Conservatoire National des Arts et Métiers
Centre Régional de Basse Normandie
Informatique Avancé, NFP121

Questionnaire à choix multiple sur Java 1.5.
Série 3

Durée : 30 minutes

Documents autorisés : Aucun

Préambule :
Prenez bien le temps de lire les énoncés. Les questions peuvent être traitées dans un ordre quelconque.
Vous devez entourer clairement les numéros correspondant à vos réponses directement sur le support qui doit être rendu pour la correction. Il n'est pas nécessaire d'argumenter vos choix.

---

1. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

class Test1 {
	public void method() {
		System.out.println("1");
	}
}
public class Test extends Test1 {
	public void method() {
		System.out.println("2");
	}
	static public void main(String[] args) {
		Test t = new Test1();
		t.method();
	}
}
```

1. affiche : 1
2. affiche : 2
3. n'affiche rien
4. ne compile pas
5. lance une Runtime error

2. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

class Test1 {
	public void method() {
		System.out.println("1");
	}
}
public class Test extends Test1 {
	public void method() {
		System.out.println("2");
	}
	static public void main(String[] args) {
		Test1 t1 = new Test1();
		Test t = (Test) t1;
		t.method();
	}
}
```

1. affiche : 1
2. affiche : 2
3. n'affiche rien
4. ne compile pas
5. lance une Runtime error

3. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

class Test1 {
	public void method() {
		System.out.println("1");
	}
}
public class Test extends Test1 {
	public void method() {
		System.out.println("2");
	}
	static public void main(String[] args) {
		Test t = new Test();
		((Test) t).method();
	}
}
```

1. affiche : 1
2. affiche : 2
3. n'affiche rien
4. ne compile pas
5. lance une Runtime error

4. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

import java.util.ArrayList;
import java.util.List;
class Test1 {
	public void method() {
		System.out.println("1");
	}
}
public class Test extends Test1 {
	public void method() {
		System.out.println("2");
	}
	static public void main(String[] args) {
		List<Test1> l = new ArrayList<Test1>();
		l.add(new Test1());
		l.add(new Test());
		for (Test1 t: l) {
			t.method();
		}
	}
}
```

1. affiche : 1 1
2. affiche : 2 1
3. affiche : 1 2
4. affiche : 2 2
5. n'affiche rien
6. ne compile pas
7. lance une Runtime error

5. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface ITest {
	private void method();
}

public class Test implements ITest {
	public void method() {
		System.out.println("1");
	}
	public static void main(String[] args) {
		Test t = new Test();
		t.method();
	}
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

6. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface ITest {
	void method();
}

public class Test implements ITest {
	public void method() {
		System.out.println("1");
	}
	public static void main(String[] args) {
		Test t = new Test();
		t.method();
	}
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

7. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface ITest {
	public void method() {};
}

public class Test implements ITest {
	public void method() {
		System.out.println("1");
	}
	public static void main(String[] args) {
		Test t = new Test();
		t.method();
	}
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

8. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface ITest1 {
	public void method1();
}
interface ITest2 extends ITest1 {
	public void method2();
}

public class Test implements ITest2 {
	public void method1() {
		System.out.println("1");
	}
	public void method2() {
		System.out.println("2");
	}
	public static void main(String[] args) {
		Test t = new Test();
		t.method1();
	}
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

9. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface ITest {
	static public void method();
}

public class Test implements ITest {
	static public void method() {
		System.out.println("1");
	}
	public static void main(String[] args) {
		Test t = new Test();
		t.method();
	}
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

10. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface I {
    int k = 1;
}

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

1. affiche : 0
2. affiche : 1
3. ne compile pas

11. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface I {
    int k = 0;
}

public class Test implements I{
    public static void main(String args[]){
      k = 1;
      System.out.println(k);
    }
}
```

1. affiche : 0
2. affiche : 1
3. ne compile pas

12. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface I {
}

class A implements I {
}

class B implements I {
}

public class Test {
	public static void main(String args[]) {
		I i1 = new A();
		I i2 = new B();
		A a = new A();
		System.out.println(i1 instanceof I);
		System.out.println(i2 instanceof B);
		System.out.println(a instanceof I);
	}
}
```

1. affiche : false false false
2. affiche : true true true
3. affiche : false true true
4. affiche : true false true
5. affiche : true true false
6. ne compile pas
7. lance une Runtime error

13. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

interface ITest {
	void method();
}

public class Test implements ITest {
	void method() {
		System.out.println("1");
	}
	public static void main(String[] args) {
		Test t = new Test();
		t.method();
	}
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

14. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

abstract public class Test{
    public Test (int i){
       System.out.println("1");
    }
    public static void main(String[] args){
       Test t = new Test(10);
    }
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

15. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

abstract class SuperClass {
      public void method();
}

public class Test extends SuperClass {
      public void method() {
            System.out.println("1");
      }
      public static void main(String[] args) {
            Test t = new Test();
            t.method();
      }
}
```

1. affiche : 1
2. n'affiche rien
3. ne compile pas
4. lance une Runtime error

16. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

public class Test {
     public static void main(String[] args) {
       Integer i1 = new Integer(127);
       Integer i2 = new Integer(127);
       Long l = new Long(127);
       System.out.println(i1 == i2);
       System.out.println(i1.equals(i2));
       System.out.println(i1.equals(l));
     }
}
```

1. affiche : true true true
2. affiche : false true true
3. affiche : false true false
4. ne compile pas
5. lance une Runtime error

17. Parmi les affirmations suivantes, lesquelles sont valides :

1. Si une classe top-level est déclarée avec le modificateur « public », elle est visible par toutes les classes de tous les packages.
2. Si une classe top-level est déclarée sans modificateur, elle est visible uniquement par les autres classes du package auquel elle appartient.
3. un membre privé est uniquement visible dans la classe où il est déclaré.
4. Les classes top-level ne peuvent être ni privées et ni protégées.

18. N1 - Quelle(s) réponse(s) déclare(nt) correctement un tableau :

1. int arr[] = new int[];
2. float arr[10] = new float[];
3. double []arr = new double[10];
4. double[] arr = new double[10];
5. double arr[] = new double[10];
6. Aucune des autres réponses

19. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

public class Test {
      public void method() throws Exception  {
            System.out.println("1");
            throw new Exception();
            System.out.println("2");
      }
      static public void main(String[] args) {
            Test t = new Test();
            t.method();
      }
}
```

1. affiche : 1 2
2. affiche : 1
3. ne compile pas
4. lance une Runtime error

20. Le programme suivant :

Fichier Test.java :

```
package org.cnam.package1;

public class Test {
	static public void main(String[] args) {
		try {
			int i = 0;
			int j = 1;
			double f = j/i;
			System.out.println("1");
		} catch (Exception e) {
			System.out.println("2");
		} finally {
			System.out.println("3");

		}
		System.out.println("4");
	}
}
```

1. affiche : 1 2 3 4
2. affiche : 2 4
3. affiche : 2 3 4
4. affiche : 2
5. affiche autre chose

21. Le mot clé « transient » :
    1. sert à indiquer qu'un attribut ne fait pas partie de la persistance d'un objet
    2. sert à indiquer qu'une méthode ne modifie pas un objet
    3. n'est pas un mot clé de Java