例");

}

}

interface Iffat6{

int x = 600;//宣告欄位，不須明示即具有final屬性

void ifmd6();

}

class Ifson6 implements Iffat6{

public void ifmd1(){

x = x+400;//不可變更final欄位值

System.out.println("變更final欄位值，x = " + x);

}

}