Université Paris Diderot **Programmation Orientée Objet**
L2-L3 **Année 2014-2015**

TD n° 10

Révisions

1 Échauffement

Exercice 1 Les programmes suivants sont-ils corrects ? Si oui, qu'affichent-ils ?

1.
```
1  public class A {
2      private static int a = 4;
3      public static void main(String[] args) { System.out.println(a); }
4  }
```

2.
```
1  class B { private static int b = 5; }

3  public class A {
4      public static void main(String[] args) {
5          System.out.println(B.b);
6      }
7  }
```

3.
```
1  class B { int i = 2; }

3  public class A {
4      public static void main(String[] args) {

6          B b1 = new B();
7          B b2 = new B();

9          System.out.println(b1.equals(b2));
10         System.out.println(b1 == b2);
11     }
12 }
```

2 Classes internes, locales

Exercice 2 Indiquez si les exemples suivants fonctionnent, c'est à dire s'ils compilent, puis s'ils s'exécutent en cas de présence d'une fonction `main`, et expliquez pourquoi.

1.

```
public class A {
    int a = 2;
    public class B {int a = 3;}
    public static void main(String[] args) {
        A wa = new A();
        B wb = new B();
        System.out.println("a = "+wa.a);
        System.out.println("a = "+wb.a);
    }
}
```

2.

```
public class A {
    public class B {public static void test(){System.out.println("Bonjour");}}
    public static void main(String[] args) {
        A a = new A();
        a.new B().test();
    }
}
```

3.

```
public class A {
    int a = 2;
    public static class B {int a = 3;}
    public static void main(String[] args) {
        A wa = new A();
        System.out.println("a = "+wa.a);
        System.out.println("a = "+B.a);
    }
}
```

4.

```
public class A {
    int a = 2;
    public class B {public void test() {a = 5;}}
    public static void main(String[] args) {
        A wa = new A();
        System.out.println("a = "+wa.a);
        wa.new B().test();
        System.out.println("a = "+wa.a);
    }
}
```

5.

```
public class A {
    int a = 2;
    private void g() { System.out.println("G"); }
    public class B { public void f() { g(); } }

    public static void main(String[] args) {
        A wa = new A();
        wa.new B().f();
    }
}
```

6.
```
1  public class A {
2      int a = 2;
3      public class B {int a = 3;
4          public void test() {A.this.a = 5;}
5      }
6      public static void main(String[] args) {
7          A wa = new A();
8          System.out.println("a = "+wa.a);
9          wa.new B().test();
10         System.out.println("a = "+wa.a);
11     }
12 }
```

7.
```
1  public class A {
2      int a = 2;
3      B b = new B();
4      public static class B { public void test() {a = 5;} }
5      public void test(){b.test();}
6      public static void main(String[] args) {
7          A wa = new A();
8          System.out.println("a = "+wa.a);
9          wa.test();
10         System.out.println("a = "+wa.a);
11     }
12 }
```

8.
```
1  public class A {
2      int a = 2;
3      public void test () {
4          class B {public void test() {a = 5;}}
5          B b = new B();
6          b.test();
7      }
8      public static void main(String[] args) {
9          A a = new A();
10         a.test();
11         System.out.println(a.a);
12     }
13 }
```

3 Héritage, surcharge, redéfinition

Exercice 3 Qu'affiche le programme suivant ?

```java
class A {
    public static String f() {
        return "A";
    }
}
class B extends A {
    public static String f() {
        return "B";
    }
}
public class exo {
    public static void myf(Object o) {
        System.out.println("Object");
    }
    public static void myf(A a) {
        System.out.println("A");
        System.out.println(a.f());
    }
    public static void myf(B b) {
        System.out.println("B");
        System.out.println(b.f());
    }
    public static void main(String[] args) {
        Object o = new A();
        Object o2 = new B();
        A a = new A();
        A a2 = new B();
        myf(o);
        myf(o2);
        myf(a);
        myf(a2);
    }
}
```

Et si les fonctions f() dans les classes A et B ne sont pas static ?

Exercice 4 Les programmes suivants sont-ils corrects ? Si oui, qu'affichent-ils ?

1.
```java
class B { private int b = 5; }

class C extends B { public int get() { return b; } }

public class A {
    public static void main(String[] args) {
        C c = new C();
        System.out.println(c.get());
    }
}
```

2.
```java
class B { final void f(){} }
class C extends B { void f(){} }
```

3.
```java
final class B {  }
class C extends B {  }
```

4.
```
class C { public void g(){} }
class D extends C {
    public static void f(){ super.g(); }
}
```

5.
```
class C { private int i = 2; }
class D extends C {
    int i = 4;
    public void f(){ super.i = 5; }
}
```

6.
```
class C { int c;  public C(int a){ c = a; } }
class D extends C { int j;  public D(){ j = 1;  super(2); }  }
```

4 Exceptions

Exercice 5 Les programmes suivant sont-ils corrects ?

1.
```
class B extends Exception {}

public class A {

    public static void main(String[] args) {
        try {
            throw new Exception();
        }
        catch (Exception e){System.out.println(e.getMessage());}
        catch (B e){System.out.println(e.getMessage());}
    }
}
```

2.
```
class B extends Exception {}

public class A {

    public static void test () throws Exception {
        try {
            throw new Exception("Table");
        }
        catch (B e){System.out.println(" B ");}
        finally { System.out.println(" F "); }
    }

    public static void main(String[] args)  {
        try {
            test();
        }
        catch (Exception e) {  System.out.println(" E "); }
    }
}
```

5 Interfaces, classes abstraites, classes anonymes

Exercice 6 Indiquez si les exemples suivants fonctionnent, c'est à dire s'ils compilent, puis s'ils s'exécutent en cas de présence d'une fonction `main`, et expliquez pourquoi.

1.
```
interface I { public int f(){ return 2; } }

public class A {
    public static void main(String[] args)  {
        System.out.println(I.f());
    }
}
```

2.
```
interface I { int i;
    public I(int a) { i = a; }
}
public class A {
    public static void main(String[] args)  {
        System.out.println(I.i);
    }
}
```

3.
```
interface I { int i = 2; }
interface J extends I {}
class B implements J {}

public class A {
    public static void main(String[] args) {
        System.out.println(B.i);
    }
}
```

4.
```
interface I {
    public void test();
    public void truc();
}

class B implements I {
    public void test(){}
}
```

5.
```
interface I {
    public void test();
}

abstract class B implements I {public void test(){}}

public class A {
    public static void main(String[] args) {
        B b = new B();
    }
}
```

6.
```
interface I {
    public int test();
    public void truc();
}

abstract class B implements I { public int test(){ return 2; } }
class C extends B {}

public class A {
    public static void main(String[] args) {
        C c = new C();
    }
}
```

7.
```
interface I {public void test();}

public class A {
    public static void main(String[] args) {
        I i = new I(){public void test(){System.out.println("Bonjour");}};
        i.test();
    }
}
```

8.
```
interface I {public void test();}
interface J {public void truc();}
abstract class B implements I,J{}

public class A {
    public static void main(String[] args) {
        I i = new B(){
                public void test(){System.out.println("Bonjour");}
                public void truc(){System.out.println("Bonsoir");}
            };
        i.test();
        i.truc();
    }
}
```

9.
```
interface I {
    int i = 2;
}

public class A {
    public static void main(String[] args) {
        System.out.println(I.i);
    }
}
```

10.
```
interface I { int i = 2; }

public class A {
    public static void main(String[] args) {
        I i = new I() {};
        System.out.println(i.i);
    }}
```

11.

```
interface I { int i = 2; }

public class A {
    public static void main(String[] args) {
        I i = new I() { int b = 2; };
        System.out.println(i.i+","+i.b);
    }
}
```

12.

```
 interface I {int i = 2;  public void f(); }

 public class A {
    public static void main(String[] args) {
        I i = new I() {
                int b = 2;
                public void f(){i = 3; b++;}
            };
        i.f();
    }
}
```

13.

```
interface I { public int f(); }

public class A {
    public static void main(String[] args) {
        I i = new I() { int b = 2; public int f(){return b;} };
        System.out.println(i.f());
    }
}
```

14.

```
interface I { int i = 2; public int f(); }
interface J extends I {}
class B implements J { public int f() {return 3;} }

public class A {
    public static void main(String[] args) {
        J j = new B();
        System.out.println(B.i+","+j.f());
    }
}
```