Informatique - Java

Cours 4

Matthieu Zimmer
Jean-Philippe Mangeot

Plan

- Test héritage
- Intégration Continue
- Github la suite
- Base de donnée

Test héritage

http://qdle.net/XXXXXXXXX

Remplacer XXXXXX par le nombre au tableau

Prise en main

#QDLE#Q#ABC*#120#

source: wikipedia, author: Muband

How many errors? A=>7 ; B=>10 ; C=>15

Constructeurs

#QDLE#Q#AB*C#90#

Affichage ?

[A]
Class C

[B]
Class A
Class B
Class C

[C]
Class C
Class B
Class A

```
class A {
    public A() {
        System.out.println("Class A");
    }
}

class B extends A {
    public B() {
        System.out.println("Class B");
    }
}

class C extends B {
    public C() {
        System.out.println("Class C");
    }
}

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

Constructeurs

#QDLE#Q#A*B#30#

Code valide ?

[A]
Oui

[B]
Non

```
class A {
    public A() {
        System.out.println("Class A");
    }
}

class B extends A {
    public B() {
        System.out.println("Class B");
    }
}

class C extends B {
    public C() {
        System.out.println("Class C");
    }
}

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

Héritage multiple

#QDLE#Q#AB*#30#

Code valide ?

[A]
Oui

[B]
Non

```
class X {
    // Class X Members
}

class Y {
    // Class Y Members
}

class Z extends X, Y {
    //Class Z Members
}
```

Héritage multiple

#QDLE#Q#A*B#30#

Code valide ?

[A]
Non

[B]
Oui

```
interface X {
    // Class X Members
}

interface Y {
    // Class Y Members
}

class Z extends X, Y {
    //Class Z Members
}
```

Héritage multiple

#QDLE#Q#A*B#30#

Code valide ?

[A]
Oui

[B]
Non

```
interface X {
    // Class X Members
}

interface Y {
    // Class Y Members
}

class Z implements X, Y {
    //Class Z Members
}
```

Héritage multiple

#QDLE#Q#AB*#30#

Code valide ?

[A]
Oui

[B]
Non

```
class X extends X {
    // Class X Member
}
```

Conception

#QDLE#Q#ABC*#70#

Quelles relations ?

[A]
Computer est une superclass
LinuxComputer hérite de AppleComputer
AppleComputer hérite de Computer

[B]
Les 3 classes sont soeurs

[C]
Computer est une superclass
LinuxComputer hérite de Computer
AppleComputer hérite de Computer

```
class Computer
class LinuxComputer
class AppleComputer
```

Surcharge

#QDLE#Q#A*BC#100#

Affichage ?

[A]
Erreur

[B]
1

[C]
2

```
class A {
    private void hello(int i) {
        System.out.println("1");
    }
}

class B extends A {
    public B() {
        hello(2);
    }
}

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

Surcharge

#QDLE#Q#A*BCD#100#

Affichage ?

[A]
2

[B]
Erreur

[C]
3

[D]
1

```
class A {
    protected void hello() {
        System.out.println("1");
    }
}

class B extends A {
    private void hello(int i) {
        System.out.println("2");
    }

    public B() {
        hello(3);
    }
}

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

Surcharge

#QDLE#Q#ABCD*#100#

Affichage ?

[A]
1
2

[B]
1
1

[C]
2
2

[D]
2
1

```
class A {
    protected void hello() {
        System.out.println("1");
    }
}

class B extends A {
    protected void hello() {
        System.out.println("2");
    }

    public B() {
        hello();
        super.hello();
    }
}

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

Surcharge

#QDLE#Q#AB*C#100#

Affichage ?

[A]
1

[B]
2

[C]
Erreur

```
class A {
    public void hello() {
        System.out.println("1");
    }
}

class B extends A {
    public void hello() {
        System.out.println("2");
    }
}

public class MainClass {
    public static void main(String[] args) {
        A a = new B();
        a.hello();
    }
}
```

Abstraction

#QDLE#Q#A*BC#100#

Affichage ?

[A]
Erreur

[B]
2

[C]
1

```
abstract class A {
    public void hello() {
        System.out.println("1");
    }
}

class B extends A {
    public void hello() {
        System.out.println("2");
    }
}

public class MainClass {
    public static void main(String[] args) {
        A a = new A();
        a.hello();
    }
}
```

Abstraction

#QDLE#Q#AB*#100#

Affichage ?

[A]
Erreur

[B]
2

```
abstract class A {
    public abstract void hello();
}

class B extends A {
    public void hello() {
        System.out.println("2");
    }
}

public class MainClass {
    public static void main(String[] args) {
        A a = new B();
        a.hello();
    }
}
```

Abstraction

#QDLE#Q#A*B#100#

Affichage ?

[A]
Erreur

[B]
2

```
abstract class A {
    public abstract void hello();
}

class B extends A {
    public void bonjour() {
        hello();
    }
}

public class MainClass {
    public static void main(String[] args) {
        A a = new B();
        a.hello();
    }
}
```

Abstraction

#QDLE#Q#A*BC#100#

Affichage ?

[A]
Erreur

[B]
2

[C]
1

```
abstract class A {
    public abstract void hello();
}

class B extends A {
    public void bonjour() {
        hello();
        System.out.println("2");
    }

    public void hello(){
        System.out.println("1");
    }
}

public class MainClass {
    public static void main(String[] args) {
        A a = new B();
        a.bonjour();
    }
}
```

Abstraction

#QDLE#Q#A*BC#100#

Affichage ?

[A]
Erreur

[B]
1
1

```
abstract class A {
    public abstract void hello();
}

abstract class B extends A {
    public void bonjour(){
        hello();
    }
}

class C extends B {
    public void hello() {
        System.out.println("1");
        super.bonjour();
    }
}

public class MainClass {
    public static void main(String[] args) {
        A a = new C();
        a.hello();
    }
}
```

Abstraction

#QDLE#Q#AB*#60#

Code ?

[A]
Erreur

[B]
OK

```
abstract class A {
    public abstract void hello();
}

abstract class B extends A {
    public void hello(){

    }

    public abstract void bonjour();
}

class C extends B {
    public void bonjour() {

    }

    public void hello(){

    }
}
```

Abstraction

#QDLE#Q#AB*#60#

Code ?

[A]
Erreur

[B]
OK

```
abstract class A {
    public abstract void hello();
}

abstract class B extends A {
    public void hello(){

    }

    public abstract void bonjour();
}

class C extends B {
    public void bonjour() {

    }
}
```

Polymorphisme #QDLE#Q#A*BCD#120#

Code ?

[A]
b1
b2

[B]
Erreur

[C]
b2
b1

[D]
b2
b2

```
abstract class A {
    public abstract void hello();
}

class B1 extends A {
    public void hello(){
        System.out.println("b1");
    }
}

class B2 extends A {
    public void hello(){
        System.out.println("b2");
    }
}


public class MainClass {
    public static void main(String[] args) {
        A a = new B1();
        b1.bonjour();
        a = new B2();
        b2.bonjour();
    }
```