# Problem 1:

In package "Problem 1"

Create interface "Face1" with method "method1" that accepts no argument and return nothing.

Create class "Implementor1" that implements interface "Face1" and has public field "data" with type int. Make the implemented method print out "Implementor1 method1"

Create following Main class.

|  |
| --- |
| public class Main {   public static void main(String[] args) {  Implementor1 im = new Implementor1();  im.data = 0;  im.method1();   Face1 f1 = im;  f1.method1();  } } |

What are the outputs? :

Implementor1 method1

Implementor1 method1

Can we add the following code to main after "f1.method1();" ? Why?

f1.data = 0;

>> ไม่ เพราะ Face1 ไม่มีการเชื่อมtype data กับclass Implementor1

//Problem 1

public interface Face1 {  
 void method1();  
}

public class Implementor1 implements Face1 {  
 int data = 0;  
  
 public void method1() {  
 System.*out*.println("Implementor1 method1");  
 }  
}

public class Main {  
  
 public static void main(String[] args) {  
 Implementor1 im = new Implementor1();  
 im.data = 0;  
 im.method1();  
  
 Face1 f1 = im;  
 f1.method1();  
 }  
}

# Problem 2:

In package "Problem 2"

Create interface "Face2" with 2 methods

1. "method1" that accepts no argument and return nothing.
2. ***default*** method, "method2", that just print "Face2 method2" .

Create class "Implementor2" that implements interface "Face2", but only implements "method1" to print " System.out.println("Implementor2 Method1");".

Create class "Implementor2\_2" that implements interface "Face2". It implements "method1" to print "System.out.println("Implementor2\_2 Method1");" and implements "method2" to print

"System.out.println("Implementor2\_2 Method2");

Create following Main class.

|  |
| --- |
| public class Main2 {   public static void main(String[] args) {  Implementor2 im = new Implementor2();  im.method1();  im.method2();  Face2 f = new Implementor2\_2();  f.method1();  f.method2();  } } |

What are the outputs?

Implementor2 Method 1

Face2 method2 //มาจาก default method2

Implementor2\_2 Method1

Implementor2\_2 Method2

//Problem2

public interface Face2 {  
 void method1();  
  
 default void method2(){  
 System.*out*.println("Face2 method2");  
 }  
}

public class Implementor2 implements Face2{  
 public void method1(){  
 System.*out*.println("Implementor2 Method 1");  
 }  
}

public class Implementor2\_2 implements Face2{  
 public void method1(){  
 System.*out*.println("Implementor2\_2 Method1");  
 }  
 public void method2(){  
 System.*out*.println("Implementor2\_2 Method2");  
 }  
}

public class Main2 {  
 public static void main(String[] args) {  
 Implementor2 im = new Implementor2();  
 im.method1();  
 im.method2();  
  
 Face2 f = new Implementor2\_2();  
 f.method1();  
 f.method2();  
 }  
}

# Problem 3

In package "Problem 3"

Create interface "Face3\_1" with method "method1" that accepts no argument and return nothing.

Create interface "Face3\_2" with method "method2" that accepts no argument and return nothing.

Create class "Implementor3" that implements interface "Face3\_1" and "Face3\_2". The method "method1" prints out "Implementor3 method1", and the method "method2" prints out "Implementor3 Method2"

Create following Main class.

|  |
| --- |
| public class Main3 {   public static void main(String[] args) {  Implementor3 im = new Implementor3();  im.method1();  im.method2();  Face3\_1 f1 = im;  f1.method1();  Face3\_2 f2 = im;  f2.method2();  ((Implementor3) f1).method2();  ((Face3\_1) f2).method1();  } } |

What are the outputs?

Implementor3 method1

Implementor3 Method2

Implementor3 method1

Implementor3 Method2

Implementor3 Method2

Implementor3 method1

//Problem3

public interface Face3\_1 {  
 void method1();  
}

public interface Face3\_2 {  
 void method2();  
}

public class Implementor3 implements Face3\_1,Face3\_2{  
 public void method1(){  
 System.*out*.println("Implementor3 method1");  
 }  
 public void method2(){  
 System.*out*.println("Implementor3 Method2");  
 }  
}

public class Main3 {  
 public static void main(String[] args) {  
 Implementor3 im = new Implementor3();  
  
 im.method1();  
  
 im.method2();  
  
 Face3\_1 f1 = im; //interface3\_1 เชื่อมกับclass implement  
 f1.method1();  
  
 Face3\_2 f2 = im; //interface3\_2 เชื่อมกับclass implement  
 f2.method2();  
  
 ((Implementor3) f1).method2(); //f1ดึงข้อมูลจากclass implement  
 ((Face3\_1) f2).method1(); //f2ดึงข้อมูลจากFace3\_1 และFace3\_1ดึงจากclass implementอีกที  
 }  
}